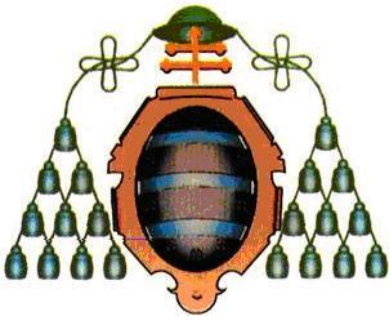

UNIVERSIDAD DE OVIEDO

Departamento de Ciencias de la Educación

Programa de Doctorado: Didácticas de las Ciencias de la Educación

TESIS DOCTORAL

Las Prácticas Cooperativas como mejora del aprendizaje en programación de computadores

Oscar Caneo Salinas

RESUMEN DEL CONTENIDO DE TESIS DOCTORAL

1.- Título de la Tesis	
Español: LAS PRACTICAS COOPERATIVAS COMO MEJORA DEL APRENDIZAJE EN PROGRAMACION DE COMPUTADORES	Inglés: PRACTICES COOPERATIVES AS IMPROVED IN COMPUTER PROGRAMMING

2.- Autor	
Nombre: OSCAR CUSTODIO CANEO SALINAS	DNI/Pasaporte/NIE:
Programa de Doctorado: DIDACTICAS DE LAS CIENCIAS DE LA EDUCACION	
Órgano responsable: CIENCIAS DE LA EDUCACION	

RESUMEN (en español)

Esta investigación aborda, el problema de la enseñanza y el de de la Programación de Computadores (PC). En el detalle, trata con el análisis, el diseño y la aplicación de una intervención basada en el uso de actividades de aprendizaje y trabajo cooperativo, con el propósito de mejorar los aprendizajes de los estudiantes que cursan la asignatura de Fundamentos de programación (FP), en la Carrera de Ingeniería en Informática de la Universidad de Playa Ancha, en la ciudad de Valparaíso, en Chile, la cual históricamente presenta tasas de reprobación que fluctúan entre el 60 y el 75% de los estudiantes.

El aprendizaje de la programación de computadores es una tarea particularmente compleja, ya que demanda el aprendizaje y el manejo de varios conceptos, procedimientos, de lenguajes para la representación de algoritmos y del lenguaje de programación, y de una metodología para tratar con la resolución de los problemas de programación.

Implica también el aprendizaje de estrategias de pensamiento que son propias de la actividad intelectual de diseñar e implementar soluciones a los problemas de programación, en las que se incluyen estrategias para usar y aplicar este conjunto de elementos de manera integrada, y estrategias para controlar y regular el uso y aplicación de estos, con el objeto de asegurar que se diseñan e implementan soluciones que son efectivas para los problemas que se abordan. Todos estos aspectos del aprendizaje son esenciales para desarrollar y perfeccionar las capacidades que se pretenden con esta asignatura.

Conforme a la experiencia, el aprendizaje de estas estrategias y consecuentemente, el desarrollo y el perfeccionamiento de las capacidades para resolver problemas de programación, no se consigue, mediante el uso de las metodologías de enseñanza tradicionales, basadas fundamentalmente en clases expositivas y de presentación de ejemplos de problemas de programación resueltos, ya que con esta forma, no es posible transmitir a los estudiantes los procesos mentales complejos que significan el manejo de las estrategias que hemos señalados, para que sean aprendidos por ellos.

Bajo este contexto, se ha elaborado y aplicado una intervención basada en la incorporación de actividades de aprendizaje y trabajo cooperativo, con las que se ha pretendido mejorar y favorecer las condiciones para que los estudiantes, aprendan los conceptos y los procedimientos, y desarrollen las formas de pensamiento y el manejo de las estrategias que se requieren para resolver problemas de programación, todo ello con el objeto de adquirir, desarrollar y perfeccionar las capacidad para diseñar e implementar soluciones a problemas de procesamiento de datos.

Para valorar los resultados de la intervención, se han tomado en consideración los resultados académicos en términos de estudiantes aprobados y reprobados para el caso del grupo experimental y control, y además se ha valorado la evolución en la calidad del trabajo cooperativo, en base a un conjunto de indicadores que se definieron para tal efecto.

Conforme a estas valoraciones, el grupo experimental presentó mejoras importantes respecto del grupo control, lo que se refleja en que el primero alcanzó un 75,59% de estudiantes aprobados, y el segundo un 39,84%. Por su parte, para el caso del grupo intervenido, se constató que mientras se avanzaba en el desarrollo de las actividades cooperativas, la mayoría de los grupos fueron experimentando mejoras significativas en la calidad del trabajo cooperativo que desarrollaban, lo que se reflejó, en las mejoras paulatinas y crecientes en los resultados de las evaluaciones individuales del grupo experimental por sobre el grupo control.

Estas mejoras también se reflejan en la buena percepción que mostraron los estudiantes respeto de la

forma de intervención, lo que se evidencia en los resultados de la encuesta de satisfacción que se les aplicó al final de la intervención.

RESUMEN (en Inglés)

This investigation addresses the problems found in the teaching and learning of Computer Programming. The details provided include the analysis, the design and intervention based on the use of learning activities and the cooperative work, with the intended purpose to improve the learning process of the students that are enrolled in the course of Fundamentals of Programming, part of the carreer of Engineering of Information Technology from the University of Playa Ancha located in Valparaiso, Chile, which has reported high rates of reprovals, between 60% to 75% of the students

Learning how to program computers is a particularly difficult task that demands a lot of attention in the learning and management of various concepts and procedures needed for the proper representation of algorithms. This also includes the knowledge needed for understanding programming and the methods needed to resolve the current programming issues.

It also involves learning the proper strategies and thought processes that are appropriate for the intellectual activity of designing and implementing solutions of the problems in programming, which include strategies for applying and using these elements in an integrated manner, and strategies for controlling and regulating the use of these elements with the purpose to assure that the design and implementations of these solutions are effective for the problems that are addressed. All these aspects of the learning methods are essential for the development and perfection of the capacities expected in this subject.

Based on the experience, the learning of these strategies and consequently, the development and perfecting of the skills for resolving problems in the programming is not achievable by the use of traditional methods of teaching, based fundamentally in lecture based classes and the presentation of examples of resolved programming issues. With this method is it not possible to transmit to students the complex mental processes that signify the management of the strategies that we have taught them as for them to continue learning on their own.

Under this context, it has been possible to elaborate and apply an intervention based on the incorporation of the activities needed for learning and cooperative works with which it has been pretended to improve and favor the conditions for the students to learn the concepts and procedures, and develop the thought process and the management of the strategies that are required to solve problems in programming. This is all with the purpose to acquire, develop and perfect the capacity to design and implement solutions for problems found in data processing.

To assess the results of the intervention, it has been taken into account the academic results of the accepted students and the students that failed for the experimental group and the control group and it has also been valued the evolution of the quality of the cooperative work, with a set of indicators that are defined for such effect.

According to these valuations, the experimental group presented important improvements in regards to the control group, which is reflected in that the first reached a percentage of 75.59 of approved students, and the second a percentage of 39.84. In the case of the experimental group, it was found that while these were advances in the development of coopertative activities, the majority of the groups were experiencing significant improvements in the quality of cooperative work that was developed, and what was reflected by gradual improvements and increased results in the individual evaluations of the experimental group over the control group.

These improvements also reflect the good perception that the students demonstrated in regards to the form of intervention,that is evident in the results of the satisfation survey that was given at the end of intervention.

SRA. DIRECTORA DE DEPARTAMENTO DE CIENCIAS DE LA EDUCACION/

No se debe esperar que el aprendizaje suceda, se deben desarrollar acciones para asegurar que realmente ocurra.

(Sugata Mitra, 1999)

Dedicado a:

Mi esposa Marianela,

Y mis hijos:
Oscar, Francisca y Esteban.

AGRADECIMIENTOS:

A la Universidad de Oviedo por darme la oportunidad de realizar este doctorado.

Al Doctor Ramón Pérez por su sabiduría y apoyo permanente en el desarrollo de este trabajo.

A mi familia por su permanete apoyo, estímulo, comprensión y cariño en el logro de este doctorado

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE GRÁFICOS

INTRODUCCION

INTRODUCCIÓN

La presente investigación aborda el problema de la enseñanza y el aprendizaje de la Programación de Computadores (PC), en el contexto de un primer curso de programación para estudiantes de la carrera de Ingeniería en Informática.

En lo específico, trata con el análisis, el diseño y la aplicación de una intervención basada en el uso de actividades de aprendizaje y trabajo cooperativo, con el objetivo de mejorar los aprendizajes de los estudiantes que cursan la asignatura de Fundamentos de programación (FP), en la Carrera de Ingeniería en Informática de la Universidad de Playa Ancha, en la ciudad de Valparaíso, en Chile.

Para estos efectos hemos desarrollado una investigación de tipo cuantitativa y cualitativa, basada en el estudio de un caso, con el uso de pre y post-test, en la que han participado un grupo experimental y un grupo control. El grupo experimental estuvo compuesto exclusivamente por estudiantes de la Universidad de Playa Ancha, en tanto el grupo control, estuvo conformado por estudiantes de la Universidad de Playa Ancha, y por estudiantes de la misma carrera y asignatura, pero de la Universidad de Valparaíso, ubicada en la misma ciudad de Valparaíso en Chile.

El perfil profesional del Ingeniero en Informática establece que se pretende la formación de un profesional con una alta capacidad científica, técnica y de gestión, habilitado para analizar, diseñar, implementar, evaluar y proponer soluciones informáticas de software eficientes, y con habilidad para mejorar procesos tecnológicos y de negocios existentes al interior de las organizaciones.

En este contexto, ha de ser un profesional que tenga dominio de distintos tipos de estructuras lógicas para solucionar problemas con el uso del computador, siendo

capaz de manejar diferentes lenguajes de programación, y de usar técnicas y disciplinas afines a los sistemas de información.

Debe disponer de la capacidad para liderar y dirigir proyectos de desarrollo afines a su área de desempeño, disponer de habilidad para el trabajo en equipos multi e interdisciplinarios, y habilidad de auto aprendizaje y de formación continua, que le permitan adaptar y aplicar las técnicas y las tecnologías emergentes en informática, a su quehacer profesional.

Así entonces, para una carrera de informática, la programación de computadores es un núcleo de conocimientos esenciales y fundamentales, ya que la informática como disciplina considera al computador y a la programación, como elementos centrales de todo su quehacer.

Es por esto que la totalidad de los programas de formación profesional en el área de las ciencias de la computación, consideran en sus mallas curriculares una o dos signaturas que tratan de manera específica con la programación de computadores, y existiendo además dentro de estos programas, varias otras asignaturas estrechamente relacionadas con la programación, por lo que los conocimientos, capacidades y habilidades de programación, son fundamentales y esenciales dentro de todo el proceso formativo de estos profesionales.

Los fundamentos para la inclusión de una o más asignaturas de programación, han sido suficientemente argumentados en varios de los documentos de las recomendaciones curriculares para la definición de planes de estudio en Ciencias de la computación, que han sido elaborados de manera conjunta por la Association for Computing Machinery (ACM) y el Institute of Electrical and Electronics Engineers (IEEE).

Estos documentos son el resultado de los análisis y reflexiones que desarrolla de forma permanentemente la comisión denominada “The Joint Task Force on Computing Curricula”, que está constituida por expertos y académicos de varios países, y que se vinculan con el área de formación profesional en Ciencias de la Computación (CcC).

Además, al revisar las organizaciones curriculares de estas titulaciones, se observa que existen varias otras asignaturas que aunque no pertenecen al área específica de conocimientos de programación, se relacionan directamente con ella o requieren de la programación. En estos documentos se entregan los argumentos para considerar que el área de conocimientos de Fundamentos de Programación (FP), tiene un carácter nuclear y por tanto, es ineludible su inclusión en cualquier plan de formación profesional en CcC.

La investigación trata con la primera asignatura de programación, la que dentro de estos planes de formación se denomina generalmente FP (que es nuestro caso), o Introducción a la Programación (IP), o Programación 1 (P1), o Diseño de algoritmos e implementación de programas (DAIP), etc.

Esta asignatura, se encuentra en el primer o segundo semestre de las mallas curriculares, no exige requisitos para cursarla, y desde la perspectiva de los resultados académicos, históricamente produce una alta tasa de reprobación de estudiantes (entre el 60 y el 75%), y junto con ello, una importante tasa de abandono de la asignatura, y en algunos casos de la carrera.

Conforme a la experiencia docente recogida durante a lo menos 10 años a la fecha, dictando asignaturas de programación de computadores en carreras de Ingeniería en informática, así como en otras especialidades de ingeniería, y en algunas carreras de pedagogía, hemos podido constatar las dificultades que tienen los estudiantes para aprender programación de computadores. Pero también hemos experimentado lo complejo que es enseñar y desarrollar en los estudiantes, las habilidades que se requieren para la programación.

Desde una perspectiva general, nuestro juicio es que las dificultades para aprender programación no están en la complejidad de los conceptos y los procedimientos que usa la programación, sino en el aprendizaje y el desarrollo de la capacidad para usarlos y aplicarlos a la construcción de las soluciones a los problemas de programación.

Corroborando nuestra experiencia, en el mundo académico se reconoce que aprender programación de computadores, en cualquier nivel educativo, es una tarea altamente compleja para la mayoría de los estudiantes, cuestión que se ha venido investigando profusamente en los últimos 15 años, principalmente desde la perspectiva de identificar cuáles son las dificultades que hacen que los estudiantes no consigan aprender y desarrollar la capacidad para resolver problemas de programación de computadores.

Sin embargo, existen pocas investigaciones que traten con el problema de cómo enseñar para ayudar a que los estudiantes superen las dificultades y consigan aprender y desarrollar la capacidad de aplicar lo que aprenden en la propia tarea de resolver los problemas.

En el ámbito de los trabajos investigativos dirigidos a identificar las dificultades que tienen los estudiantes para aprender programación podemos citar el de Brooks (1983) que trata con la elaboración de una teoría respecto de la comprensión de los programas computacionales; el de Perkins & Martin (1986), referido a la fragilidad del

conocimiento y la inefectividad de las estrategias de resolución de los programadores novatos; la investigación de Mayer (1987), que trata con los aspectos cognitivos del aprendizaje y el uso de un lenguaje de programación y otro desarrollado por este mismo autor en el año 1989, que trata con los aspectos psicológicos que intervienen cuando los novatos aprenden programación de computadores; la reflexión desarrollada por Dijkstra (1989), entorno a lo que él llama "la crueldad de enseñar programación de computadores"; el trabajo de du Boulay (1989) en torno a la identificación de algunas dificultades en el aprendizaje de la programación; la investigación de Perkins, Hancock, Hobbs, Martin & Simmons (1989), referida a establecer las condiciones para el aprendizaje en los programadores novicios; el trabajo de Brooks (1990), quién categoriza el conocimiento de programación y hace un análisis para su aplicación; la investigación de Détienne (1990), que trata con el enfoque basado en esquemas o planes que es característica del conocimiento de programación de los expertos; la desarrollada por Ormerod (1990), en torno a establecer la relación entre la cognición humana y la programación; el trabajo de Pennington & Grabowski (1990), el que trata con la identificación de las tareas intelectuales en la programación; el desarrollado por Corritore & Wiedenbeck (1991), quienes hacen una descripción de cómo los novicios abordan el aprendizaje cuando tratan con la comprensión de los programas; la investigación de Davies (1993), referida a la relación entre los modelos y las teorías de las estrategias de programación; la investigación de Cañas, Bajo & Gonzalvo (1994), que trata con la relación entre los modelos mentales y la programación de computadores; la de Brusilovsky, Kouchnirenko, Miller, and Tomek (1994), quienes hacen una revisión de la enseñanza de la programación para novicios desde la perspectiva de los enfoques y las herramientas que se han utilizado; el trabajo de Ye & Salvendy (1996), que trata con la identificación de la relación entre el conocimiento de experto versus el de novicios en la programación de computadores, considerando diferentes niveles de abstracción; el trabajo de Fincher (1999), quien intenta responder a la pregunta "¿qué hacemos como docentes cuando enseñamos programación?"; el trabajo de los autores Byrney & Lyons (2001), respecto de los efectos de los atributos de los estudiantes sobre el éxito en programación; la investigación de Milne and Rowe (2002), en torno a la visión que tienen los estudiantes y los profesores respecto de las dificultades para aprender y enseñar programación; el trabajo de Gonzalez (2004), en torno a la idea de utilizar la teoría del constructivismo en un curso de introducción a la programación; y el trabajo de Raadt (2007), en que hace una revisión de las investigaciones australianas respecto de la resolución de problemas en los programadores novatos.

En el ámbito de las investigaciones que han tratado con el problema de cómo enseñar programación podemos señalar el trabajo de Hundhausen (2002), sobre el uso de una herramienta para la visualización de los algoritmos para apoyar la comprensión

de los mismos; el trabajo de Truong, Bancroft & Roe (2005), quienes desarrollaron un sitio Web para apoyar el aprendizaje de los conceptos de programación; el trabajo de los autores Boada, Soler, Prados y Poch (2006), referido al uso de un entorno virtual para apoyar la docencia en un curso básico de programación; el trabajo de los autores Dann, Cooper & Pausch (2006) en el que se propone el uso del lenguaje de programación Alice y su entorno, como recurso para enseñar programación; el trabajo de El-Sheikh (2006) quien propone un ambiente de aprendizaje adaptativo para mejorar las experiencias de aprendizaje en cursos introductorios de programación de computadores; y la investigación de los autores Fernández, Trella, Galindo y Aranda (2006) referido al uso de nuevas tecnologías para la enseñanza de la introducción a la programación.

Para nosotros las razones por las cuales el aprendizaje de la programación de computadores es difícil para la mayoría de los estudiantes, están dadas en un grado menor por el tipo de contenido a aprender, y en un grado mucho mayor, por las dificultades que representa el aprendizaje y el desarrollo de las capacidades para usar estos contenidos en términos de conocimientos, en las tareas de resolución de los problemas. Así también, para nosotros ambas cuestiones y en especial la segunda, complejizan el proceso de enseñanza y el acto de transmitir a los estudiantes la forma de llevar a cabo los procesos de pensamiento, esto es, cuando se aplican los conocimientos y se ponen en práctica las capacidades para resolver los problemas.

Los contenidos con que trata la PC es una mezcla de fundamentos matemáticos y fundamentos de la propia disciplina de la programación, además de ciertos elementos de tipo tecnológicos.

La disciplina de la PC construye sus fundamentos a partir de elementos matemáticos como el concepto de algoritmo y la lógica de primer orden, los cuales se combinan con los fundamentos de una metodología de resolución de problemas, que es propia de la disciplina de la programación.

Por su parte los elementos de tipo tecnológicos tienen que ver con los lenguajes de programación y con el computador como máquina de cálculo que puede ser programada.

La habilidad para resolver problemas de programación, supone disponer de la capacidad para integrar de manera adecuada todos estos elementos, lo que constituye en sí misma la capacidad más compleja de aprender y desarrollar. La ejecución de esta integración requiere de varios procesos mentales, dirigidos fundamentalmente al uso de la metodología y a la construcción y el trabajo con distintos modelos de representación.

Además, si bien al resolver cada problema se usa la misma metodología, la propia solución es el resultado de una integración y organización particular de estos elementos, lo que exige que el aprendizaje de la programación se enfoque desde la perspectiva de la resolución de problemas y no desde la perspectiva de la ejercitación, entendido esto último como el aprendizaje de un procedimiento que se aplica de manera repetitiva para una cierta clase de problemas, como podría ser por ejemplo, aprender a encontrar las raíces que son solución de una ecuación de segundo grado, en cuyo caso lo que se aprende es a aplicar la formula cuadrática (un algoritmo conocido), que es siempre la misma para cualquier ecuación de este tipo.

Como en cualquier aprendizaje referido a la resolución de problemas, lo fundamental está en enseñar cómo se seleccionan, integran y organizan los distintos tipos de conocimientos implicados en un problema particular, y enseñar cuales son las estructuras y estilos de pensamiento que deben usarse durante el proceso de resolución. Reiteramos que para nosotros, estos son los dos aspectos más complejos de la enseñanza y el aprendizaje de la programación.

Conforme a nuestra experiencia, para que los estudiantes aprendan y desarrollen esta capacidad, no basta sólo con que el docente les presente y explique los elementos teórico-conceptuales de la disciplina, y junto con ello les ejemplifique cómo estos elementos se integran y aplican a la resolución de los problemas de programación.

Creemos que para que la enseñanza y el aprendizaje sean más efectivos, se requiere poner a los estudiantes en situaciones intencionadas, mediante las cuales tengan buenas oportunidades para adquirir y desarrollar esta capacidad. Hablamos de situaciones que logren que ellos se impliquen de manera directa en la práctica de resolución de los problemas, hablamos de situaciones diseñadas para que consigan aprender y desarrollar la capacidad para integrar y aplicar los conocimientos, y que de paso, sean instancias que se constituyan en una suerte de laboratorio donde los estudiantes tengan buenas oportunidades para practicar con la ejecución de las actividades de pensamiento que se requieren para la resolución de los problemas de programación.

Así, con el objeto de abordar la complejidad del proceso de enseñanza y aprendizaje de la programación de computadores en el contexto de la asignatura de Fundamentos de programación de la carrera de Ingeniería en informática de la Universidad de Playa Ancha, es que en el desarrollo de esta investigación hemos diseñado y aplicado un conjunto importante de actividades de trabajo cooperativo, con la intención de que los estudiante tengan mejores oportunidades para implicarse y trabajar de manera activa en tareas de resolución de problemas de programación.

Para ello, se ha diseñado y puesto en práctica un plan de clases en el que se integran las clases expositivas dirigidas al tratamiento de contenidos y la presentación de ejemplos para ilustrar el proceso de resolución de problemas, y las actividades de trabajo cooperativo con las que perseguimos el propósito de que los estudiantes, mediante la práctica directa e intencionada con las tareas de resolución de los problemas, obteniendo provecho de la practica personal en la ejecución de estas tareas, pero por sobre todo, sacando provecho del trabajo del grupo, transformándose este en el entorno que facilita y potencia el desarrollo de los procesos de pensamiento, reflexión, discusión, argumentación que se implican en el propio acto de aplicar los conocimientos disciplinares a la tarea de resolución de los problemas.

Durante el desarrollo de las actividades cooperativas, hemos monitoreado el trabajo de los estudiantes con dos propósitos. El primero guiarles y apoyarles en el trabajo aclarándoles dudas cuando lo hemos estimado necesario, y en segundo término, elaborar un registro que nos permita valorizar el desarrollo del trabajo en equipo en tres ámbitos: Interdependencia positiva al interior de los grupos, Relaciones psicosociales dentro del grupo, y Construcción de significados durante el trabajo grupal.

Para estos efectos, hemos definido un conjunto de indicadores para cada uno de estos tres aspectos, y mediante ellos hemos registrado la valorización de ellos, para cada grupo, y en cada una de las actividades cooperativas.

Conforme a nuestras observaciones y valoraciones, hemos podido verificar la evolución en la calidad del trabajo cooperativo que presentaron la mayoría de los grupos, lo cual se vio reflejado en los resultados académicos alcanzados por los estudiantes en las evaluaciones individuales.

Para valorizar las mejoras en los resultados de académicos, definimos un conjunto de competencias en los ámbitos de: Capacidad para comprender y explicar los fundamentos teórico-formales de la programación de computadores, Capacidad para aplicar la metodología de la programación a la resolución de problemas de procesamiento de datos, y Capacidad para construir e implementar programas computacionales.

Los resultados de estas valoraciones muestran que el grupo experimental alcanzo niveles significativamente mejores que los alcanzado por el grupo control, lo que además se confirma con el nivel de aprobación de la asignatura que para el caso del grupo experimental alcanzó al 75.59% en tanto en el grupo control alcanzó solo al 39.84%.

Además, al grupo experimental se le aplicó un cuestionario, con el objeto de conocer la opinión respecto del nivel de satisfacción de los estudiantes para con la experiencia que significó la incorporación y desarrollo de las actividades de trabajo cooperativo, como parte importante de la asignatura. Entre otros hallazgos mostrados por los resultados de la encuesta, destacamos que del análisis se verifica que un porcentaje importante de los estudiantes considera que las actividades de trabajo cooperativo, favoreció de manera importante el logro de sus aprendizajes individuales.

Con el propósito de estudiar y abordar de forma integral la problemática de la enseñanza y el aprendizaje de la programación de computadores, en el contexto de un primer curso de este tipo para la carrera de Ingeniería en Informática, el presente documento lo hemos organizado en siete capítulos.

El capítulo uno, lo iniciamos analizando la compleja combinación de conocimientos que conforma la disciplina de la PC, la cual es de carácter tecnológico, y cuyos fundamentos se encuentran en la matemática, la ingeniería y la tecnología computacional, y donde estos fundamentos se integran a partir de elementos de la teoría matemática, el pensamiento abstracto y las técnicas de diseño, en este capítulo abordamos sus elementos teóricos desde tres perspectivas. En primer lugar analizamos los elementos que configuran la estructura lógico-conceptual de la PC, ámbito en el cual analizamos los aspectos que nos permitan poder caracterizar los elementos conceptuales que configuran esta disciplina. Para estos efectos, hemos organizado este análisis en cuatro aspectos. Primero, situando en términos generales la PC como un área de conocimientos que forma parte de las Ciencias de la Computación (que de aquí en adelante referiremos como CcC), luego abordando la condición de interdisciplinariedad que afecta a la PC, continuando con un análisis de los objeto con que trata la PC, para terminar analizando la dimensión aplicativa que establece la razón de ser de este cuerpo de conocimientos.

En segundo término, tratamos con los elementos que consideramos que deben formar parte de la estructura lógico-cognitiva de quién resuelve problemas de programación. Para ello identificamos y analizamos los tipos de conocimientos que se encuentran implicados en la tarea de resolver problemas de programación. Para estos efectos hemos estructurado el análisis de esta perspectiva en términos de la caracterización de los problemas de procesamiento de datos, para luego dar paso al análisis de cómo los distintos tipos de conocimientos implicados se usan e integran a la hora de aplicarlos en la propia tarea de resolver los problemas de procesamiento de datos.

Finalmente, y apoyados en las dos perspectivas anteriores, desarrollamos el análisis en términos generales del problema de la enseñanza de la PC. Bajo este propósito, analizamos el rol que juegan las estrategias de resolución a la hora de determinar

cómo se utilizan los conocimientos. Además, utilizamos este análisis para discutir cual debe ser a nuestro juicio, el enfoque de enseñanza de la PC.

En el capítulo dos, analizamos el aprendizaje de la disciplina de la PC desde la perspectiva de la propia asignatura de FP, lo que hacemos desde tres ámbitos que nos parecen fundamentales:

1.- Las demandas curriculares, ya que la asignatura debe responder a la definición del tipo de profesional que se quiere formar,
2.- Las exigencias cognitivas, y
3.- La forma en que se desarrolla la docencia en la misma.

Para estos efectos, en primer lugar, analizamos los elementos que caracterizan el área de conocimientos a la cual pertenece esta asignatura, para lo cual usamos como referente el último informe de recomendaciones curriculares desarrollado de manera conjunta por la ACM y la IEEE (CSC2008).

En este informe, junto con presentar los elementos que caracterizan al área de conocimientos de los Fundamentos de Programación (FP), se presentan y justifican los elementos curriculares que se recomiendan tener en cuenta a la hora de diseñar un plan de formación profesional en CcC. En esta misma perspectiva, se entregan poderosos argumentos para respaldar por qué el área de conocimientos de FP debe ser considerada como nuclear, central e ineludible para estos planes de formación profesional.

Para complementar la caracterización de la asignatura, analizamos los propósitos que se persiguen con su desarrollo, y los contenidos que en ella se tratan para alcanzar los objetivos de la misma.

A continuación analizamos los elementos que son propios del desarrollo de la asignatura, en términos de su número de créditos y su carga horaria. En este análisis contrastamos estos aspectos para varias universidades nacionales y extranjeras, junto con el caso de la Universidad de Playa Ancha en Chile, que es dónde hemos realizado la intervención. Luego tratamos con la caracterización general de los docentes que desarrollan la asignatura, principalmente en el contexto de nuestro país.

Este capítulo lo cerramos con el análisis de la problemática del aprendizaje de la asignatura en estudio, para lo cual nos apoyamos tanto en las investigaciones que han sido publicadas en torno al tema, como en nuestra propia experiencia docente en el desarrollo de la asignatura.

En el capítulo tres, desarrollamos el análisis de los principios que consideramos que han de influir en el proceso de aprendizaje de los FP de computadores. Para estos

efectos analizamos en primer término la importancia y relevancia que tiene la asignatura de FP dentro de los cinco programas de formación profesional en computación e informática que hoy son reconocidos por la ACM. Así, apoyándonos en el documento The Guide to Undergraduate Degree Programs in Computing (CC2005) presentamos la definición de los propósitos que deberían alcanzarse con el desarrollo del área de conocimientos de FP.

Basados en esto, analizamos los objetivos y los contenidos de la asignatura FP en el contexto del plan de formación profesional de la Carrera de Ingeniería en Informática de la Universidad de Playa Ancha, que es donde hemos llevado a cabo la intervención. En este ámbito analizamos los bloques temáticos que aborda la asignatura, precisando las intencionalidades que se persiguen con el desarrollo de cada uno de ellos.

A continuación desarrollamos el análisis de los elementos que se han de tener en cuenta para el aprendizaje de los conocimientos y las capacidades que se abordan con la asignatura. Para estos efectos, analizamos los factores que inciden en la complejidad de la asignatura y como consecuencia de ello, las dificultades que la misma tiene para los estudiantes.

Luego, en la idea de trata de manera más detallada estas dificultades, analizamos las características del tipo de actividad cognitiva que tiene lugar cuando se trata con la actividad intelectual de resolver problemas de procesamiento de datos, para lo cual utilizamos un ejemplo que constituye uno de los problemas típicos de procesamiento de datos que abordamos en los cursos de FP.

Hecho este análisis pasamos a establecer los principios básicos para el aprendizaje de las capacidades que se requieren para el diseño e implementación de soluciones a los problemas de procesamiento de datos.

Finalizamos este capítulo con el análisis de las estrategias metodológicas que, a nuestro juicio, deberían tenerse en cuenta para llevar a cabo las tareas intelectuales para la resolución de problemas de programación.

En el capítulo cuatro, analizamos los elementos que nos permiten proponer un modelo alternativo de enseñanza y aprendizaje para la asignatura, el que sustentamos en los elementos de análisis que hemos planteado en los capítulos anteriores.

En estos términos, como quedará establecido hasta este punto, aprender los FP de computadores en el contexto de las demandas de un primer curso de Programación de una carrera de ciencias de la computación, implica por una parte, aprender y comprender las bases teórico-formales, los procedimientos y las técnicas en los

cuales se sustenta la disciplina de la programación, pero también implica el aprendizaje, el desarrollo y el perfeccionamiento de las capacidades y habilidades para aplicar las bases teórico-formales, los procedimientos y las técnicas, para diseñar e implementar soluciones a problemas de procesamiento de datos, cuestión que como hemos planteado, es el objetivo sustantivo de este tipo de asignatura.

Conforme a los argumentos y análisis presentados, en el logro de estas capacidades y habilidades, se encuentran implicados el aprendizaje de tres tipos de conocimientos; los de tipos declarativos y conceptuales, los de tipo procedimentales, y aquel tipo especial de conocimiento, que se denomina conocimiento estratégico, es el conocimiento que se construye como resultado de la propia práctica en la resolución de los problemas de procesamiento de datos.

Argumentamos que el conocimiento estratégico, es un conocimiento que el resolutor va construyendo a través de la propia práctica en la resolución exitosa de problemas, constituyéndose de paso, en una fuente de conocimiento adicional de la cual podrá disponer para su aplicación, cuando trate con problemas similares, o cuando trate con problemas en los cuales una parte de la solución se corresponda con una solución ya construida y/o conocida. Por tanto, decimos que la construcción de este tipo de conocimiento tiene un carácter recursivo.

En el ámbito de los objetivos que se persiguen con este primer curso de programación de computadores, la adquisición y el manejo del conocimiento declarativo se relaciona con el aprendizaje de las bases teórico-formales de la disciplina. Por su parte, la adquisición y el manejo del conocimiento procedimental está referido al aprendizaje de los procedimientos y las técnicas de programación. Y para nosotros, la adquisición del conocimiento estratégico se alcanza en la medida que se van desarrollando y perfeccionando las capacidades y habilidades para usar y aplicar los conocimientos declarativos, procedimentales y las técnicas, para diseñar e implementar soluciones a problemas de procesamiento de datos.

A la luz de los elementos de análisis y basados en nuestra experiencia presentamos los argumento para establecer que el desarrollo de las capacidades para aplicar los conocimientos teórico-formales, los procedimientos y las técnicas de diseño e implementación de soluciones a los problemas, constituye la mayor dificultad de los estudiantes en este tipo de cursos.

Conforme a lo que se expone y analiza en este contexto, el desarrollo de estas capacidades está asociado con la habilidad para poner en juego un conjunto de estrategias cognitivas y estrategias de pensamiento al momento de desarrollar las tareas de búsqueda, diseño e implementación de soluciones a los problemas de procesamiento de datos. Hablamos de estrategias dirigidas a identificar, seleccionar

y organizar los conocimientos declarativos, procedimentales y los estratégicos cuando se trabaja en la resolución de un problema.

Estas estrategias cognitivas y de pensamiento se han de poner en juego también, para comprender y manipular distintas abstracciones y para trabajar con distintas formas de representaciones y modelos de estas abstracciones, siempre desde la perspectiva de construcción del conocimiento estratégico.

Finalizamos, este apartado, analizando las limitaciones del método de enseñanza que se utiliza y que se ha utilizado para desarrollar la asignatura de FP, las que se refieren esencialmente a que no se favorece el aprendizaje, el desarrollo y el perfeccionamiento de la capacidad para resolver los tipos de problemas con que trata la asignatura.

Dadas, las características complejas de los conocimientos y capacidades que demanda la asignatura exige la puesta en práctica de metodologías que consigan que los estudiantes adopten un rol activo en sus procesos personales de aprendizaje, como condición para alcanzar aprendizajes de mejor calidad.

Enfatizamos de que tratamos con un aprendizaje que es complejo no solo por lo tipos de conocimientos que se deben aprender, sino que por sobre todo, porque se requiere desarrollar la capacidad para utilizar estos conocimientos al aplicarlos a situaciones concretas de resolución de problemas de procesamiento de datos y la programación de computadores, contexto para el cual, además de los conocimientos, se requiere de determinadas estrategias de pensamiento que son de carácter complejo.

Así, presentamos los argumentos para establecer la importancia de crear condiciones que permitan abordar las complejidades de estos aprendizaje, para lo cual proponemos utilizar una estrategia de aprendizaje y trabajo cooperativo, y conforme a ello, establecemos las orientaciones para el diseño del plan de desarrollo de la asignatura, el cual además de las clases expositivas para la presentación y ejemplificación de los usos y la aplicaciones de los contenidos, se han de considerar un conjunto importante de actividades de trabajo cooperativo cuya intencionalidad es que nuestros estudiantes adquieran, desarrollen y perfeccionen las capacidades para resolver problemas de programación.

En el capítulo cinco, abordamos los aspectos referidos al experimento que se ha llevado a cabo en esta investigación.

En estos términos exponemos los objetivos generales y específicos de la investigación, y planteamos las hipótesis de la misma.

Junto con ello describimos los participantes, el procedimiento que hemos diseñado y aplicado, ámbito en el cual describimos en detalle el plan de clases que se ha puesto en práctica con el desarrollo de la asignatura.

Seguidamente exponemos los detalles de las variables y sus medidas, y presentamos los instrumentos que hemos utilizado para la recopilación de la información, y detallamos el diseño del experimento que hemos llevado a cabo.

En el capítulo seis, presentamos los resultados de la intervención los cuales hemos organizado en la siguiente secuencia:

- Resultados de la evaluación de conocimientos previos,
- Resultados referidos al logro de competencias;
- Resultados referidos a las tasas de aprobación y;
- Resultados referidos a la evolución de los indicadores del trabajo cooperativo; y
- Resultados referidos a la encuesta de satisfacción.

En el capítulo siete y final, presentamos las conclusiones e implicaciones de la investigación.

CAPITULO 1

LA DISCIPLINA DE LA PROGRAMACIÓN DE COMPUTADORES (PC) Y SUS ELEMENTOS TEÓRICOS

1. LA DISCIPLINA DE LA PROGRAMACIÓN DE COMPUTADORES (PC) Y SUS ELEMENTOS TEÓRICOS.

INTRODUCCIÓN

La PC es una disciplina tecnológica, que posee una compleja combinación de conocimientos, cuyos fundamentos se encuentran en la matemática, la ingeniería y la tecnología computacional.

Su campo aplicativo se relaciona con el diseño e implementación de programas computacionales, para resolver problemas de procesamiento de datos.

En la idea de dilucidar y comprender la compleja combinación de conocimientos que posee la disciplina, en este capítulo tratamos en primer término, con la clarificación de la estructura lógico-conceptual de la PC.

Luego de lo anterior y atendiendo al carácter aplicativo de la disciplina, intentamos clarificar cuál es la estructura lógico-cognitiva que entra en juego cuando los conocimientos disciplinares se aplican a la tarea de resolver problemas de procesamiento de datos.

Desarrollados los dos aspectos anteriores, concluimos este capítulo, con una primera aproximación al problema de la enseñanza de los conocimientos de la disciplina, en el contexto de un primer curso de programación para estudiantes de Ingeniería en Informática.

1.1. Conocimientos de la PC y su estructura lógico-conceptual

1.1.1. Perspectiva general de la PC.

Describir la estructura de conocimientos de la PC es una tarea particularmente compleja, ya que en su estado actual, esta estructura está compuesta por

conocimientos que tienen orígenes diversos tanto en lo referido al campo disciplinar al cual pertenecen, como en lo referido a la cronología en que ellos se fueron gestando. Además, algunos de estos conocimientos se desarrollaron a partir de intencionalidades por encontrar soluciones a problemas que eran un tanto distintos que los problemas que preocupan hoy a la PC.

Así, en su estado actual, esta estructura debe ser entendida como la consecuencia de un proceso de evolución que se mantiene aún en nuestros días, y que ha estado condicionado por la necesidad de ir dando respuesta y ofrecer soluciones a problemas de complejidad creciente, y para diversas áreas de aplicación.

Desde la perspectiva aplicativa, tanto en sus inicios como en su estado actual, la estructura de conocimientos de la PC está compuesta por dos categorías de conocimientos:

1.- Aquellos que son de utilidad para el diseño de soluciones a los problemas, y
2.- Aquellos que sirven a la implementación de las soluciones expresadas como programas computacionales.

La primera categoría de conocimientos, se relaciona principalmente con los fundamentos para construir modelos abstractos para representar la solución de un problema, en tanto los conocimientos de la segunda categoría, se relacionan con la tecnología computacional que permite transformar los modelos en un programa computacional.

Las primeras teorizaciones que conforman la base formal y conceptual de la PC, comenzaron a desarrollarse mucho antes de la puesta en funcionamiento del primer computador electrónico con capacidad para ser programado. Tal es el caso de la teoría de la computabilidad, la que es el resultado de los trabajos de Turing (1936) y Gödel (1934), en torno al propósito de hacer precisar la noción de función efectivamente calculable.

Estos trabajos, hicieron posible la conceptualización de algoritmo, entendido éste como la definición de un procedimiento, generalmente matemático, para la obtención de ciertos resultados, entendiendo que una de las características más importante de estos procedimientos, es que ellos pueden ser ejecutados de manera manual por un resolutor humano, o de manera automática mediante el uso de una máquina hipotética, que en nuestro caso es el computador.

Para la PC, este concepto es una idea poderosa, ya que constituye la base formal para tratar con el diseño de secuencias de acciones y operaciones (un procedimiento), que permitan expresar la solución a un determinado problema, a partir de la cual se materialice la escritura de un programa computacional.

Los primeros fundamentos teóricos de la PC fueron aportados por Ada Lovelace, quien en 1843, publicó una serie de influyentes notas sobre la máquina analítica que había comenzado a diseñar Charles Babbage a partir del año 1817, el cual siguió perfeccionando hasta su muerte en el año 1871. En razón de esto, Ada Lovelace es considerada la primera programadora en la historia de la computación.

Este conjunto de conocimientos, dio origen a la conceptualización de un modelo de computador, lo que sirvió de base para el diseño de los primeros computadores electrónicos programables, construidos por Howard Aiken y Konrad Zuse en el año 1947, modelo que posteriormente fue complementado y perfeccionado por John von Neumann en el año 1949.

El modelo de computador juega un rol fundamental en la conceptualización de una máquina, que puede ser programada para realizar de manera automatizada, variadas acciones y operaciones, con lo cual esta máquina tiene la potencialidad de ejecutar los algoritmos.

Encontramos aquí, tres de los elementos más fundamentales de la PC, el primero es el concepto de algoritmo, el que constituye la base teórica en la que se sustenta el formalismo para la definición y la descripción, de secuencias de acciones y operaciones, que permiten modelar procesos de transformación de información.

El segundo elemento es el modelo de computador, el cual constituye la base para conceptualizar una máquina que tiene la capacidad de ejecutar las secuencias de acciones y operaciones.

El tercer elemento es el lenguaje de programación, el cual constituye la base teórico-conceptual para la definición de un conjunto de reglas sintácticas, semánticas y de un lenguaje, para escribir los algoritmos como un programa que sea entendible para la máquina computador, esto como condición para que dicho algoritmo pueda ser efectivamente ejecutado por ella.

En su estado actual, la PC es reconocida como un cuerpo, que se orienta al estudio sistemático de los procesos algorítmicos que describen y transforman información y sus correspondientes implementaciones como programas computacionales, los cuales, han de poder ser efectivamente ejecutados por un computador real.

Por tanto su estructura de conocimientos está compuesta por fundamentos teóricos-formales, y por la definición de determinados procedimientos prácticos para la aplicación de estos fundamentos, con el objetivo de diseñar e implementar soluciones computacionales a situaciones concretas de procesamiento de datos. Además, en su dimensión aplicativa, intervienen fundamentos teóricos-formales y procedimientos que son propios de la tecnología computacional.

Por tanto, la PC es un cuerpo de conocimientos de orientación tecnológica, cuyos fundamentos se encuentran en la matemática, en la ingeniería, y en la tecnología computacional. Para Denning (1999) la PC es un cuerpo disciplinar en el que confluyen la teoría, el pensamiento abstracto, y técnicas de diseño de carácter específico.

La teoría tiene su origen en la matemática y de ella se toma la conceptualización de algoritmo, lo que permite trata con la caracterización de los objetos que intervienen en la solución, hipotetizando las relaciones entre ellos, determinando la veracidad de esas relaciones, e interpretando los resultados.

Para estos efectos, un algoritmo constituye un modelo en el que se relacionan los objetos que conforman el espacio de un problema, y que al ser organizados de manera lógica, contienen y expresar el diseño de una solución para un problema de procesamiento de datos.

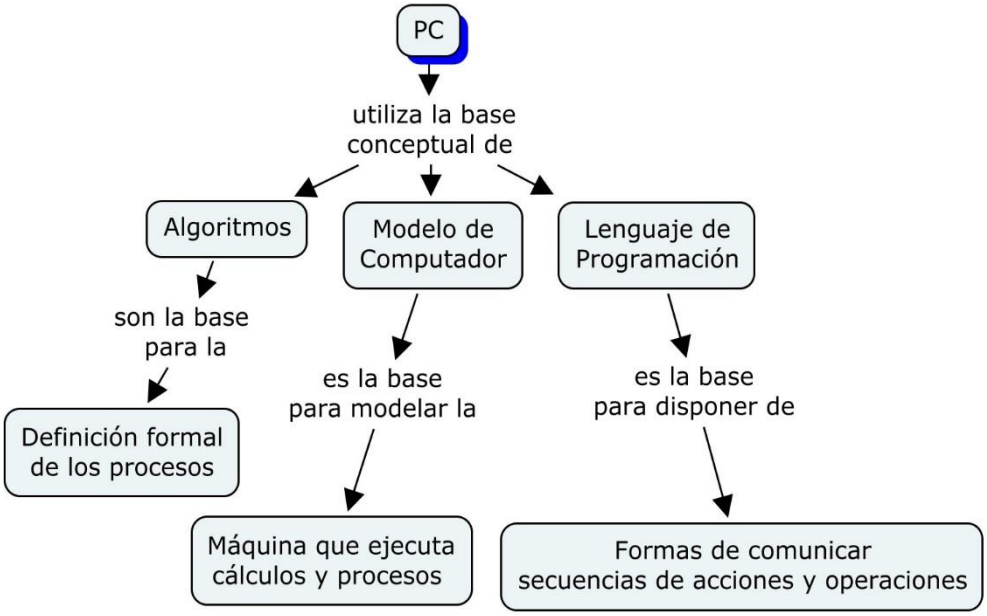

Figura 1: Elementos Teórico-Conceptuales de la PC (Caneo, 2009)

El pensamiento abstracto, lo entendemos como la capacidad de deducir, sintetizar, interpretar, analizar, y modelar las situaciones problemas, o parte de ella.

Cuando se trata con los procesos de búsqueda y diseño de soluciones a un problema de procesamiento de datos, la construcción de modelos es esencial para comprender las situaciones problemas, y para proponer soluciones al mismo. Para estos efectos, la elaboración de una abstracción se rige por un proceso que parte de una propuesta de hipótesis respecto de la situación, para dar paso a la construcción de uno o más modelos.

Estos modelos se prueban para analizar sus resultados, y así evaluar su validez. Esta forma de pensamiento es esencial para la PC, ya que constituye el formalismo para la construcción de los modelos del espacio del problema, el modelo de solución del problema, y el modelo de implementación computacional de solución al problema.

La ejecución efectiva de los actos de pensamiento, dirigidos a la construcción de estos distintos modelos, es el aspecto más crítico de la tarea de resolución de los problemas de programación, puesto que, basta con que sólo uno de estos modelos sea incompleto o errado, para que el proceso completo falle.

Desde la perspectiva del proceso de enseñanza y aprendizaje, este es el aspecto más difícil de enseñar y aprender, ya que implica el uso de estilos de pensamiento de orden superior, como es la capacidad para crear y trabajar con abstracciones.

Por su parte, las técnicas de diseño tienen su origen en la ingeniería y en la tecnología computacional, y son necesarias para el diseño y construcción de las soluciones a los problemas, y para la implementación del programa computacional de solución. Estas técnicas se materializan en la metodología de resolución de los problemas de procesamiento de datos, la cual define las etapas y los procedimientos técnicos y prácticos para ejecutarlas.

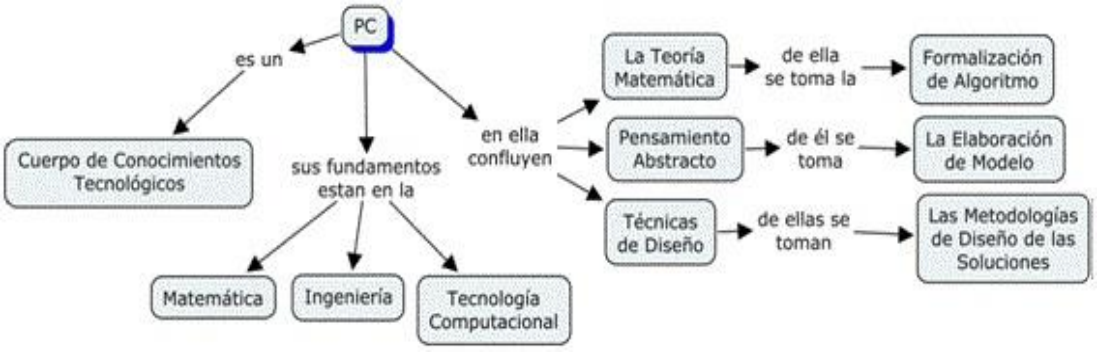

Figura 2: Elementos que intervienen en la PC (Caneo, 2009)

1.1.2. Interdisciplinaridad de la PC.

En la actualidad se continúa debatiendo respecto de los méritos relativos de la matemática, la ciencia y la ingeniería, y respecto de cuál de estas tres componentes, la teoría, el pensamiento abstracto para la construcción de modelos o el diseño, es el más fundamental para la PC y su campo aplicativo.

Nosotros creemos que estos tres elementos están intrínsecamente entrelazados. Ejemplos de la teoría aparecen cuando se elabora el pensamiento abstracto y el diseño; ejemplos de modelamiento aparecen cuando hacemos uso de la teoría y el diseño. En tanto el uso de las técnicas de diseño implican también el uso de la teoría y el pensamiento abstracto.

A pesar de su inseparabilidad, estas tres fuentes de conocimientos son distintas, puesto que cuando se aplican, cada una da sustento a un proceso diferente. Así:

1.- La teoría da sustento al proceso de describir y probar las relaciones entre los objetos,
2.- El pensamiento abstracto, da sustento al proceso de identificar y usar las relaciones entre los objetos, para hacer predicciones que puedan ser comparadas con situaciones del mundo real, es decir, da sustento al proceso de elaboración de los modelos del espacio del problema, de solución del problema, y de implementación del programa computacional de solución del problema, y
3.- El diseño se relaciona con el proceso de implementar casos específicos de las relaciones y usarlas para realizar acciones útiles, como por ejemplo, diseñar las soluciones.

Como se observa, si bien es posible identificar situaciones en que interviene la teoría, el pensamiento abstracto y el diseño, estos tres elementos son inseparables cuando se trata con el proceso mismo de resolución de los problemas y la implementación de los programas de solución.

La PC se ubica entonces, entre el proceso central de la matemática aplicada, las ciencias, la ingeniería y la tecnología. Como hemos señalado, para nosotros en la PC estos tres procesos se encuentran en el mismo nivel de importancia fundamental, por lo que esta es una disciplina en la que se produce una mezcla de conocimientos y de interacciones entre la teoría, la abstracción y el diseño, y el elemento central, es el interés común en la experimentación y el diseño como transformación de objetos, en el que interviene el soporte de la tecnología computacional para el desarrollo de ciertas etapas de estos procesos.

1.1.3. Los objetos con que trata la PC.

Para Heckhausen (1975), una de las formas de analizar una estructura de conocimientos, consiste en revisar los objetos de estudio con que trata el cuerpo de conocimientos.

En la figura 3 en la siguiente página, presentamos los objetos que forman parte de la disciplina y las relaciones entre ellos.

Luego de ella, describimos estos objetos y la forma en que ellos aportan a la disciplina.

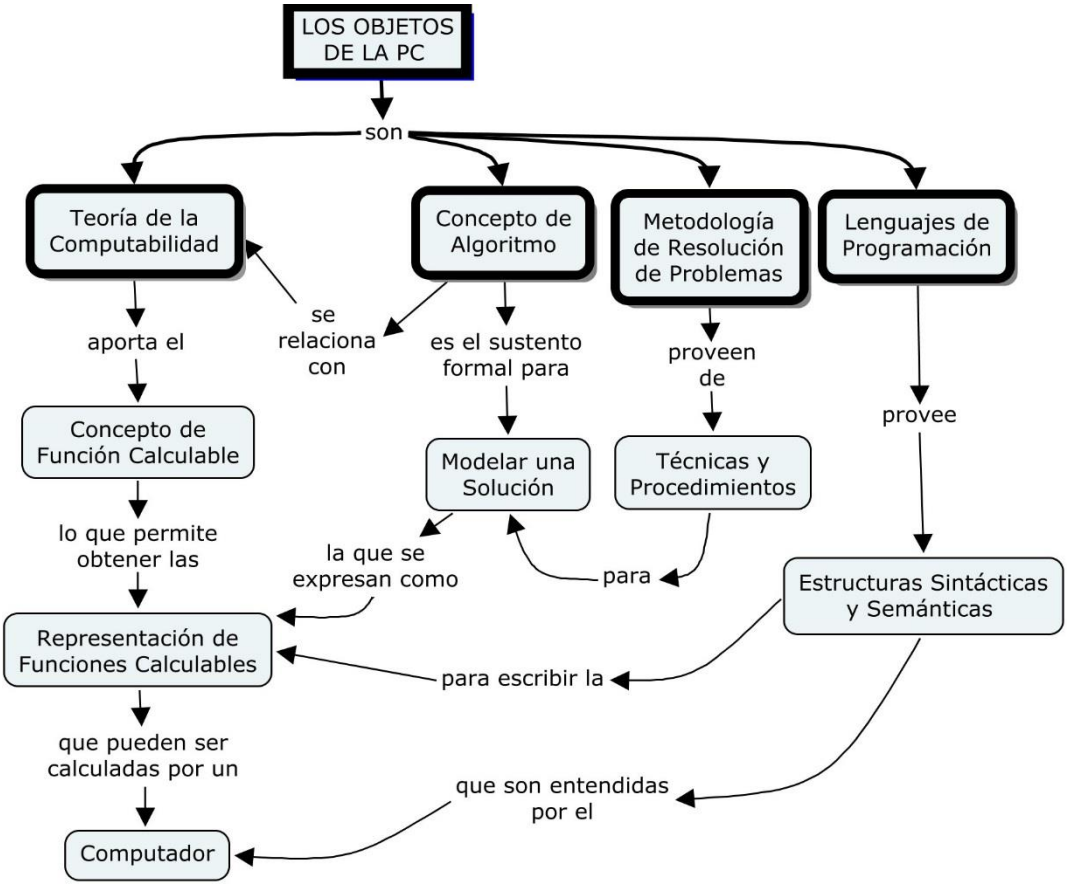

Figura 3: Relaciones entre los objetos con que trata la PC (Caneo, 2009)

1.1.3.1. La teoría de la computabilidad.

La teoría de la computabilidad, es el marco de referencia formal para establecer el concepto de función calculable, esto es, la conceptualización de funciones cuyos valores pueden ser calculados de forma automática, o efectiva, mediante un algoritmo, lo que se denomina, teoría de la computabilidad efectiva.

En palabras simples, esta teoría establece que es posible definir una función que sea efectivamente calculable, si existe un algoritmo para el cálculo de sus valores que sea a su vez efectivo.

Esta teoría tiene dos aspectos:

A.- Define una construcción racional de lo que efectivamente es una función calculable, y

B.- Da cuenta de una hipótesis de tipo empírico-matemática vinculada con la obtención de los resultados del cálculo efectivo de la función.

De esto último, para la PC es de utilidad el aspecto representacionista de una función, esto es, la descripción de ella mediante el uso de un lenguaje interpretado, con lo

cual se permite que el procedimiento para obtener los resultados de la función calculable, pueda ser expresado mediante un algoritmo y consecuentemente como un programa computacional.

Sin embargo para hacer posible su aplicabilidad, esta teoría se sustenta en el objeto computador, que es el recurso material tecnológico que permite la ejecución computacional de la función calculable.

Por tanto, esta teoría es de gran importancia para la PC, no solo por la formalización de lo que se ha de entender por función efectivamente calculable, sino también por la relación que se da entre esta teoría y los algoritmos, como mecanismo para presentar la descripción del procedimiento para calcular una función, haciendo de paso explicita la necesidad de la existencia de determinados lenguajes, uno algorítmico y otro computacional, y del computador como máquina que es capaz de ejecutar el procedimiento definido algorítmicamente.

En estos términos, el cómputo de una función se lleva a cabo mediante la ejecución de pasos de un procedimiento, por lo que el cálculo efectivo de una función, supone la aplicación de un método algorítmico que expresa una secuencia de pasos que incluyen acciones y operaciones, y que organizados conforme a una lógica de ejecución, definen el procedimiento para calcular la función.

Así, utilizando el marco conceptual de la teoría de la computabilidad, es posible establecer que cada uno de los pasos de un cómputo efectivo, será a su vez efectivo; y esto es así porque las reglas que se han de utilizar en cada uno de los pasos de los cómputos, tendrán que ser también efectivas, con lo cual la computabilidad rechaza de manera implícita el hecho de que cada paso no constituya un cálculo efectivo o no realizable.

En resumen, un cómputo efectivo involucrará reglas efectivas, para lo cual, estas reglas han de ser componentes de un método o de un sistema formal, y una serie de pasos efectivos, que permita calcular el valor de una función para un argumento dado.

Esta base formal es de gran importancia para pensar en las características que ha de tener el diseño de los procedimientos algorítmicos para resolver problemas de procesamiento de datos, ya que con ello se determina que, no sólo el procedimiento algorítmico ha de ser efectivo, sino que también, cada paso del procedimiento algorítmico ha de serlo.

Para la PC, estas condiciones son fundamentales para pensar en las características que han de tener los procedimientos algorítmicos para que estos puedan ser

expresados mediante un lenguaje computacional, y finalmente, para que puedan ser ejecutados por una máquina real como el computador.

1.1.3.2. La importancia del concepto de algoritmo para la PC

Conforme a lo que hemos planteado en los párrafos anteriores, el concepto de algoritmo está fuertemente relacionado con la teoría de la computabilidad, y es la base formal para tratar con el modelamiento de las soluciones a los problemas.

Para la PC, este concepto tiene dos dimensiones, la formal y la aplicativa. La formal está determinada por el propio concepto y las caracterización de lo que debe entenderse por un algoritmo. En su dimensión aplicativa, está relacionada con las formas de representación y expresión y con el lenguaje algorítmico.

Este formalismo, es uno de los componentes más esenciales en la conformación de la estructura del conocimiento respecto de la PC, ya que constituye la base conceptual en la cual se sustenta, la definición de las características, que debe tener el diseño de los procedimientos lógico-matemáticos mediante los cuales, se da solución a problemas de procesamiento de datos. Con esto, se asegura que estos procedimientos poseen las características que son propias de un algoritmo, con lo cual se garantiza que ellos podrán ser ejecutados por una máquina, como el computador.

Estas dos dimensiones son complementarias, en el sentido que la dimensión formal, asegura la dimensión aplicativa, la cual se manifiesta en la posibilidad de que los procedimientos lógico-matemáticos, puedan realmente ser ejecutados por el computador o por una máquina equivalente.

Visto esto desde la perspectiva del aprendizaje de la PC, las dos dimensiones son también fundamentales. Para quien aprende a programar computadores, es esencial dominar la significancia del concepto de algoritmo, ya que este constituye el referente formal que se debe tener en cuenta para diseñar los procedimientos lógico-matemáticos de los que hablamos, con lo cual se asegura que estos procedimientos (los algoritmos), podrán ser ejecutados por el computador programable.

Sin embargo debemos precisar que, aun cuando para la solución de un cierto problema, se haya elaborado un algoritmo conceptualmente correcto, esto no asegura que el diseño lógico-procedimental lo sea, con lo cual el procedimiento de solución expresado mediante él, no será efectivo para resolver el problema, cuestión que ocurre cuando la organización lógica de las acciones del procedimiento, o las propias acciones contenidas en el algoritmo, no son adecuadas para obtener los resultados que demanda la solución.

Dicho en términos más simples, el procedimiento lógico-matemático puede ser ejecutado por la máquina computador, pero sin embargo, los resultados que arroja su ejecución no son los que se esperan para dar una solución adecuada al problema. Asegurar la efectividad del procedimiento diseñado para resolver un cierto problema de procesamiento de datos, representa una dificultad importante en el aprendizaje de la PC.

1.1.3.3. Los lenguajes de programación.

Un lenguaje de programación es una construcción de significación sintáctica y semántica, de tipo artificial que ha sido creada para expresar las computaciones, y para que ellas puedan ser interpretadas y ejecutadas por un computador, es decir, es el medio que permite traducir las acciones del procedimiento lógico-matemático, contenidas en un algoritmo, para expresarlas en un lenguaje que los computadores puedan reconocer y ejecutar.

Los constructos de un lenguaje de programación permiten expresar como instrucciones, las secuencias simples de acciones de un algoritmo, y disponen de mecanismos para expresar acciones de control sobre conjuntos de instrucciones, conformando así secuencias más complejas.

Así entonces, el lenguaje de programación es el objeto que hace posible que los modelos de experimentación de las soluciones a un problema, puedan ser probadas en una máquina real como es el computador, constituyéndose entonces, en el vehículo que automatiza la ejecución de las soluciones diseñadas, con lo cual éstas pueden ser probadas y verificadas de una forma directa.

1.1.3.4. La metodología de resolución de problemas de procesamiento de datos.

La metodología de resolución de problemas de procesamiento de datos, es el objeto central y nuclear de la PC en el sentido que en ella se establece, el marco de referencia formal y procedimental, de la técnica que se aplica para buscar e implementar las soluciones a los problemas de procesamiento de datos. En esta perspectiva, es el objeto que actúa como integrador de los demás objetos.

En el estado actual de desarrollo de los conocimientos de la PC, este marco de referencia es fundamental, ya que aunque en sus inicios, la PC sólo trataba con la implementación de soluciones a problemas basados esencialmente en cálculos de expresiones algebraicas, hoy en día los problemas con que trata la disciplina, provienen de áreas muy diversas, como por ejemplo el cálculo de predicados, el procesamiento inferencial de bases de conocimientos, la minería de datos, la robótica, etc.

Por tanto, para tratar con estas nuevas áreas, la PC ha debido experimentar con nuevas metodologías, que sean más efectivas para la búsqueda, el diseño y la implementación de las soluciones, lo que también, ha debido ser complementado con el desarrollo paralelo de nuevos lenguajes de programación, que permitan realizar las implementaciones de estos tipos de procesamientos. Es así como, uno de los campos investigativos de la PC que se encuentra en permanente evolución, es precisamente el referido a las metodologías de resolución de los problemas.

Así, la metodología actúa como el elemento que integra los objetos señalados, ya que es la formalización de un conjunto de procedimientos, que establecen un método ordenado y sistemático, para abordar el proceso de resolución de los problemas de procesamiento de datos, y por tanto, constituye el método científicamente concebido para resolver esta clase de problemas.

Desde la perspectiva instruccional, el aprendizaje de la PC demanda en primer término, el dominio de la significancia de los aspectos formales de los objetos que hemos descrito aquí, pero también, exige del dominio de los procedimientos para utilizar estos formalismos, cuando se aplican al diseño e implementación de soluciones a problemas de procesamiento de datos.

Por tanto, es un aprendizaje que se orienta al dominio de procedimientos de orden práctico que tienen una base conceptual y formal, cuyas significancias también deben ser dominadas.

1.1.4. Aplicabilidad de la PC

Como se ha planteado, la PC constituye un cuerpo de conocimientos cuya orientación fundamental, es el tratamiento de los problemas de procesamiento de datos, específicamente, el tratamiento de las tareas de diseño e implementación de soluciones computacionales para esta categoría de problemas.

Para Barchini, et. al. (1998), la PC es un cuerpo de conocimientos de carácter científico tecnológico y que además está compuesto por elementos multidisciplinares que se integran e interrelacionan.

1.1.4.1. Los conocimientos conceptuales

En su componente conceptual y formal, la PC se aboca al estudio de los fenómenos relacionados con el diseño y la implementación de soluciones a problemas de procesamiento de datos, para lo cual, cuenta con una base conceptual, y con métodos y procedimientos que permiten captar y estudiar los fenómenos de su área de competencia.

Por otra parte, es un cuerpo de conocimientos que se encuentra en permanente evolución, mediante transformaciones que se han establecido como necesarias, para abordar de manera efectiva y eficaz, los problemas de procesamiento de información, en áreas diversas y de complejidad creciente.

Esta evolución, es consecuencia de las experiencias que se generan a partir de la propia aplicación del cuerpo de conocimientos, y que impactan al conjunto de sus elementos constitutivos, como sus objetos de estudio, su base conceptual y sus metodologías. El ejemplo más concreto de estos cambios evolutivos son los estilos de programación, los cuales han ido desde el modelo de programación imperativa, pasando por el modelo de programación basada en eventos, hasta los actuales modelos de programación orientada a objetos.

Para responder a su carácter aplicativo y práctico, la PC se configura a partir de dos cuerpos de conocimientos que se encuentran en estrecha relación de dependencia y utilidad. Por una parte están los conocimientos que son de utilidad para la búsqueda y diseño de soluciones a los problemas, y por otra, los conocimientos que son de utilidad para realizar los procesos de implementación de los programas de solución.

1.1.4.2. Los conocimientos para la búsqueda y el diseño de soluciones

Con ellos se establecen los fundamentos metodológicos y los aspectos procedimentales, para elaborar y trabajar con los modelos de representación. En términos más específicos, estos modelos de representación permiten tratar con los distintos niveles de complejidad del espacio del problema, para ayudar a la compresión del mismo, transitando desde una situación problema planteada en términos abstractos y con muchas incertidumbres (el enunciado del problema), hasta una situación concreta, que es cuando se ha conseguido diseñar un algoritmo (un modelo de procedimiento), para resolverla.

1.1.4.3. Los conocimientos para la implementación del programa

Son de utilidad para obtener la automatización del procedimiento algorítmico de solución. Como ya se ha planteado, en el ámbito de la PC, el fin último del proceso de resolución de los problemas, es la obtención de un programa computacional que permita automatizar la solución del problema, bajo la condición de que dicho programa, pueda ser ejecutado en un computador real, para lo cual se requiere tratar con el modelo de computador y con el modelo de programa en ejecución.

En estos términos, en este cuerpo de conocimientos se encuentran los fundamentos y los procedimientos que permiten convertir la representación algorítmica de la solución del problema, en un programa computacional, para lo cual, se requiere del uso de un lenguaje de programación y de un entorno de programación.

Para estos efectos, se integran los conocimientos matemáticos y los conocimientos de la CcC. Con la integración de estos conocimientos, la disciplina dispone de los elementos teórico-formales y de los procedimientos, que permiten dar el rigor disciplinar al desarrollo práctico de los procesos para buscar y diseñar soluciones a los problemas de procesamiento de datos y para implementar los programas de solución correspondientes.

1.1.4.4. Los conocimientos de la metodología de la PC

La metodología integra todos los conocimientos anteriores, y por tanto, es el bloque de conocimientos que da sustento al proceso completo de resolución de los problemas de procesamiento de datos.

Conforme al proceso evolutivo de la disciplina del que hemos hablado, en la actualidad existen varias metodologías de programación, que se diferencian por la forma de conceptualizar los objetos y las relaciones entre ellos, y de la forma metodológica que se usa para diseñar e implementar las soluciones a los problemas.

Para los efectos del presente trabajo, tratamos con la metodología de la programación estructurada. La principal característica de este método, es que su modelo de programación trabaja diferenciando los datos de los procesos, con lo cual, la metodología trabaja sobre el diseño de los procesos que operaran sobre los datos para obtener resultados.

Así entonces, la metodología de la PC actúa como una estructura que organiza e integra los conocimientos de la disciplina, lo que da rigor científico a la propia disciplina y al proceso de aplicación de los conocimientos, en las tareas de resolución de los problemas de programación, haciendo que este proceso sea integral, sistemático y organizado.

Por tanto, la metodología es el referente teórico-formal y procedimental que contiene la base teórica y conceptual, el método y los procesos que se han de aplicar para resolver los problemas de procesamiento de datos y para programar las soluciones. En estos términos, se ha de entender que la propia tarea de buscar, diseñar, e implementar como programa una determinada solución, requiere de una integración efectiva de todos los elementos de la metodología.

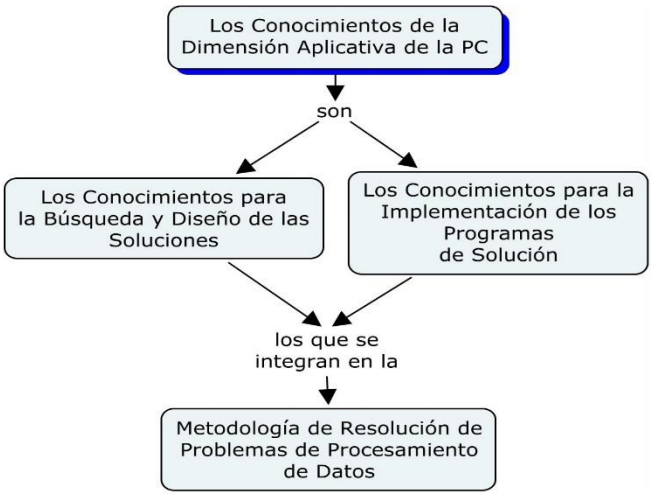

Figura 4: Los conocimientos que intervienen en la aplicación de la PC (Caneo, 2009)

Nos parece de suma importancia destacar este aspecto, especialmente en lo que concierne a los objetivos del presente trabajo, ya que es común creer que la actividad intelectual de la PC, se relaciona exclusivamente con el conocimiento y uso de un lenguaje de programación, desconociendo con ello, la importancia fundamental que tiene el conocimiento y el uso de la propia metodología de programación, y por tanto desconociendo también la importancia que tiene el uso integrado de los conocimientos que componen las bases teórico conceptual, y procedimental que son parte de ella.

En esta perspectiva, creemos que es de importancia pedagógica considerar que:

- Cuando se limita la PC sólo a los aspectos relativos al uso de los lenguajes de programación, se está desconociendo la relevancia disciplinar que tiene la metodología de la programación, ya que ella es un fundamento que, da rigor científico a la actividad intelectual de resolución de estos problemas.
- En la conducción del proceso de enseñanza, es necesario tener presente en todo momento, que estas dos áreas de conocimientos, la metodología y el lenguaje de programación, deben ser abordadas y aprendidas de manera integral y coherente.
- La integración y coherencia en el tratamiento de los contenidos, es un aspecto que ha sido destacado y argumentado de manera especial en las recomendaciones curriculares contenidas en el informe CC2001 (2001), en lo referido a la definición de los principios que se deben considerar, al momento de establecer y tratar los contenidos del cuerpo de conocimientos de Fundamentos de Programación (FP) en los planes de formación profesional en CcC.
- En él se formulan y defienden poderosos argumentos para establecer que el cuerpo de conocimientos de FP tiene un carácter nuclear e ineludible, para cualquier plan de formación profesional en CcC, toda vez que se reconoce,

que los estudiantes han de disponer de buenos niveles de manejo de estos conocimientos, y de las bases metodológicas de la PC, como una forma de garantizar el domino en la disciplina desde una perspectiva científica y técnica.

- En el mismo informe se precisa también, que las habilidades y capacidades para programar computadores son competencias que cualquier titulado de CcC debe poseer, entendiendo que esto va más allá de las habilidades para utilizar lenguajes de programación. Estas habilidades y capacidades deben considerar un adecuado nivel de manejo de la metodología de la programación.

Como lo hemos señalado, la metodología de programación, se organiza en torno a la ejecución de cuatro etapas para la resolución de los problemas de procesamiento de datos y la programación. Estas son, la etapa de análisis del problema; la etapa de diseño de la solución; la etapa de implementación del programa de solución; y la etapa de prueba y depuración del programa de solución.

Cada una de ellas, tiene una base teórico-formal que la justifica dentro del propio proceso de solución de los problemas, y posee también, una base procedimental que define el método y la técnica para ejecutarlas.

Conforme a los principios en que se sustenta la metodología de programación, con la ejecución de cada etapa se avanza desde un cierto estado del proceso de resolución, hacia otro estado más cercano a la propia solución que se espera construir, todo ello, conforme a un proceso que va generando ciertos productos intermedios, y que en la medida en que avanza en el proceso de resolución, van transitando desde productos que poseen un alto grado de abstracción hacia productos más concretos, y que culmina con el programa computacional de solución para el problema. En términos más prácticos, estos productos son modelos que representan los estados (inicial, intermedios, o final), del proceso de resolución del problema.

Para nosotros en la dimensión aplicativa de los conocimientos que estamos analizando, y en particular el trabajo con las etapas de la metodología de la programación debe considerar que:

- Para la etapa de análisis del problema, los formalismos y procedimientos, contienen los conocimientos y el método que se ha de aplicar para llevar a cabo la tarea de establecer la definición del problema a resolver. Más específicamente, son los conocimientos en los que se sustenta el método a aplicar para, analizar e interpretar la información contenida en el enunciado de un problema, y a partir de ello, identificar, seleccionar, definir y establecer cuáles son los datos y procesos que permiten caracterizar el problema.

Mediante la definición de estos elementos, se precisan cuáles son los datos o información que se requieren para resolver el problema, cuales son los resultados que ha de generar la solución y cuáles son los procesos que transformarán los datos de entrada en los resultados. Así, la aplicación de los fundamentos teórico-formales y procedimentales de esta etapa, permiten establecer la definición de la especificación del problema, la cual constituye el modelo de partida para la resolución del problema.

- Para la segunda etapa de la metodología, la base teórico-formal y procedimental que la sustenta, contiene los conocimientos y el método que se han de aplicar para diseñar y representar la solución al problema. En términos más concretos, son los conceptos y conocimientos que establecen el método que permiten convertir el primer modelo, en el modelo de solución del problema, y por tanto tratan con la identificación de los procesos que están implicados en la solución y con la organización lógica de la secuencia de acciones y operaciones para resolverlo.

  En el desarrollo de esta etapa intervienen conocimientos matemáticos, referidos a los procesos de cálculo, así como también conocimientos de la lógica de primer orden, los cuales son de utilidad para la definición de las expresiones lógicas, que controlarán las secuencias de acciones y operaciones contenidas en la solución.

  Esta etapa culmina con la representación algorítmica de la solución, ámbito en el cual interviene los conocimientos referidos a las técnicas de representación algorítmica, y al uso del lenguaje algorítmico. Estos son conocimientos en los que se sustenta el modelo de solución del problema, y su expresión como un algoritmo de solución.

  Como hemos expuesto y argumentado en apartados anteriores, la estructura de conocimientos que define la base formal y procedimental de las dos primeras etapas de la metodología de la programación, tienen un carácter esencial para la propia tarea de resolución de los problemas, ya que son las etapas que tratan con la elaboración de dos de los modelos más importantes para alcanzar la solución, que son el modelo del espacio del problema y el modelo de solución del problema.

  Con el primero de estos modelos, se obtiene la especificación del problema, es decir la definición de los elementos que caracterizan el problema, tales como el área del problema, los datos o información con que se cuenta para resolverlo, y los resultados que debe producir la solución. En tanto, en el modelo de solución se encuentra la organización lógica de las acciones y operaciones que han de ejecutarse para transformar los datos de entrada en los resultados. Este modelo de solución expresado como un algoritmo, constituye el modelo de entrada para la siguiente etapa de la metodología.

- Para la tercera etapa, los conocimientos que componen la base teórico-formal y procedimental, están fuertemente arraigados en la tecnología computacional, ya que sus conceptualizaciones se basan en el computador como máquina de cálculo y que puede ser programada mediante el uso un lenguaje computacional. Además consideran el uso de un software para la creación y el manejo del entorno de programación.
- Por último, para la cuarta etapa, los elementos con conforman la base teórico-formal y procedimental en que se sustenta su desarrollo y aplicación tienen el propósito de probar y depurar el programa de solución. Los conocimientos que la componen, se relacionan con los conceptos de prueba y depuración de programas, y con los conocimientos referidos a las técnicas que se aplican para probar y depurar un programa.

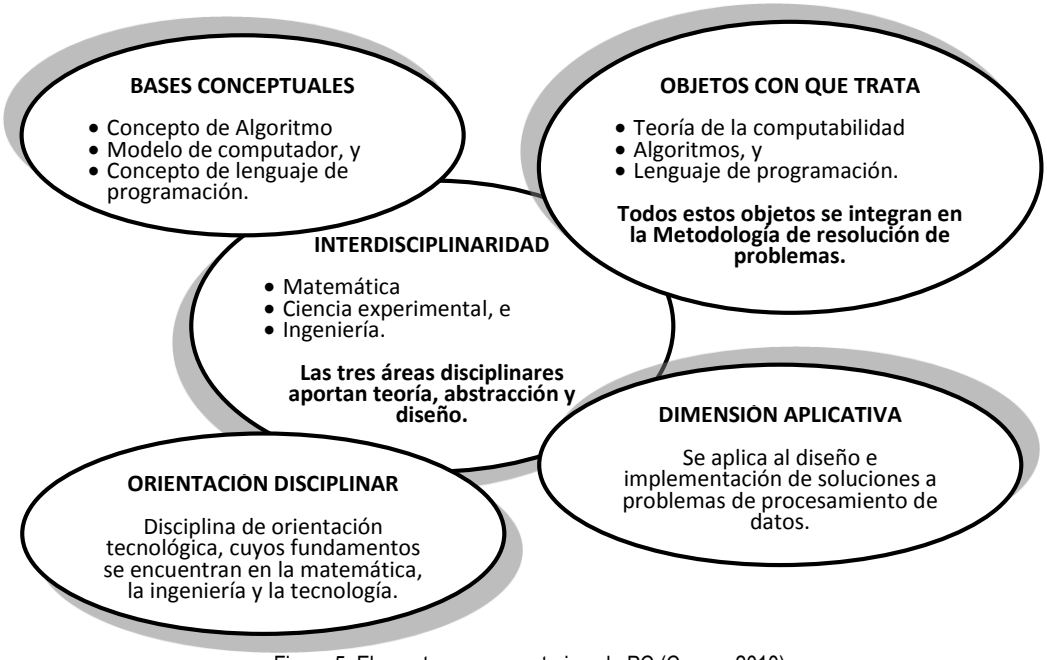

Figura 5: Elementos que caracterizan la PC (Caneo, 2010)

En síntesis como hemos expuesto en estos apartados, la estructura de conocimientos de la disciplina de PC, se construye a partir de diversas fuentes de conocimientos, los que para ser utilizados y aplicados a la resolución de los problemas de programación, deben integrarse de manera coherente y efectiva.

En su estado actual, la PC es reconocida como una disciplina cuyo objeto de estudio y aplicación, es el diseño de algoritmos para resolver problemas de procesamiento de datos, y su implementación como programa computacional. Como disciplina, constituye un cuerpo de conocimientos de orientación tecnológica, que se fundamenta en la matemática, la ingeniería y la tecnología computacional.

Cuando este cuerpo de conocimientos se aplica a la solución de los problemas, confluyen la teoría, el pensamiento abstracto como herramienta para la construcción y el trabajo con ciertos modelos, y las técnicas de diseño de las soluciones.

Como se ha expuesto, la PC es entonces un cuerpo de conocimientos interdisciplinares.

Conforme a esto, la PC trata con objetos tales como:

- Teoría de la computabilidad, tomada de la matemática aporta el concepto de función calculable,
- Concepto de algoritmo, tomada también de la matemática, es la base formal para tratar con el modelado y la definición de procedimientos ejecutables para resolver los problemas,
- Lenguaje de programación, tomado de la tecnología computacional, es el recurso para expresar los algoritmos como un programa computacional, y
- Metodología de resolución de problemas de procesamiento de datos, que es de la propia PC, permite tratar con la búsqueda, el diseño y la implementación de las soluciones a los problemas.

En estos términos, el principal propósito de la PC es aplicar su cuerpo de conocimientos a la búsqueda diseño e implementación de soluciones computacionales a problemas. En esta perspectiva aplicativa, el cuerpo disciplinar se organiza en dos áreas de conocimientos, por una parte los conocimientos que sirven a las tareas de búsqueda y diseño de las soluciones, y por otra, los conocimientos que sirven para implementar los programas computacionales, para que un cierto procedimiento algorítmico pueda ser ejecutado por la máquina computador.

1.2. La PC y su estructura lógico-cognitiva

En este apartado, desarrollamos el análisis de los conocimientos para la PC, desde la perspectiva de la estructura lógico-cognitiva que entra en juego al momento de llevar a cabo el conjunto de actividades intelectuales, que son demandadas para diseñar las soluciones a los problemas de procesamiento de datos, y para implementar el programa computacional correspondiente.

Llamamos estructura lógico-cognitiva de los conocimientos para la PC, a la resultante de la integración y organización lógica de los conocimientos de la disciplina y los conocimientos del espacio del problema, esto es, al conjunto de conocimientos que entran en juego, cuando se aborda la tarea intelectual de resolver un problema particular de procesamiento de datos.

Para estos efectos, analizaremos los tipos de conocimientos que se deben poseer, y lo más importante, la forma en que estos conocimientos se deben activar, usar, relacionar, organizar e integrar, con el objeto de cumplir de manera efectiva con la

tarea de diseñar e implementar soluciones a problemas de procesamiento de datos y el respectivo programa computacional.

En este análisis, iremos también acercándonos al problema general de la enseñanza de los conocimientos y habilidades que debe aprender un estudiante de ciencias de la computación, para constituirse en un resolutor efectivo de este tipo de problemas, cuestión que desarrollaremos con detalles en el siguiente apartado.

1.2.1. Los conocimientos implicados en la PC

Si bien en la actualidad, la PC trata con el diseño e implementación de soluciones para problemas muy diversos, en el primer curso de programación de una carrera de Ingeniería en informática, generalmente se trabaja con problemas referidos a procesos de cálculos matemáticos, de baja complejidad. Son problemas que tratan con valores numéricos, y con procedimientos matemáticos y lógicos, que operan sobre los valores numéricos para obtener ciertos resultados, que en su mayoría, son también numéricos.

Para ilustrar esta caracterización de los tipos de problemas que se tratan en la primera asignatura, consideremos el siguiente enunciado de un problema típico de este curso:

> *Implemente un programa computacional que permita determinar si un conjunto de tres segmentos pueden o no formar un triángulo. Si lo forma, el programa debe determinar si los segmentos forman un triángulo equilátero, isósceles o escaleno.*

Precisemos en primer término, que aunque este enunciado hace referencia a un problema matemático, la solución no demanda la obtención de resultados de tipo numérico, ya que los resultados que se esperan son conclusiones en el sentido de establecer si los segmentos pueden formar un triángulo o no, y para el caso en que puedan formarlo, el resultado que se espera es la determinación del tipo de triángulo que formarían los tres segmentos, en definitiva, hablamos de resultados que son conclusiones literales y no resultados numéricos.

Este es un problema matemático, ya que para que el conjunto de segmentos pueda formar un triángulo deben cumplirse ciertas reglas matemáticas, o más específicamente, ciertas condiciones de la geometría las cuales han de ser evaluadas mediante cálculos de expresiones algebraicas.

Consecuente con esto, la evaluación de las reglas para determinar si forman o no un triángulo, demanda el uso de expresiones lógicas, y de manera complementaria, la evaluación para determinar el tipo de triangulo, también implica el uso de este tipo de expresiones.

Así, para diseñar el procedimiento algorítmico de solución, se requiere del conocimiento sobre los procedimientos matemáticos para determinar si un conjunto de tres segmentos forma un triángulo, y para determinar su tipo, además de la construcción de operaciones lógicas, que controlen estas secuencias de acciones, con el objeto de que el algoritmo responda de manera correcta, a todos los casos posibles de conjuntos de tres segmentos que se le den al algoritmo, con lo cual se cumple con uno de los requisitos más importantes del diseño de soluciones, que es su generalidad, que significa que la solución ha de ser efectiva para cualquier conjunto de tres segmentos.

La condición de problema matemático se verifica también por el hecho de que los datos que se requieren para evaluar cualquier caso particular de tres segmentos, corresponden a datos numéricos, los que además, deben cumplir con la condición matemática de ser valores positivos.

El análisis anterior no permite identificar, los primeros conocimientos que han de formar parte de la estructura lógico-cognitiva que se requiere para, diseñar la solución y para implementarlas como un programa computacional, hablamos de los conocimientos referidos a las formas declarativas y procedimentales de las cuatro operaciones básicas, como son la suma, la resta, la multiplicación y la división.

En esta dimensión, importa también el conocimiento respecto del significado matemático de cada una de las operaciones y la forma de combinarlas para construir y operar con expresiones matemáticas en las que intervienen varias de estas operaciones.

Así también, la evaluación de casos particulares de conjuntos de tres segmentos, requiere la construcción de expresiones lógicas para controlar secuencias acciones para cada caso particular de tres segmentos.

Estos dos ámbitos de conocimientos deben integrarse con los conocimientos para diseñar y representar la secuencia algorítmica, que permita resolver el problema. Este último ámbito de conocimientos, tiene dos aspectos que son de importancia para el diseño de las soluciones a los problemas:

A.-Es la base conceptual para caracterizar las condiciones que ha de cumplir una secuencia de acciones y operaciones, para que sea considerada una solución algorítmicamente válida. Que una solución sea algorítmicamente válida significa que, cada acción contenida en el algoritmo cuenta con una definición precisa y no ambigua. Significa también que, la ejecución del algoritmo es predecible, esto es, que su ejecución nunca generara resultados inesperados, y significa también, que el algoritmo para un mismo conjunto de valores de entrada, generará los mismos resultados de salida.

El conocimiento y manejo de estos significados, es fundamental para la utilización de los algoritmos como herramienta de modelamiento del diseño de las soluciones, y para la representación adecuada de ellos, como paso previo a su implementación como programa computacional.

A este respecto nos parece importante precisar que:

1. Aunque un algoritmo exprese una solución que es válida, lo que significa como ya dijimos, que las acciones contenidas en él cuentan con una definición precisa y no ambigua, ello no es garantía de que el mismo resuelva el problema para el cual ha sido diseñado, puesto que el hecho de que la solución sea correcta, dependerá de si la organización lógica de las acciones y las operaciones, es adecuada para resolver el problema con que se trata.
2. Esto significa que, dada las características particulares de cada problema de procesamiento de datos, no es posible suponer, que el conocimiento respecto de los algoritmos, pueda ser construido a partir de un cierto conjunto de algoritmos genéricos, que sirvan para resolver cualquier problema de procesamiento de datos, con lo cual queremos clarificar y enfatizar, que una de las características importantes de la aplicación de estos conocimientos, es que cada problema de procesamiento de datos, demanda un diseño algorítmico particular.

B.-El lenguaje algorítmico y las formas de representación algorítmica, constituyen el recurso para aplicar y usar el concepto de algoritmo dentro del proceso de resolución de los problemas. El lenguaje y las formas de representación algorítmicas, son el tipo de conocimiento que provee de un conjunto de símbolos y significados, no solo para representar los algoritmos de una forma adecuada, sino que también para hablar de los algoritmos y de las soluciones que se expresan mediante ellos, constituyéndose así en elementos que permiten que los algoritmos, puedan ser evaluados en términos de su efectividad para resolver el problema, y de paso puedan ser también comunicados y comprendidos.

Disponer de formas adecuadas de representación algorítmica, permite al resolutor, realizar ejecuciones simbólicas de ella, esto es, realizar seguimientos a las secuencias de acciones y operaciones, con el objeto de constatar la correctitud de la solución, o en su defecto, detectar los errores en su diseño. Y por último, el seguimiento de un algoritmo permite su perfeccionamiento, con el objeto de encontrar la solución más óptima para el problema, ya que el seguimiento permite, ir detectando y eliminando pasos y/o operaciones innecesarias y/o repetitivas dentro de él.

Como se observa del análisis anterior, tanto el concepto de algoritmo, como el lenguaje y las formas de representación de los algoritmos, conforman un conjunto de

conocimientos que necesariamente deben estar en la estructura lógico-cognitiva de quien trata con la resolución de este tipo de problemas, ya que son los elementos formales y operacionales que le sirven para diseñar las soluciones, para comunicarlas, probarlas y optimizarlas, o para descartarlas como válidas.

Otro componente que ha de formar parte de la estructura lógico-cognitiva para la programación de computadores, es el concepto de computador como máquina de cálculo que puede ser programada. Este conocimiento es esencial para conceptualizar el computador como la máquina que actuará como recurso tecnológico, para automatizar la solución que se diseñe para un problema.

En estos términos consideramos que:

- Este conocimiento le permite al resolutor, disponer de un modelo apropiado de computador, y reconocer y comprender sus características y principios de funcionamiento, además de los recursos que éste posee, para utilizarlo como herramienta tecnológica para la ejecución de los programas de solución que han sido diseñados. En este mismo ámbito, el modelo de computador provee del sustento formal, para comprender las capacidades de programación que esta máquina posee y la forma de concebir la organización lógica de ella, y que hacen posible que pueda ser programada.
- Es importante destacar que si bien el modelo de computador no constituye un conocimiento de la estructura lógico-conceptual, que se relaciona con los elementos cognitivos para el diseño de las soluciones, este modelo y sus elementos, son de importancia fundamental para llevar a cabo la implementación de las soluciones como un programa computacional.
  El modelo de computador permite la conceptualización de una máquina que:
  - ♦ Tiene capacidades de procesamiento,
  - ♦ Es capaz de interpretar y ejecutar acciones y cálculos, para lo cual hace uso de un lenguaje, y
  - ♦ Dispone de un conjunto de recursos para el procesamiento, tales como memoria, unidad de procesamiento, y diversos dispositivos de almacenamiento de entrada y salida.

Estrechamente vinculado con el modelo de computador, están los conocimientos referidos al lenguaje de programación. Estos le proveen al resolutor, los elementos conceptuales y de orden práctico, que al ser integrados, le permiten implementar los programas computacionales. Son los conocimientos referidos al concepto de lenguaje de programación, al de paradigma o modelo de programación, y los referidos a los aspectos declarativos que posee el lenguaje de programación.

Los aspectos declarativos del lenguaje, definen los elementos sintácticos y semánticos de las variables, las constantes, los tipos de datos, las sentencias, los operadores y las estructuras de programación que soporta el lenguaje, por lo que son los conocimientos que el resolutor habrá de poner en juego, al momento de llevar a cabo la implementación de los programas de solución.

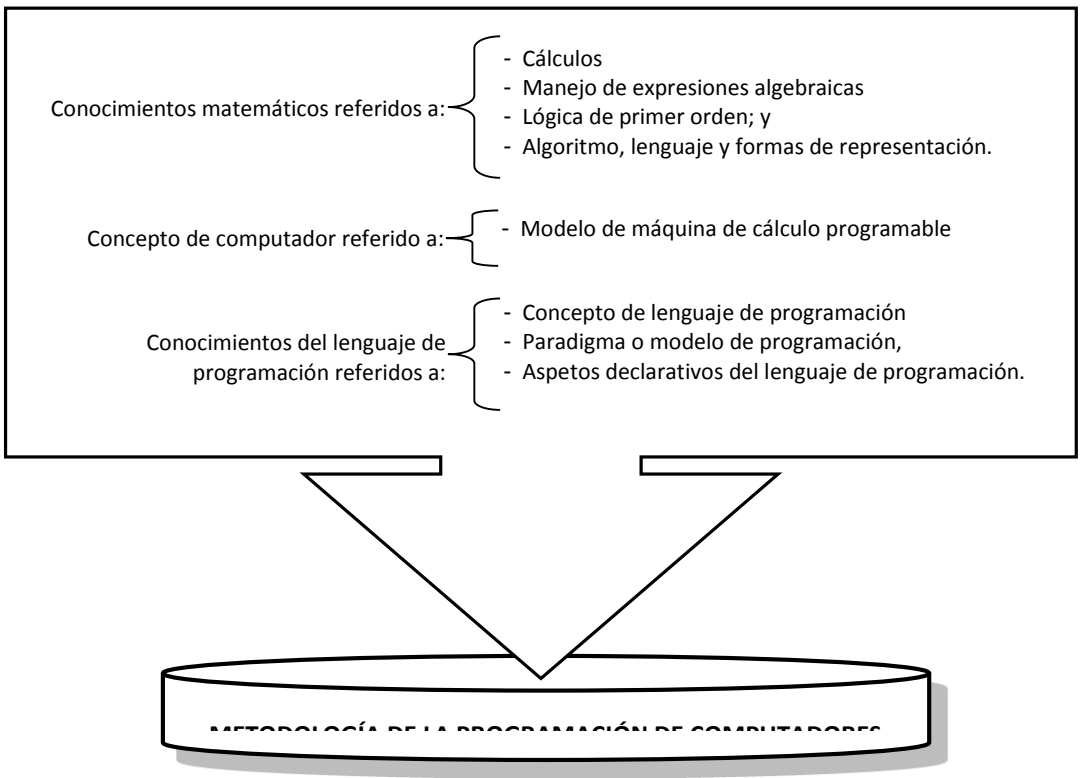

Figura 6: Los conocimientos para la PC y su integración en la Metodología de la Programación (Caneo, 2010)

Si bien estos conocimientos no intervienen en el diseño de las soluciones a los problemas, al igual que en el caso del modelo de computador, son los conocimientos que el resolutor tiene que conocer, comprender y utilizar, para implementar los programas de solución, y bajo esta perspectiva, constituyen otro de los elementos que han de formar parte de la estructura lógico-cognitiva de la que estamos hablando.

Todos estos conocimientos de la estructura lógico-cognitiva que hemos individualizado en los apartados anteriores, se integran y sistematizan en lo que se denomina la metodología de programación de computadores.

Esta integración, da el carácter formal, disciplinar e ingenieril, a la propia actividad intelectual de la resolución de problemas de programación de computadores, por tanto, la metodología es de importancia fundamental para abordar con apego disciplinar, la propia tarea de resolver estos problemas.

En ella se encuentran los pasos metodológicos para abordar de forma sistémica e integrada el proceso de resolución de los problemas.

1.2.1.1. La integración de los conocimientos

Desde la perspectiva de la organización del proceso de resolución de los problemas, la metodología de la programación, actúa como una unidad de conocimientos que integra y da coherencia, a todos los demás elementos que conforman la estructura lógico-cognitiva de la PC.

En estos términos, la metodología de la programación es el marco de referencia conceptual y procedimental para aplicar correctamente todos los formalismos, los conceptos y los procedimientos que hemos mencionado.

Desde la perspectiva práctica, en la metodología se encuentra la definición del porqué, el para qué y el cómo el resolutor deberá ir utilizando y aplicando los conocimientos de la estructura lógico-cognitiva, a la hora de ejecutar de los distintos pasos metodológicos.

La integración de estos conocimientos, se produce al ejecutar las cuatro etapas de la metodología para la resolución de los problemas de programación Estas etapas son:

1.- Análisis del problema,
2.- Diseño de la solución,
3.- Implementación del programa, y
4.- Prueba y depuración del programa de solución.

Dado el carácter aplicativo de esta metodología, en su estructura de conocimientos se definen los propósitos de cada etapa, y los procedimientos y técnicas que han de usarse para desarrollarlas, lo que como conjunto, define el proceso completo de resolución de los problemas de procesamiento de datos.

Po tanto, estos son los conocimientos de orden metodológico y procedimental, que deberá tener integrado el resolutor en su estructura lógico-cognitiva, esto como condición para llevar a cabo las tareas de resolución de estos problemas, desde una perspectiva disciplinar e ingenieril.

Por tanto, para utilizar y aplicar la metodología, el resolutor deberá conocer, comprender e integrar en su estructura lógico-cognitiva, los fundamentos de cada etapa, los procedimientos y las técnicas para usarla, pero además, dependiendo de cada etapa, deberá ir integrando los conocimientos que hemos señalado en los párrafos anteriores.

Así, por ejemplo, al utilizar y aplicar los fundamentos de la etapa de implementación del programa, deberá integrar los conocimientos del modelo de computador, el concepto de lenguaje de programación, el de paradigma de programación, y los

conocimientos referidos a los aspectos sintácticos y semánticos del lenguaje de programación.

1.2.1.2. El rol de las estrategias de resolución de problemas

Sin embargo, aunque el resolutor domine la metodología de la programación, y haya conseguido integrar en ella todos los conocimientos que hemos descrito, en la práctica, esto no es suficiente para que él pueda llevar a cabo de manera exitosa y completa las tareas de resolución de los problemas.

Conforme a nuestra experiencia, para lograr este propósito el resolutor debe integrar además en su estructura lógico-cognitiva, estrategias de resolución de problemas. Las estrategias de resolución de problemas, constituyen un tipo especial de conocimiento que es esencial para acometer este tipo de tareas intelectuales, no sólo en el ámbito de la programación de computadores, sino que también para resolver otras clases de problemas, como por ejemplo, los problemas matemáticos y los problemas físicos.

Varios autores entre los que podemos citar a Ali (2005), Bishop (1995), Brooks (1990). Brusilovsky, Kouchnirenko, and Tomek, (1994), Burkhardt, Détienne, & Wiedenbeck (1997), Cañas, Bajo, & Gonzalvo, (1994), Corritore, & Wiedenbeck, (1991), Davies (1993), Détienne (1990), du Boulay (1989), Evans (1996), Gilmore (1990), Hoc (1989), Kessler & Anderson (1989), Linn & Dalbey (1989), Mayer (1989), Milne and Rowe (2002), Pennington & Grabowski (1990), Robins, Rountree & Rountree (2003), van Merrienboer (1990), y Ye & Salvendy (1996), parecen coincidir que en general para resolver problemas, se requieren tres clases de conocimientos:

1.- Declarativos, los que corresponden al conocimiento de los formalismos, los principios y/o reglas que están relacionadas con el problema que se quiere resolver.
   Así por ejemplo, para el caso del problema de determinar si un conjunto de tres lados forman un triángulo, el concepto de triangulo y las condiciones que deben cumplir un conjunto de tres lados para formar un triángulo, son parte del conocimiento declarativo
2.- Procedimentales, que es el referido al conocimiento de procedimientos para hacer algo. Situándonos en el mismo ejemplo de la determinación de si un conjunto de tres lados forma un triángulo, correspondería al conocimiento respecto de cómo realizar los cálculos para determinar si el conjunto de lados forman un triángulo, así como también, por el procedimiento para obtener resultados a partir de ciertas expresiones de la lógica de primer orden, y
3.- Estratégicos o condicionales (que referiremos de aquí en adelante con estratégicos), que corresponde al conocimiento que le permite al resolutor llevar a cabo varias tareas intelectuales que son esenciales para resolver esta

clase de problemas. Para nosotros, este conocimiento estratégico es el más importante para conformar el conocimiento experto del resolutor.

En estos términos, mediante el uso del conocimiento estratégico (en gran medida del conocimiento de experto que se tiene), se podrán llevar a cabo las tareas intelectuales de identificar y seleccionar, los conocimientos declarativos y procedimentales que se encuentran implicados en la solución del problema, para luego dar paso a la organización de los mismos, y de esa forma diseñar una solución para el problema.

Ahora bien, esto que explicamos como una secuencia lineal de tareas intelectuales basadas en el conocimiento experto, en la práctica no es así, ya que si bien esto puede iniciarse como una secuencia lineal, la propia ejecución, generalmente implica iteraciones y reiteraciones de una o más de estas tareas.

Esto es consecuencia también del conocimiento estratégico (experto), y es el resultado de la utilización de ciertas estrategias, mediante las cuales el resolutor va evaluando la pertinencia y efectividad, que tienen los resultados de la ejecución de cada tarea intelectual, en términos de conducirle a la solución del problema.

Hablamos de una suerte de conocimiento meta-estratégico ya que permite evaluar la efectividad en la aplicación del propio conocimiento estratégico. Este es el sentido más profundo de considerar este conocimiento estratégico, como conocimiento de experto, ya que no sólo es el conocimiento para guiar las tareas intelectuales esenciales del proceso de resolución de estos problemas, sino que, es el conocimiento que permite, evaluar y hasta corregir, la forma en que se está usando este conocimiento experto, cuando se aplica para diseñar la solución a un problema de procesamiento de datos particular.

Lo que se sabe hasta ahora, respecto del conocimiento experto en la resolución de problemas desde la psicología cognitiva, es que las capacidades para realizar procesos de abstracción, constituye una las estrategias de resolución de problemas más importantes, lo que coincide con lo planteado por Mullera and Haberman (2008), quienes señalan que la abstracción es uno de los conceptos más fundamentales para la práctica de la resolución de problemas en CcC.

1.2.1.3. El rol de la capacidad de abstracción

Guralnik (1984,) precisa que el proceso de abstracción involucra la elaboración de una idea, que se construye a partir del análisis de las cualidades o propiedades de un objeto o situación, y opera sobre la base de llevar a cabo una separación de los elementos que componen el objeto o situación.

Para Mullera and Haberman (2008), las principales características de la abstracción son:

- La generalización de ejemplos específicos;
- La identificación, extracción y el aislamiento de los componentes esenciales; y,
- La acción de ignorar u ocultar los detalles irrelevantes.

Este mismo autor clarifica que mediante la abstracción se construye una entidad, la que es un modelo o una representación de un objeto o situación, siendo este modelo el resultado del proceso de abstracción.

Por tanto, el proceso de abstracción da origen una representación mental, que construye el resolutor y que le permite, capturar solo los aspectos y elementos esenciales del problema, reduciendo con ello la complejidad aparente que podría tener el problema para alguien que no usa, o que usa de manera incorrecta, el proceso de abstracción.

Volviendo sobre el problema ejemplo de determinar si un conjunto de tres magnitudes forma un triángulo, la construcción de la representación mental del triángulo que resulta del proceso de abstracción, debería estar basada en un triángulo caracterizado por la magnitud de sus segmentos, y no por sus ángulos interiores, ya que la información con que cuenta, es la referida a la primera característica.

Esta misma representación mental debería rescatar las propiedades que deben cumplir el conjunto de tres segmentos para poder formar un triángulo, así como las propiedades que deben cumplir para formar un triángulo equilátero, isósceles, o escaleno.

Así, el proceso de abstracción aplicado en la etapa de análisis del problema, debe permitir identificar y organizar la información que es esencialmente relevante para establecer de manera precisa, el espacio del problema y para construir la especificación del problema.

Si se aplica a la etapa de diseño de la solución, debe permitir identificar y organizar los procesos de cálculos, o de otro tipo, que son relevantes para el diseño de la solución. Aplicada a la etapa de implementación del programa computacional, debe permitir identificar y organizar las estructuras de programación que son relevantes para la construcción del programa.

Y finalmente, aplicada a la etapa de prueba y depuración del programa, ha de permitir identificar y establecer, los casos o situaciones de prueba que permitan verificar la

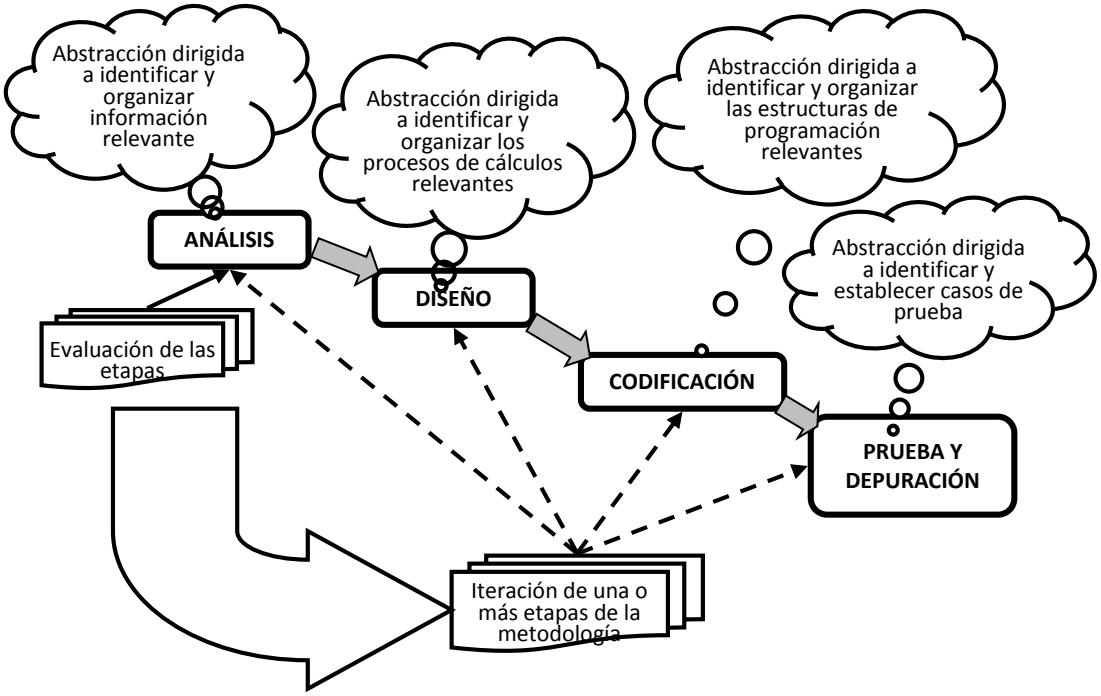

Figura 7: Secuencialidad de las etapas de la metodología, y los tipos de abstracción que intervienen. (Caneo, 2010)

correctitud del programa de solución.Como vemos, el conocimiento estratégico, y dentro de él, la abstracción como capacidad de pensamiento, para construir y trabajar con modelos de representación, es una herramienta poderosa y de indiscutible utilidad para la resolución de estos problemas.

La experiencia que hemos recogido durante varios años enseñando FP, nos muestra que, aunque los estudiantes dispongan de los suficientes conocimientos declarativos y procedimentales, pero sin embargo no consigan desarrollar el conocimiento estratégico, que les permita usar adecuadamente las dos categorías de conocimientos anteriores, definitivamente no lograran alcanzar la capacidad de resolver los problemas de procesamiento de datos, que es el objetivo fundamental de un primer curso de programación y también del currículum en CcC.

Esta exigencia se encuentra expresamente señalada en el informe de las recomendaciones curriculares elaborado y presentado por la ACM y el IEEE, donde se precisa que:

> *"All computer science students must learn to integrate theory and practice, to recognize the importance of abstraction, and to appreciate the value of good engineering design" (ACM, 2001, p. 12).*

Hecho el análisis de los elementos que conforman la estructura lógico-cognitiva para la programación, y la relevancia y el rol que tienen cada uno de estos conocimientos y las estrategias que hemos señalado, para llevar a cabo la tarea intelectual de resolver los problemas de procesamiento de datos, en el contexto de un primer curso de FP, en los siguientes apartados nos abocaremos a ilustrar mediante un ejemplo,

como los elementos de esta estructura, intervienen en la resolución de los problemas típicos que se abordan en un primer curso de programación.

1.2.2. Caracterización de los problemas de procesamiento de datos

Como hemos señalado, los problemas que se abordan en un primer curso de FP, se caracterizan por ser problemas matemáticos, que plantean situaciones que demandan generalmente soluciones cuantitativas de carácter simple. Esto significa, que son problemas que se caracterizan por plantear situaciones que tratan con valores numéricos y sus relaciones, basadas en operaciones matemáticas, y cuyas soluciones generalmente se traducen en resultados de tipo cuantitativos.

Otra características importante de este tipo de problemas, es que demandan soluciones de carácter general, en el sentido de que el diseño de la propia solución, debe cumplir con el requisito de ser válida para una amplia variedad de problemas del mismo tipo, o clase del problema.

Para ilustrar estos aspectos, consideremos el enunciado del siguiente problema particular:

> *Suponga que va de compras a una tienda que se encuentra ofreciendo rebajas de hasta un 40%. Calcule el valor a pagar por un artículo, el cual tiene marcado un precio de $3.000 sin rebaja, y conforme a la oferta tiene una rebaja de un 15%.*

El problema anterior constituye un caso particular, de una cierta clase de problemas de tipo cuantitativo que podríamos llamar, “problemas de determinación del valor a pagar por un artículo en liquidación”. Clasificamos a este problema como un caso cuantitativo particular, ya que se está demandando el cálculo del valor a pagar para el caso particular de un precio y un porcentaje de descuento.

La generalización de este problema, para formularlo como un típico problema de procesamiento de datos puede enunciarse como:

> *Suponga que va de compras a una tienda que se encuentra ofreciendo rebajas de hasta un 40%. Diseñe un programa que permita calcular y mostrar el valor a pagar por un cierto artículo en liquidación, teniendo en cuenta que todos los artículos no tienen el mismo porcentaje de rebaja.*

En este último enunciado aunque se trata del mismo tipo de problema cuantitativo, expresa un mayor grado de generalización para la situación planteada, ya que no se indica el valor del artículo, ni el porcentaje de rebaja que se ha de aplicar, precisando sólo que dicho porcentaje no puede ser superior al 40%.

Analicemos un segundo ejemplo de carácter más complejo y relacionado esta vez, con el cálculo del valor a pagar por un cliente por el consumo mensual de energía

eléctrica. El enunciado del problema no generalizado podría tener la siguiente formulación:

> *Calcule el valor que deberá pagar un cliente por su consumo mensual de energía eléctrica, considerando que ha consumido 54 KWH, y que conforme a los tramos de tarifas por consumo, se cobran $30 por cada KWH cuando el consumo es hasta 50 KWH, $42 por cada KWH de exceso cuando se han consumido más de 50KWH y hasta 80 KWH, y $60 por cada KWH de exceso cuando se han consumido más de 80 KWH. Considere además que el valor a pagar por el cliente esta afecto al impuesto IVA.*

Si se analiza este enunciado, nuevamente es posible apreciar que se está tratando con un caso particular de un problema cuantitativo.

Una generalización de este problema puede enunciarse como:

> *Construya un programa computacional que permita determinar el valor a pagar por un cliente por su consumo mensual de energía eléctrica, considerando que el plan de tarifas es el siguiente:*
> - *Hasta 50 KWH de consumo, se deben pagar $30 por cada KWH*
> - *Más de 50 KWH y hasta 80 KWH, se deben pagar $42 por cada KWH de exceso.*
> - *Por consumos superiores a 80 KWH, se deben pagar $60 por cada KWH de exceso*
>
> *Considere además que el valor a pagar por el cliente esta afecto al impuesto IVA.*

En esta generalización, se está demando una solución cuantitativa que deberá ser efectiva, para el cálculo del valor a pagar por un cliente cualquiera, independientemente del consumo mensual.

El grado de generalización, es una característica que se refleja en los enunciados de todos los problemas de procesamiento de datos, lo que por tanto, debe reflejarse en el diseño de las soluciones y en el programa de solución.

En estos términos, la programación de computadores y su actividad más concreta, como es la búsqueda e implementación de soluciones a problemas de procesamiento de datos, se orienta al diseño de soluciones que deben ser efectivas, para un conjunto de variantes de un mismo tipo de problema, como por ejemplo, todas la variantes referidas al problema de determinar el valor a pagar por un artículo que, teniendo un valor de venta de $100.000, o $3.500, o $18.000 o cualquier otro valor, esta afecto a un descuento de un 60%, o un 21%, o 35% o cualquier otro valor. O como el caso del problema de determinar el valor a pagar por la energía eléctrica, cuando se ha tenido un consumo de 5(KWH) o 57(KWH) o 12(KWH), o cualquier otro valor y por tanto estará afecto a una tarifa Q, o R, o S, o T, o cualquier otra.

Particularmente, consideramos que la generalización de la solución a los problemas de programación, tiene un efecto directo sobre la complejidad de la misma, ya que en el diseño de ella, deberán considerarse procesos de tipo lógicos para controlar las distintas condiciones de borde, con el objeto de asegurar que la solución es efectiva para todas las variantes del mismo problema.

En muchos casos, para responder a la generalización de las soluciones, se requiere también del diseño de procesos lógicos, que controlen las restricciones que poseen los problemas, las que dependiendo del enunciado, pueden estar referidas a los valores que pueden tomar los datos o información de entrada.

Así por ejemplo, para un cierto problema estas restricciones podrían estar referidas a que ciertos valores de entrada deban ser positivos y menores que 100, o para otro problema, la restricción podría estar referida a que los resultados de un cierto cálculo deben ser sólo valores enteros, etc.

Conforme a nuestra experiencia, consideramos que la complejidad producida por la necesidad de responder a la generalización de las soluciones, constituye otro de los aspectos difíciles de abordar por los estudiantes.

Para responder a la generalización de los problemas, no basta con que nuestros estudiantes aprendan respecto del diseño de una solución para un caso particular de un problema, sino que, deben desarrollar la habilidad de diseñar soluciones que sean efectivas para la generalización del mismo, es decir, soluciones que sean válidas para la clase de problemas a la cual pertenecen un cierto caso particular del mismo.

Esto implica además, que quién diseña las soluciones no sólo debe pensar en los casos generales y regulares para los cuales la solución debe ser válida, sino que, debe pensar en aquellos casos no regulares o excepcionales, y por supuesto, pensar y considerar, cómo responderá la solución a esos casos, como la excepción de que al ejecutar el programa, por error se ingrese como dato, un valor negativo para el precio del artículo, para el caso del primer ejemplo que hemos dado, o que para el caso del segundo ejemplo, se ingrese como dato un valor de consumo de energía eléctrica también negativo.

El caso es que, aun entendiendo que estas dos situaciones que ilustramos constituyen casos excepcionales, puesto que se supone que el valor de un producto no puede ser negativo, y que tampoco puede serlo el valor del consumo de energía eléctrica de un cliente, el diseño de las soluciones debe asegurar que el programa que resulta, produce resultados esperables bajo cualquier condición de estos (y otros) valores de entrada, en definitiva, para cualquier situación particular de cada clase de problemas.

Veamos otro caso. Supongamos por ejemplo, que se requiere diseñar una solución para el problema de encontrar las raíces que son solución de una ecuación de segundo grado de la forma $Ax^2 + Bx + C = 0$

En primer término, el diseño de una solución para este problema debe ser efectiva para cualquier valor de los coeficientes A, B, y C, incluyendo la posibilidad de que

los valores de B y/o C puedan ser cero, y también deberá funcionar de manera esperable para el caso excepcional de que el valor de A sea cero, en cuyo caso, la solución debe arrojar como resultado la indicación de que no se está tratando con una ecuación de segundo grado. Y por último, la solución también debe ser efectiva para aquellos casos en que la ecuación tenga soluciones imaginarias.

Por tanto, desde la perspectiva de la complejidad que esto significa para el estudiante, no es suficiente con que éste piense en el diseño de una solución particular para una ecuación de este tipo, como podría ser la ecuación $3x^2 + 2x - 5 = 0$, que corresponde a una ecuación que tiene soluciones reales, u otra como $x^2 - 4x + 20 = 0$, que corresponde a una ecuación que tiene soluciones no reales o imaginarias, sino que, debe pensar en el diseño de una solución que sea efectiva, para cualquier problema de esta clase, es decir, que sea efectiva para cualquier ecuación de segundo grado.

Esto significa que la solución ha de ser efectiva para cualquier grupo de valores de los coeficientes A, B, y C, incluyendo valores positivos, negativos y ceros de los mismos, y que por añadidura, para los casos en que el coeficiente A sea cero, el resultado de la solución indique que los coeficientes no se corresponden con una ecuación de segundo grado.

Otra característica de las soluciones de los problemas de programación, es que en ellas están implicados el conocimiento y uso de recursos tales como el lenguaje algorítmico, el lenguaje computacional y el computador como máquina de procesamiento.

Así, que una vez que se ha pensado en la solución generalizada para una cierta clase de problemas, dicha solución debe expresarse en un lenguaje algorítmico que sea apropiado tanto para hablar de la propia solución y su efectividad, como para que esta pueda ser implementada como un programa computacional.

Por tanto, luego de expresar la solución en el lenguaje algorítmico, esta ha de expresarse como instrucciones en un lenguaje de programación, con lo cual finalmente será posible verificar de manera más concreta, si la solución es correcta al ser ejecutada con el computador, para los casos de prueba que se requieran para esta constatación.

Esta ejecución del programa, demandará a su vez el manejo de otros recursos tales como un software para gestionar el entorno de programación, y ciertos elementos de hardware como, el teclado, el monitor, unidades de disco, el procesador y le memoria, entre otros recursos que son necesarios para crear, almacenar y ejecutar el programa computacional resultante.

Conforme a nuestra experiencia, lo más complejo en el aprendizaje del manejo de estos recursos por parte de los estudiantes, es el uso del lenguaje algorítmico. Creemos que la dificultad está en la complejidad que representa para ellos, elaborar la organización lógica de las acciones y operaciones que debe considerar la solución al problema, para que responda de manera correcta a la generalización de la que hemos hablado.

Mediante estos ejemplos, hemos ilustrado y analizado por una parte, la característica de generalidad que se exigen de las soluciones a estos problemas, y por otra, hemos evidenciado la variedad de conocimientos y recursos de pensamiento, que se deben poner en juego para el desarrollo de cada etapa del proceso de resolución de los problemas.

Estos son aspectos que a nuestro juicio, constituyen importantes fuentes de dificultades para nuestros estudiantes, y que por tanto tiene implicancias importantes sobre el aprendizaje.

1.2.3. El procesamiento cognitivo en la resolución de problemas de procesamiento de datos.

Como hemos señalado, un aspecto que nos parece crítico en la propia tarea de resolver los problemas de procesamiento de datos, tiene que ver con el desarrollo del proceso integración de los distintos conocimientos de la estructura lógico-cognitiva al momento de ejecutar la tarea de resolución de estos problemas.

En este apartado intentamos describir en detalle cómo debería producirse el procesamiento cognitivo durante la resolución de los problemas de programación, es decir, cómo se debería ir produciendo la integración de los conocimientos, y cómo debería dirigirse el pensamiento abstracto para construir y usar los distintos modelos, que son el resultado de la integración de los conocimientos.

En esta idea, a continuación desarrollamos esta descripción desde la perspectiva de la ejecución del propio proceso de resolución, clarificando los procesos y estrategias cognitivas que el resolutor debería poner en juego a la hora de resolver estos problemas. Para estos efectos, utilizamos un ejemplo típico de problema de procesamiento de datos que se aborda en un curso de FP.

Ahora bien, frente a la complejidad que implica detallar las actividades intelectuales que podrían producirse en la mente de un resolutor, cuando trabaja en la solución del problema, en la explicitación que presentamos a continuación, nos enfocaremos fundamentalmente, en ir señalando los conocimientos, las estrategias que entran en juego, y los procesos y los productos que se deberían ir obteniendo, al aplicar la metodología de resolución.

Supongamos el siguiente enunciado:

> *Construya un programa que permita encontrar las raíces, que son solución de una ecuación de segundo grado de la forma $Ax^2 + Bx + C = 0$. El programa debe constatar si la ecuación corresponde realmente a una ecuación de segundo grado, y además deberá indicar con un mensaje cuando la solución sea imaginaria.*

El proceso de resolución del problema debería iniciarse tomando como base, el conocimiento referido a la metodología de resolución de problemas de procesamiento de datos. Como hemos argumentado, estos conocimientos establecen el marco conceptual y procedimental del proceso completo de resolución, y además, es el cuerpo de conocimientos que actúa como integrador del resto de los elementos que conforman la estructura lógico-cognitiva de la PC.

Como se ha hecho mención, la metodología de resolución se encuentra organizada en cuatro etapas: Análisis del problema; Diseño del algoritmo de solución; Implementación del programa de solución; y Prueba y depuración del programa de solución.

Al desarrollar cada etapa, se ha de trabajar desde el enunciado del problema, hacia la implementación del programa de solución, conforme a un proceso que va desde la formulación abstracta del problema, hasta la implementación concreta de su programa de solución, el que debe cumplir con el requisito de poder ser ejecutado en un computador real.

Por tanto, el proceso de resolución, es un proceso de refinamiento continuo, que durante su desarrollo, va generando productos intermedios (los distintos modelos), y que en la medida en que se progresa en él, se va transitando desde una situación problema formulada en términos abstractos, hacia la formulación concreta de una solución al mismo, expresada como un programa computacional, que puede ser ejecutado en un computador real.

Por tanto, para llevar a cabo el proceso de resolución de estos problemas, se requiere que el estudiante, conozca, comprenda, y aplique los conocimientos de la estructura lógico-cognitiva, y por otra, se requiere que disponga de habilidades de pensamiento, específicamente hablamos de la capacidad para realizar variados procesos de abstracción y a partir de ellos, elaborar y procesar varios modelos, los que van desde el modelo de la situación inicial del problema, en el cual se captura e interpreta, el enunciado del problema, pasando por el modelo de solución del mismo, el que se materializa en el algoritmo de solución, hasta llegar al modelo del programa de solución, el cual captura el funcionamiento computacional de la solución.

En los conocimientos que conforman la estructura lógico-cognitiva, se encuentran los formalismos y los conceptos, que dan sustento a la metodología y a la ejecución de ella, además de los procedimientos, para desarrollar cada etapa de la metodología y

para elaborar los productos que cada una de ellas deben ir generando, cuando se avanza en el proceso de resolución.

En este contexto, las habilidades de pensamiento, son necesarias para realizar los procesos de abstracción que permitan transitar desde lo abstracto hacia lo concreto. Esto se logra, mediante la construcción de distintos modelos, los cuales identifican y caracterizan, los elementos que son esenciales para cada etapa del proceso de resolución, al tiempo que van permitiendo dilucidar los elementos para crear nuevos modelos, más concretos y más cercanos a la solución final del problema.

En este sentido, el proceso de refinación que se lleva a cabo cuando se trabaja en la resolución de estos problemas, es un proceso que gradualmente, va reduciendo los niveles de abstracción del problema, transitando desde el estado inicial del mismo, donde sólo se conocen un conjunto de características y condiciones del problema, instancia en la cual el resolutor desconoce por completo cuál es su solución, hasta llegar al programa computacional de solución.

Para estos efectos, la metodología de programación establece los fundamentos y los procedimientos para trabajar en la tarea de reducir los niveles de abstracción, en base principalmente, a la elaboración y el trabajo con distintos modelos.

Se evidencian entonces tres condiciones para resolver estos problemas, por una parte el manejo de los conocimientos de la estructura lógico-cognitiva, por otra, las habilidades de pensamiento abstracto dirigidas a la elaboración y la manipulación de modelos, y la habilidad para llevar a cabo la integración adecuada de los conocimientos y las habilidades de pensamiento abstracto.

Teniendo presente esta perspectiva, nos concentraremos ahora en clarificar cómo, los conocimientos de la estructura lógico-cognitiva y las habilidades de pensamiento abstracto y de integración, van interviniendo en el proceso de elaboración y manejo de los modelos, hasta alcanzar la resolución del problema que hemos considerado como ejemplo.

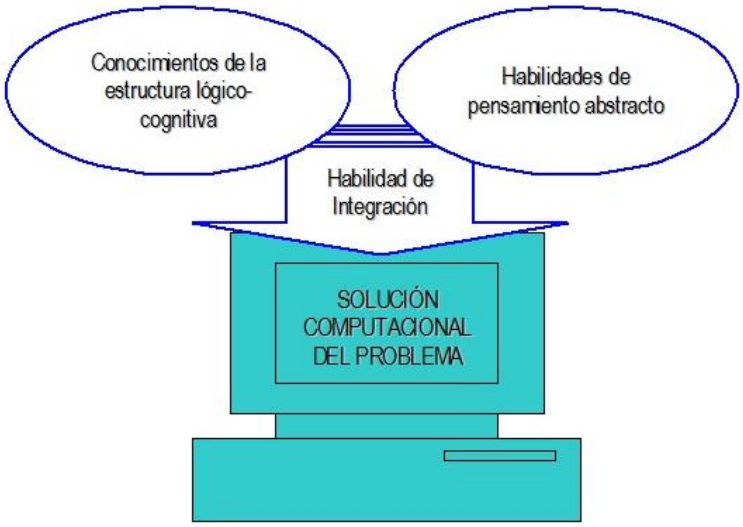

Figura 8: Condiciones para la resolución de los problemas de procesamiento de datos

La metodología considera el desarrollo de cuatro etapas para la resolución de estos problemas (análisis, diseño, codificación y prueba). A continuación describimos el procesamiento cognitivo que debería darse en cada una de ellas.

1.2.3.1. El procesamiento cognitivo en la etapa de análisis.

El marco conceptual y procedimental de la primera etapa de la metodología, está dirigido al desarrollo del análisis del problema. Su propósito, es establecer con suficiente claridad la definición y la especificación del problema. Definir el problema significa, identificar y establecer el espacio del problema y los elementos que permiten caracterizarlo.

La ejecución de esta tarea, demanda la elaboración de un modelo o representación del problema. Para la construcción de este modelo, se debe ejecutar un proceso de abstracción con el objeto de interpretar el enunciado del problema, e identificar las características relevantes del mismo.

En este contexto, el estudiante debería identificar el área o espacio del problema, además de caracterizar la información que se necesita para resolverlo, y los resultados que debe producir la solución. En esta identificación, está implicado el conocimiento declarativo, que para el caso del problema que usamos como ejemplo, está compuesto por todos aquellos conocimientos de base referidos a la ecuación de segundo grado.

Este modelo, es la base para elaborar la especificación del problema, la que contienen los detalles de la información y/o datos de entrada que se requieren para resolverlo, como también, cual es la información y/o datos de salida, los resultados, que debe producir la solución y el programa que se diseñe.

Para el caso de nuestro problema ejemplo, el correcto desarrollo de estas tareas debe producir la siguiente especificación:

- Área del problema: problema referido a la determinación de las soluciones de una ecuación de segundo grado de la forma general,
- Datos de entrada: los valores de los coeficientes A, B, y C
- Información y datos de salida:
  - Mensaje “la ecuación no es de segundo grado”,
  - Mensaje “la solución es imaginaria”,
  - Los valores que son solución de la ecuación de segundo grado.

Esta especificación es el producto que debe generar la ejecución adecuada de la primera etapa de la metodología. Con esto se ha dado el primer paso para reducir el nivel de abstracción del enunciado inicial del problema, obteniendo con ello, el modelo del espacio del problema.

Conforme a lo que hemos detallado, en la ejecución de esta primera etapa intervienen los conocimientos de la metodología, y en particular los referidos a los elementos conceptuales y procedimentales de la primera etapa, pero además, interviene los conocimientos declarativos propios del área del problema y las estrategias de pensamiento que permite llevar a cabo los procesos de abstracción.

Como hemos argumentado en apartados anteriores, la abstracción constituye una estrategia de pensamiento que es absolutamente necesaria, para la resolución de este tipo de problemas. Si bien en los conceptos y los procedimientos asociados con esta etapa, encontramos el porqué y el cómo ha de ejecutar la etapa, estos conocimientos no podrán ser aplicados, si el estudiante no dispone de una estrategia de pensamiento que permita identificar, seleccionar y organizar los elementos que permiten caracterizar la situación problema. Yu-Fen and Stephen (1994), han destacado la importancia que tiene para la PC, la habilidad cognitiva para construir los distintos modelos.

La abstracción es una capacidad intelectual, que se requiere para el desarrollo de prácticamente todas las etapas del proceso de resolución de estos problemas. Nuestra experiencia nos muestra que, la capacidad para realizar procesos de abstracción, es una de las dificultades importantes que tienen los estudiantes en los cursos de FP.

En el contexto del desarrollo de la etapa de análisis del problema, este proceso de abstracción requiere que el estudiante, maneje los conocimientos propios del espacio del problema, y demanda la capacidad de establecer relaciones entre los elementos que lo caracterizan, con el objeto identificar, interpretar y organizar los elementos que son relevantes, para así, establecer las condiciones bajo las cuales debe ser resuelto,

atendiendo además a las características de generalización, que son propias de las soluciones de estos problemas.

Nuestra experiencia nos muestra que aun cuando, nuestros estudiantes no tienen dificultades en el manejo de los conocimientos del área del problema, y son capaces de resolver casos particulares de él, tienen dificultades importantes para tratar con la generalización del mismo.

Para el caso particular de este ejemplo, no tienen dificultades para identificar, interpretar y organizar los elementos que son relevantes para una ecuación de segundo grado en particular, incluso cuando esta no se encuentra en su forma general, sin embargo, tienen dificultades importantes al momento de realizar estas tareas, cuando se les precisa de que han de pensar en cualquier ecuación de segundo grado. Creemos que la dificultad está, en la demanda por integrar los conocimientos que ellos poseen respecto de varios casos particulares de ecuaciones se segundo grado, es decir, pensar en la generalización del problema.

Para autores como Goei & Pieters (1991); Mayer (1987); Perkins, Schwartz, & Simmons (1988); Sloane & Linn (1988), los programadores novicios generalmente fallan en la construcción de los modelos que se requieren en la PC, y las investigaciones han mostrado que las preconcepciones, se muestran como el principal obstáculo de los estudiantes para aprender a programar.

Este primer modelo del espacio del problema que se refleja en las especificaciones del mismo, son el punto de partida para el desarrollo de la segunda etapa de la metodología, la cual se enfoca en el diseño de la solución.

1.2.3.2. El procesamiento cognitivo en la etapa de diseño

En el desarrollo de esta segunda etapa, intervienen los conocimientos conceptuales y procedimentales que definen el propósito y el procedimiento para ejecutarla, y los conocimientos declarativos y procedimentales que son propios del área del problema.

La estrategia de pensamiento abstracto se aplica ahora con el objeto de, identificar los procesos matemáticos y lógicos que se encuentran implicados en la solución. Este proceso de abstracción debe trabajar en primer término, relacionado el modelo del problema con las especificaciones del mismo, y su propósito es elaborar el modelo de procesos que servirá de base para diseñar la solución.

El paso siguiente es organizar el modelo de procesos en una secuencia lógica, con el objeto de establecer la organización secuencial de los procesos que intervendrán en la solución.

Basados en nuestra experiencia, la construcción y organización de este modelo de procesos, configura otra dificultad importante para los estudiantes de los cursos de programación. Aun cuando ellos, no presentan dificultades importantes en la compresión de los conceptos y los procedimientos asociados con la ejecución de esta etapa, como tampoco en el manejo de los conocimientos declarativos y procedimentales que son propios del área del problema, si evidencian dificultades importantes para realizar el proceso de abstracción que se requiere para la elaboración del modelo de procesos.

Existen otros casos en que si bien los estudiantes, son capaces de identificar los procesos de cálculo que se encuentran implicados, como por ejemplo, establecer que se debe calcular el argumento de la raíz para saber si la solución es imaginaria, son incapaces de organizar estos y otros procesos de cálculo, como una secuencia lógica que permita resolver el problema. La identificación de estos procesos de cálculo y la determinación de la organización lógica de los mismos, son tareas fundamentales para la generación de la solución.

En términos más específicos, en relación con el problema que tomamos como ejemplo, en nuestra práctica docente hemos observamos que si bien los estudiantes, no tienen dificultades para establecer los procesos que se requieren para resolver un caso particular de una ecuación de segundo grado, si presentan dificultades importantes a la hora de definir y organizar un proceso que considere todas las condiciones que son necesarias para resolver un problema de este tipo.

Para esta situación, significa definir y organizar un proceso para cualquier ecuación de segundo grado, con lo cual, la generalización del problema, nuevamente constituye una fuente importante de dificultades para los estudiantes.

A este respecto Rist (1995), ha estudiado las diferencias en el desempeño que tiene los programadores novicios, cuando se enfrentan a tareas de comprensión de un programa ya escrito, en comparación a cuando tratan con el diseño de un programa. En sus estudio muestra que la comprensión de programas no presenta mayores dificultades para los estudiantes, sin embargo, cuando trabajan en el diseño de los programas, tienen grandes dificultades en la definición y organización de los procesos, lo que significa que sus desempeños son malos cuando deben diseñar las solución.

Este autor señala además que, este bajo desempeño produce en ellos altos niveles de frustración, ya que no son capaces de identificar que parte del proceso de generación del programa no manejan.

Para el caso de nuestro problema, la secuencia lógica de los procesos se puede describir como:

- Ingresar los datos de entrada,
- Verificar si la ecuación es de segundo grado,
- Verificar si la solución es imaginaria,
- Encontrar las dos soluciones, y
- Mostrar los mensajes o datos de salida.

Este modelo preliminar de procesos debe ser refinado, para establecer las relaciones entre ellos, y de esa forma, alcanzar un modelo de procesos suficientemente detallado, el que podría ser expresado como:

- Inicio
- Ingresar el valor del coeficiente A.
- Verificar si el coeficiente A es cero, de ser así, se indica con un mensaje que la ecuación no es de segundo grado; en caso contrario, se continúa con el siguiente paso.
- Ingresar los valores de los coeficientes B y C
- Calcular el valor de la expresión $B^2 - 4 * A * C$
- Verificar si el valor de la expresión anterior es menor que cero, de ser así, se indica con un mensaje que las soluciones son imaginarias.; en caso contrario, se continúa con el siguiente paso.
- Calcular el valor de la expresión $\frac{-B+\sqrt{B^2-4*A*C}}{2*A}$ como primera solución.
- Calcular el valor de la expresión $\frac{-B-\sqrt{B^2-4*A*C}}{2*A}$ como segunda solución.
- Mostrar los valores de las dos soluciones.
- Terminar.

Para que este modelo de procesos detallado sea de utilidad para el diseño de la solución, debe ser formalizado mediante una representación algorítmica.

Para ello, utilizando el formalismo del concepto de algoritmo, el lenguaje algorítmico y los formalismos de la lógica de primer orden, se deben elaborar las representaciones algorítmicas que contienen las acciones y operaciones del modelo de solución, además de las operaciones lógicas que controlaran las secuencias de acciones y operaciones. En el curso de FP se trabaja con dos formas de representación algorítmica: Diagramas de flujo y Seudocódigo.

Desde la perspectiva del propio proceso de resolución, cada una de estas formas de representación algorítmica, presenta ventajas que justifican su utilización. Los diagramas de flujo, al usar símbolos que incluyen palabras claves y flechas que los conectan, constituyen una forma gráfica y esquemática que facilita la representación, e interpretación del flujo de las acciones y operaciones, que deben realizarse para obtener los resultados de un cierto procesamiento.

Para atender a las condiciones de generalización de las soluciones a los problemas de procesamiento de datos, los diagramas de flujo disponen de recursos esquemáticos, para expresar flujos alternativos controlados por condiciones de tipo lógicas, mediante las cuales se controlan las distintas variantes de procesamiento que pueden darse dentro de un mismo problema, para así responder a las condiciones de generalización que demanda la solución.

Para el caso del problema ejemplo que estamos ilustrando, estos casos pueden ser; ecuación con solución real o ecuación con solución imaginaria; ecuación con coeficiente A distinto de cero o igual a cero.

Por otra parte, los diagramas de flujo facilitan el seguimiento de las acciones y operaciones que han de ejecutarse, para cada caso que requiere el tratamiento de la generalización de la solución, lo cual, consecuentemente facilita la verificación de la correctitud del algoritmo de solución. Desde una perspectiva más general, los diagramas de flujos constituyen una técnica de representación algorítmica que es cercana a los modelos de procesos y a la organización de ellos.

Aunque los diagramas de flujo no fueron creados con propósitos pedagógicos, en nuestra práctica docente nos son de utilidad para esquematizar el modelo de solución del problema, para explicar y analizar la correctitud de la solución, y para trabajar sobre la visualización y la realización de las modificaciones que podría requerir un algoritmo que no es apropiado para resolver un cierto problema.

Los diagramas de flujo son de utilidad también, para ilustrar los detalles de funcionamiento de soluciones a ciertos problemas clásicos de procesamiento de datos, con los que tratamos en un primer curso de programación, como por ejemplo los algoritmos de ordenamiento y los de búsqueda.

Sin embargo, la representación algorítmica en diagrama de flujo no facilita la implementación de la solución como programa computacional, que es el siguiente paso metodológico. En otras palabras, no hay un paralelo directo entre un diagrama de flujo y un programa computacional escrito en un cierto lenguaje. Por tanto, para superar esta limitación, el diagrama de flujo requiere ser expresado en una forma más cercana a los lenguajes de programación estructurados, como es el caso de las representaciones algorítmicas basadas en pseudocódigo.

El pseudocódigo utiliza un conjunto de palabras claves y estructura, mediante las cuales se expresan las organizaciones de las distintas acciones y operaciones de la solución algorítmica, y que son muy similares a la forma de expresión que utilizan los lenguajes de programación estructurados. Con ello, aunque al igual que los diagramas de flujo, esta forma de expresión algorítmica no fue elaborada con propósitos pedagógicos, en nuestra práctica docente, resultan ser un recurso para

conseguir que los estudiantes, aprendan más fácilmente el proceso de convertir un algoritmo en un programa computacional.

Conforme a nuestra experiencia, la traducción de un algoritmo en diagrama de flujo, a su equivalente en pseudocódigo, constituye otras de las dificultades importantes para los estudiantes. Creemos que la dificultad se encuentra en la complejidad de usar y aplicar el modelo de lenguaje de programación, el cual incluye la necesidad de manejar, el modelo de computador como máquina de procesamiento de sentencias e instrucciones, que reconoce y maneja distintos tipos de datos, y que es capaz de interpretar y ejecutar organizaciones de sentencias e instrucciones.

Si bien estas dos formas de representación algorítmicas, poseen un lenguaje implícito que debe ser aprendido y manejado para construir las representaciones, dadas las características gráficas que posee la representación mediante diagramas de flujo, estas resultan más fáciles de entender y utilizar por parte de los estudiantes, incluso cuando las soluciones algorítmicas consideran estructuras más complejas, como condiciones anidadas, y/o al algún tipo de ciclo, los cuales también pueden estar anidados.

Por tanto, el recurso esquemático y gráfico que utilizan los diagramas de flujo facilitan el “ruteo” o seguimiento de las secuencias de acciones, tanto si estas tienen una estructura que es absolutamente secuencial (una sola secuencia de ejecución de las acciones), como cuando existen estructuras selectivas simples, dobles o anidadas, como también cuando existen estructuras repetitivas, como es el caso de los ciclos.

Las formas de representar estas últimas estructuras en el lenguaje de pseudocódigos, constituyen una de las dificultades importantes para los estudiantes. Creemos que las dificultades más específicas se encuentran en el entendimiento, la utilización y organización de las palabras claves que usan los pseudocódigos, para expresar las estructuras de tipos selectivas simples y dobles, y las estructuras repetitivas. Las dificultades son aún mayores, cuando se requiere el manejo de estas estructuras anidadas, es decir, estructuras selectivas dentro de otras, o estructuras repetitivas dentro de otras, o combinaciones de anidaciones de estructuras de estos tipos.

Conforme a las características que hemos descrito, en la figura 9 de la siguiente página, se muestra la representación algorítmica en diagrama de flujo, que se corresponde con el modelo de procesos que presentamos anteriormente.

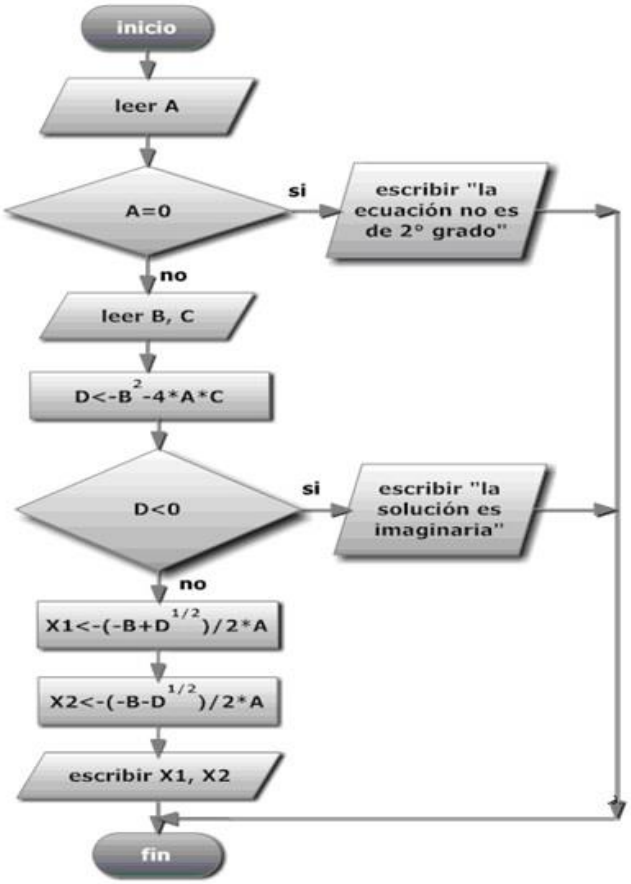

Figura 9: Diagrama de flujo para el algoritmo que encuentra las raíces solución de una ecuación de segundo grado (Caneo 2010)

La representación algorítmica en pseudocódigo que se corresponde con este diagrama de flujo es la siguiente:

```
inicio
leer a
si a=0 entonces
 escribir "La ecuación no es de segundo grado"
sino
 leer b, c
 d ← b^2-4*a*c
 si d<0 entonces
   escribir "La solución es imaginaria"
 sino
   x1 ← (-b+d^1/2)2*a
   x2 ← (-b-d^1/2)2*a
   escribir x1, x2
 fin_si
fin_si
fin
```

Con la construcción, validación y optimización de estas formas de representación algorítmica, finaliza la segunda etapa de la metodología, dando paso a la etapa de implementación del programa computacional de solución.

1.2.3.3. El procesamiento cognitivo en la etapa de codificación del programa

En el desarrollo de la tercera etapa, interviene dos tipos de conocimientos, los referidos a la etapa de codificación o implementación del programa, y los referidos al uso del lenguaje de programación. En el ámbito de los conocimientos procedimentales, también encontramos dos categorías, los referidos a los procedimientos para desarrollar la etapa, y los referidos al uso del entorno de programación. Estos últimos corresponden a conocimientos procedimentales de orden técnico, ya que se relacionan con el manejo de un software para editar, compilar y ejecutar programas.

En esto interviene también, la habilidad para elaborar un modelo de estructura del programa. Para la elaboración de este modelo, el estudiante debe realizar un proceso de abstracción, que opera sobre la representación algorítmica en pseudocódigo, y se centra en la identificación de las variables y los tipos de datos, y en las estructuras de programación que se deducen del algoritmo de solución. Luego este modelo, debe relacionarlo con los recursos que posee el lenguaje de programación para, declarar y manipular tipos de datos, y para organizar las estructuras de programación que pueden ser implementadas en el lenguaje.

Hecho este proceso, el paso siguiente es implementar las secuencias de acciones y operaciones, como instrucciones y estructuras del lenguaje de programación. Para el caso de nuestro problema de ejemplo, la implementación del programa es la siguiente:

```
#include<iostream.h>
#include<math.h>
main()
{float a,b,c,d,x1,x2;
 cin>>a;
  if (a==0)
    cout<<"\nLa ecuación no es de 2° grado\n";
  else
    {cin>>b>>c;
       d=(b*b-4*a*c);
       if (d<0)
            cout<<"\nLa solución es imaginaria\n";
       else
       {x1=(-b+sqrt(d))/2*a;
            x2=(-b-sqrt(d))/2*a;
            cout<<x1<<" "<<x2;}}}
```

Nuestra experiencia nos muestra que la elaboración del modelo de procesos, y la tarea de relacionar este modelo con el lenguaje de programación, es un proceso que no presenta grandes dificultades para los estudiantes, en especial cuando ellos cuentan con una buena comprensión e interpretación de los lenguajes algorítmicos.

Así también, en general no tienen dificultades para, comprender y manejar la sintaxis y la semántica del lenguaje de programación.

Sleeman et al. (1988), señalan que si bien para la implementación de un programa se requiere de un proceso de abstracción, cuando este se realiza sobre un objeto más concreto, como son las representaciones algorítmicas, el proceso tiene menor grado de complejidad que cuando se hace sobre objetos más abstractos, como es el enunciado del problema.

Algunas dificultades mayores se aprecian cuando deben tomar decisiones, respecto de los tipos de datos que deben utilizar y declarar, en especial para determinar el ámbito de valores posibles que puede tomar una variable. Así también, presentan algunas dificultades para determinar cuál es el tipo de implementación de ciclo más apropiado para una determinada repetición. Sin embargo, nuestra experiencia nos muestra que estas dificultades se superan rápidamente con las explicaciones respecto de ejemplificaciones y con ejercicios apropiados.

1.2.3.4. El procesamiento cognitivo en la etapa de prueba y depuración.

El proceso de resolución del problema culmina con la etapa de prueba y depuración del programa. En la ejecución de esta etapa, se encuentran implicados los conocimientos referidos a los conceptos de prueba y depuración de un programa, junto con el conocimiento referido a los procedimientos para desarrollar la prueba y depuración.

El procedimiento de prueba consiste básicamente en verificar si el programa de solución resuelve el problema que se enunció. Las pruebas se realizan inicialmente contrastando los resultados que arroja la ejecución del programa, con los resultados que se demandan en el enunciado del problema. Luego se realizan pruebas más específicas, como por ejemplo para el caso del problema que hemos desarrollado, se debería verificar que resultados arroja la ejecución del programa si se ingresa un valor de coeficiente $A = 0$ y el caso de las soluciones imaginarias.

El desarrollo de las pruebas de un programa no reviste mayores dificultades para los estudiantes, en especial cuando se trata de problemas simples, que cuentan con una definición del espacio del problema y con una especificación, suficientemente clara y simple.

Sin embargo cuando los problemas son más complejos, los estudiantes tienen muchas dificultades. Los problemas más complejos, tienen algoritmos de solución más complejos, generalmente con muchas estructuras, y/o muchas bifurcaciones controladas por condiciones lógicas, y por tanto, el programa de solución contendrá

un mayor número de instrucciones y estructuras condicionales y ciclos, y anidaciones de estructuras de este tipo.

Como señalamos en párrafos anteriores, estas dificultades ya aparecen en la implementación del programa que resulta del algoritmo de solución, las que al no ser resueltas, generarán errores en el programa resultante, los que se pondrán de manifiesto en la última etapa de la metodología, al momento de llevar a cabo las pruebas del programa.

Sin embargo también debemos considerar que, conforme a la complejidad de los problemas, los errores que se evidencian en la etapa de prueba del programa, podrían tener su origen en una o más de las etapas anteriores. Así por ejemplo, los errores en la etapa de análisis del problema, pueden producir una inadecuada definición del espacio y del modelo del problema, y consecuentemente una inadecuada especificación del mismo, y por tanto, a su vez, esto generará errores en el diseño de la solución y en las correspondientes representaciones algorítmicas, y consecuentemente se producirán errores en la implementación del programa.

Con esto queremos enfatizar que, dada la naturaleza del proceso de resolución de estos problemas y de las dificultades que tienen los estudiantes, para tratar con los distintos conocimientos y estrategias de pensamiento, que se encuentran implicadas, los errores pueden producirse en varias instancias de uso y aplicación de los conocimientos y estrategias.

Nuestras experiencias nos muestran que la etapa de diseño de la solución, es la más crítica para la generalidad de los estudiantes, le siguen las dificultades de la primera etapa, luego las dificultades para llevar a cabo la etapa de prueba y depuración del programa, siendo la tercera etapa la que representa las menores dificultades para ellos.

Durante el desarrollo de la asignatura, utilizamos las situaciones de ocurrencia de errores, para tratar con los estudiantes, los aspectos de detección, identificación y corrección de los mismos, y al mismo tiempo, usamos estos errores como fuentes de aprendizaje. Así, los errores pasan a constituirse en situaciones para identificar, cuáles son las causas que producen estos errores en el proceso, enfatizando de qué manera los conocimientos y las estrategias han sido mal utilizadas, y por sobre todo, utilizando estas fuentes de errores para corregir los aspectos que los producen.

En otras palabras, tratamos de impulsar en ellos el aprendizaje por la vía del error, contexto en el cual, nos parece que es absolutamente necesario que nuestros estudiantes se impliquen e involucren directamente en el proceso de resolución de los problemas, y en el propio proceso de aprendizaje.

Esta perspectiva tiene otra implicancia que creemos fundamental para el aprendizaje de la programación de computadores. Como hemos planteado en apartados anteriores, tanto la programación de computadores, como la ingeniería informática en un ámbito más general, tratan con el diseño de soluciones a problemas de procesamiento de datos, soluciones que en la mayoría de los casos son soluciones únicas, en el sentido de que no son soluciones conocidas para quien trabaja en su diseño.

Dadas esta característica, los profesionales del área de la informática deben desarrollar la capacidad y habilidad de analizar la eficacia y eficiencia de las soluciones que diseñan para un cierto problema, lo que implica, analizar el cómo ha sido ejecutado el propio proceso para llegar a una solución, y cómo se han tenido en consideración los distintos elementos que forman parte de ella.

Dicho en otros términos, para un informático, no solo ha de interesar si una cierta solución es apropiada para un cierto problema, sino que frente al caso en que la solución no sea apropiada, debe tener la capacidad de analizar y determinar qué elementos han sido considerados de manera inadecuada, y que hacen que la solución fracase, y de qué manera se deben corregir esos elementos para conseguir que la solución sea afectiva. Esta capacidad debe también estar dirigida a analizar y determinar de qué manera, una solución que siendo eficaz, puede mejorarse para hacerla más eficiente, teniendo en cuenta que son soluciones que hacen uso de un conjunto amplio de recursos de datos, software y hardware.

Retomando nuestro análisis respecto de las dificultades que tienen los estudiantes con el desarrollo de la última etapa, nuestra experiencia nos ha mostrado que la principal dificultad está en establecer los “casos de prueba”, entendidos estos como las situaciones bajo las cuales el programa de solución, debe ser probado con el objeto de verificar si la ejecución del mismo, produce los resultados esperados.

Para establecer estos casos de prueba, se ha de considerar el espacio del problema y la especificación del mismo establecida en la etapa de análisis, elementos que determinan los resultados que debe producir la ejecución. Los casos de prueba se deben diseñar tanto para las situaciones regulares de solución del problema, como para aquellos casos de carácter especial. Así por ejemplo para el caso del problema que estamos usando como ejemplo, los casos de prueba deben considerarse para valores de coeficientes A, B y C, que deberían producir raíces solución que no son imaginarias, y también para coeficientes que produzcan soluciones imaginarias, y para el caso de que el coeficiente A sea cero.

Si bien los estudiantes tienen algunas dificultades cuando tratan con la generación de casos de pruebas, para los primeros problemas que desarrollamos en clases, estas dificultades las superan rápidamente con la práctica y las orientaciones que les

damos. Las mayores dificultades las experimentan en la determinación de los casos de prueba para las situaciones de soluciones o resultados no regulares, aunque para el caso de este ejemplo, como para otros, las dificultades están en la determinación de cuáles son los valores o conjunto de valores que deben producir los resultados no regulares.

En esta etapa se llevan a cabo también las tareas de depuración del programa de solución, con el objeto de identificar y corregir los errores que se detecten en las pruebas. Como planteamos en párrafos anteriores, los errores en un programa de solución pueden producirse en cualquiera de las etapas del proceso de resolución, lo que hace que la tarea de identificar en qué etapa o etapas, se han producido los errores, sea lo que representa mayores dificultades para la generalidad de los estudiantes.

Esta tarea es más compleja para ellos, en la medida en que se trata con problemas más complejos, y cuando dadas las características de los problemas, las soluciones demandan mayores grados de generalización, como es el caso de problemas cuya solución debe ser efectiva para un conjunto restringido de datos o valores de entrada.

Así también, considerando que los errores pueden producirse en cualquiera de las etapa, o en más de una de ellas, los estudiantes tienen menos dificultades en la identificación y corrección de los errores, cuando estos se encuentran en la etapa de implementación del programa, y más dificultades cuando estos se producen en las etapas más tempranas del proceso de resolución, siendo lo más complejo de manejar para ellos, los errores en la etapa de diseño de la solución y en la especificación del problema.

Respecto de esto, McCauley et al. (2008) señalan que los aspectos más críticos para el proceso de depuración de programas es disponer de una adecuada representación del espacio del problema y de un adecuado modelo de estructura del programa.

Como hemos planteado, una adecuada representación del problema, es fundamental para comprender que resultados se esperan de la solución, en tanto el modelo de estructura del programa, es esencial para comprender cómo se ejecutan las secuencias de instrucciones. Estos mismos autores señalan que los estudiantes construyen modelos errados, o en algunos casos incompletos.

Como vemos, la construcción de los distintos modelos, en especial, los primeros que se construyen durante el proceso de resolución, constituyen la principal dificultad que tienen los estudiantes en los cursos de programación de computadores. A este respecto Robins et al. (2003), afirman que un buen programador, es un individuo que no solo tiene un adecuado dominio de los conocimientos, sino que por sobre todo dispone de estrategias de pensamiento, y en especial, es un sujeto que tiene la

habilidad para realizar procesos de abstracción, que le permitan ir creando los modelos que se requieren en cada parte del proceso de resolución de los problemas de programación.

Mullera and Haberman (2008), señalan que para superar estas dificultades es fundamental diseñar metodologías de enseñanza y aprendizaje, que promuevan en los estudiantes la adquisición y el desarrollo de habilidades de pensamiento abstracto y la capacidad de construir modelos. Estos autores enfatizan que para lograr esto, es necesario poner a los estudiantes en situaciones que les permitan hablar de sus modelos, validarlos y contrastarlos, como una forma de perfeccionar sus estrategias de pensamiento para la construcción de los modelos.

En la siguiente figura 10 en la página siguiente, presentamos un esquema en el que se ilustra la secuencia de desarrollo de las cuatro etapas de la metodología. Además, se muestran los tipos de conocimientos y estrategias de pensamiento que deben ir interviniendo en cada etapa, los modelos que se van generando y participando, y los productos intermedios que deben producirse con la ejecución de cada etapa, hasta llegar al programa computacional de solución.

En resumen, en estos apartados hemos analizado los elementos cognitivos que están implicados cuando se trata con el diseño, e implementación de soluciones computacionales a problemas de procesamiento de datos.

Para el caso particular de los tipos de problemas con que se trata en un primer curso de programación (que es el contexto de este trabajo investigativo), la resolución de ellos requiere el uso de conocimientos referidos a la formas declarativas y procedimentales de las operaciones básicas, y respecto de las formas de combinarlas mediante expresiones algebraicas que contengan varias de estas operaciones. En este contexto, se requiere también del manejo de las reglas de agrupación y precedencia de operadores.

También se requiere de conocimientos, tanto conceptuales como procedimentales, respecto de los algoritmos y de las formas de representación que son apropiadas para la resolución de los problemas de programación de computadores. En relación con las formas de representación algorítmicas, se requiere también el conocimiento del lenguaje algorítmico, cuestión que es esencial para interpretar, comprender y evaluar la eficiencia y eficacia de cada representación.

Otro de los conocimientos que se requiere, es el concepto de computador como máquina de cálculo y programable. Esta conceptualización es vital para entender, y aplicar las capacidades que tiene esta máquina para la automatización de soluciones. Como complemento a esto, se requiere también del conocimiento referido a un

lenguaje de programación, como medio para expresar los algoritmos como un programa.

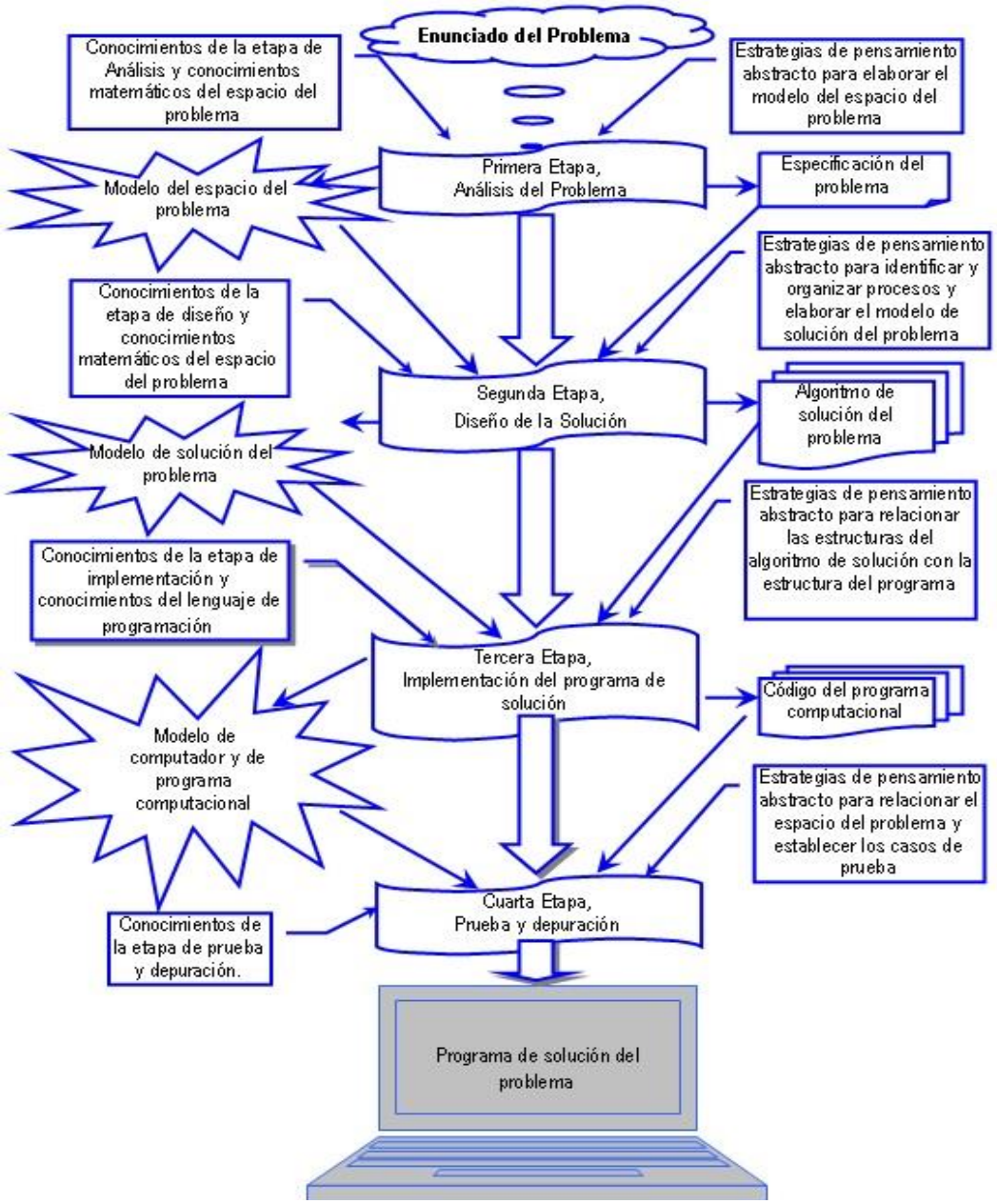

Figura 10: Las etapas de la metodología, los productos intermedios que se generar a partir del uso de las estrategias de pensamientos, y la construcción de los distintos modelos. (Caneo, 2010)

Ahora bien, conforme a lo expuesto y ejemplificado, a la hora de aplicar este conjunto de conocimientos, estos se han de integrar con los conceptos y los procedimientos de la metodología de resolución de los problemas y la programación. En este proceso de integración juega un rol fundamental, el conocimiento estratégico, el trabajo con abstracciones y el manejo de modelos de representación de los distintos estados del

problema, desde su enunciado hasta su solución expresada como programa computacional.

En síntesis, en estos apartados hemos expuesto los elementos cognitivos que están implicados cuando se trata con el diseño e implementación de soluciones computacionales, para problemas de procesamiento de datos.

Para el caso particular de los tipos de problemas con que se trata en un primer curso de programación, la resolución de ellos requiere el uso de conocimientos referidos a las formas declarativas y procedimentales, de las cuatro operaciones básicas y las formas para combinarlas en expresiones algebraicas que contengan varias de estas operaciones, y en este contexto, se requiere también del manejo de las reglas de agrupación y precedencia de operadores.

Se requiere también, de conocimiento tanto conceptual como procedimental, respecto de los algoritmos y sus formas de representación, lo que incluye el conocimiento del lenguaje algorítmico.

Otro de los conocimientos requeridos es el concepto de computador como máquina de cálculo y que puede ser programada, y como complemento de esto, se requiere del conocimiento de las formas declarativas y el uso de un lenguaje de programación.

Además del manejo de estos conocimientos, se requiere de la capacidad para integrarlos y sistematizarlos en la metodología para la resolución de problemas de programación.

Por último, para resolver este tipo de problemas, se requiere también de conocimiento de tipo estratégico, contexto en el cual, la elaboración y el manejo de abstracciones y modelos, es fundamental para el diseño de las soluciones. Este tipo de conocimiento, conforme a lo que hemos detallado, es el que permite identificar, integrar y usar, todos los otros conocimientos, al momento de tratar con la búsqueda, el diseño y la implementación de una solución para un problema particular.

Conforme a lo planteado, es precisamente el aprendizaje y manejo del conocimiento estratégico, las abstracciones y los modelos, lo que presenta las mayores dificultades para los estudiantes.

1.3. El problema de la enseñanza de la PC

Habiendo presentado en los apartados anteriores, la estructura lógico-conceptual y lógico-cognitiva de los conocimientos que intervienen en la PC, en este apartado desarrollamos el análisis general del problema de la enseñanza de esta disciplina, a partir de lo cual, pretendemos ir también estableciendo los criterios de orden

metodológico que nos permitan justificar la dimensión didáctica, que hemos desarrollado en nuestro modelo de intervención.

1.3.1. Las estrategias de pensamiento que deben enseñarse

Conforme a lo que hemos presentado en los apartados anteriores, en la resolución de los problemas de procesamiento de datos, intervienen tanto conocimientos como estrategias de pensamiento. Los conocimientos son de dos tipos, conceptuales o declarativos y procedimentales, en tanto las estrategias son básicamente dos, el pensamiento abstracto, y la construcción y el trabajo con modelos.

Davies (1993), ha destacado la importancia que tiene para la tarea de programar computadores, el dominio de los conocimientos y de las estrategias, recalcando que solo disponer del conocimiento no es suficiente, citando por ejemplo que, aunque un individuo tenga un amplio dominio de la sintaxis y la semántica de un lenguaje de programación, éste no será capaz de diseñar programas, si no maneja las estrategias de resolución de estos problemas.

Ormerod (1990), señala que la capacidad para elaborar, lo que él llama “planes” o “esquemas”, es fundamental para la PC. Para este autor, un plan consiste en un conjunto de proposiciones que son organizados por sus contenidos semánticos. Para Rogalsky & Samurçay (1990), un plan, más que una interpretación estática respecto de algo, es una “acción orientada” hacia algo, es decir, una estrategia.

Confirmando la importancia que tiene la elaboración de un plan, Rist (1995, p 510) señala:

> *“There is considerable evidence in the empirical study of programming that the plan is the basic cognitive chunck used in program design and understanding”*

La construcción de un plan o un esquema, requiere del uso de ciertas estrategias de pensamiento, en especial de las estrategias de pensamiento abstracto. Como hemos señalado, el pensamiento abstracto es fundamental para ir construyendo, los distintos modelos de representación que demanda la resolución de los problemas de procesamiento de datos.

Conforme a lo expuesto, en el proceso de resolución de un problema de procesamiento de datos se encuentra implicada la construcción de varios modelos. Estos modelos en las primeras etapas, son muy abstractos, hasta llegar a un modelo muy concreto, como es del programa computacional, el que al ser ejecutado por un computador real, deberá producir los resultados que se esperan de la solución.

Además, los distintos modelos están fuertemente relacionados y tiene una alta dependencia entre ellos, puesto que el modelo del espacio del problema da pie para

la construcción del modelo de solución, a su vez, este modelo de solución del problema, permite la construcción del modelo del programa de solución, y todos estos modelos como conjunto, dan pie para la construcción del modelo de prueba y depuración del programa final.

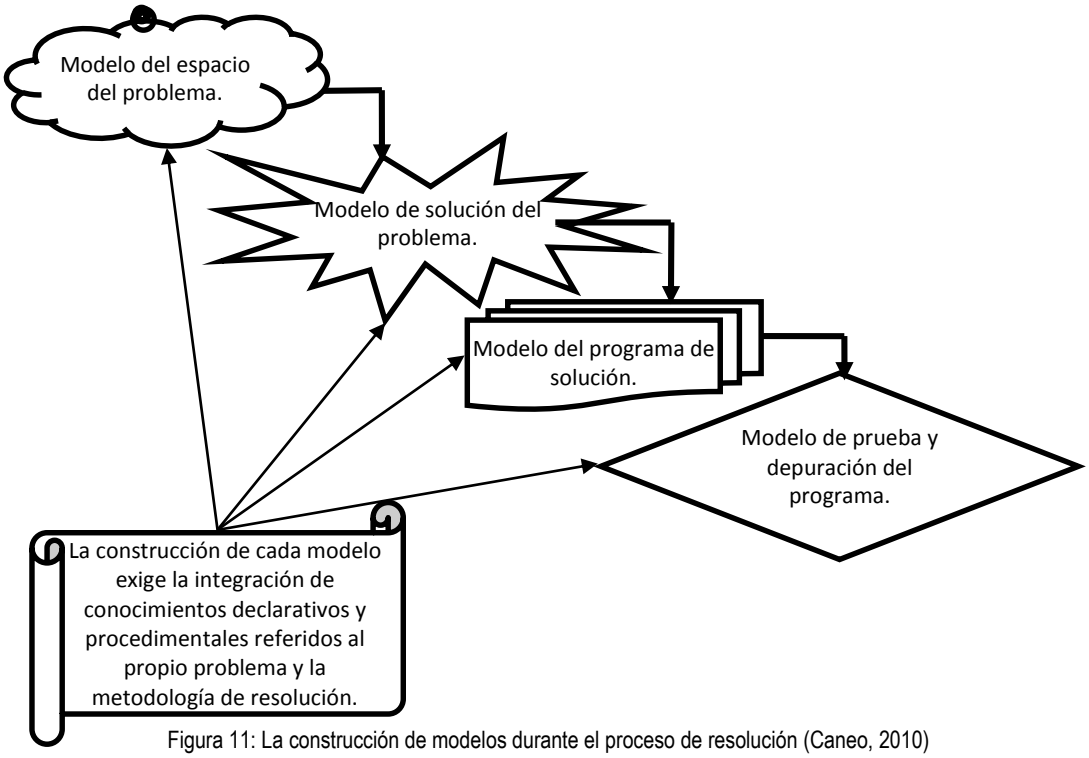

Figura 11: La construcción de modelos durante el proceso de resolución (Caneo, 2010)

El pensamiento abstracto, y la capacidad de construir los distintos modelos de representación que requiere la actividad intelectual de la PC, constituyen una habilidad cognitiva. Anderson (1980) señala que el término habilidad cognitiva, se refiere a la capacidad para realizar variados procedimientos intelectuales entre los que se encuentra la habilidad para resolver problemas matemáticos, físicos y en nuestro caso, para resolver problemas de programación.

Además de la construcción de los modelos, la PC demanda la habilidad para relacionar los conocimientos conceptuales y los procedimentales, y además la capacidad para determinar cómo estos conocimientos habrán de ser aplicados a la solución de un problema. En estos términos, las actividades intelectuales para la resolución de estos problemas, se orientan al desarrollo de un proceso, de carácter constructivo, y no un proceso de simple uso de determinados conocimientos, que siguen un patrón estandarizado. Cada problema requiere de una construcción e integración de conocimientos diferente.

Así, disponer de este tipo de habilidades es fundamental para la PC, en especial, si pensamos que cada problema de programación es distinto a otro, y por eso hablamos de problemas de programación y no de ejercicios de programación. La consecuencia

directa de esta diferencia es que la identificación, uso, integración y organización de los conocimientos conceptuales y procedimentales que se construye para un problema, no puede aplicarse como un patrón estandarizado para todos los problemas de programación.

Visto en términos más amplios, el trabajo en todas las áreas de conocimientos de las CcC demanda este tipo de habilidades. Así por ejemplo en el desarrollo de proyectos de software, esta habilidad juega un rol importante y vital, por lo que para un estudiante de CcC es esencial disponer de ella.

Varios autores coinciden en señalar que el pensamiento abstracto y la capacidad para construir y manejar diferentes tipos de modelos, son habilidades que cualquier estudiante de programación no solo debe adquirir, sino que por sobre todo, desarrollar, dominar y perfeccionar (Astrachan et al. (1998); Bucci et al. (2001); Caspersen & Bennedsen (2007); Evans (1996); Hazzan (2006); Kramer (2007); Muller (2007); Ye & Salvendy (1996))

En estos términos, la abstracción se constituye en la herramienta de pensamiento que, posibilita y facilita la construcción de distintos modelos que se requieren en programación. Así por ejemplo, al realizar el proceso de análisis de un problema, la abstracción implica, ver el problema de forma tal que sea posible extraer de él toda la información que es esencialmente relevante, para diseñar su solución, ignorando la información que es irrelevante a este propósito. El paso siguiente, es utilizar esta información relevante para construir el modelo del espacio del problema.

De manera similar, al aplicar el proceso de abstracción en la etapa de diseño de la solución, implica identificar todos los procesos de cálculo que son esenciales y relevantes para la solución, con lo cual se podrá dar el siguiente paso, que consiste en organizar estos procesos para elaborar el modelo de procesos y el modelo de estructura de procesos.

Para Muller and Haberman (2008) las principales características de la abstracción son:

- La generalización de ejemplos específicos
- La identificación, extracción y aislamiento de los componentes esenciales, e
- Ignorar u ocultar los elementos irrelevantes

Estos mismos autores distinguen dos facetas del proceso de abstracción:

1.- El reconocimiento de patrones y
2.- La identificación de estructuras.

Para nosotros la identificación de estas dos facetas, marca la diferencia entre lo que es el proceso de abstracción y el producto o productos que genera el proceso de

abstracción, que es un modelo o una representación del objeto que ha sido sometido al proceso de abstracción.

Conforme a lo que hemos expuesto, el pensamiento abstracto como habilidad de pensamiento superior, y la construcción de diferentes modelos, que resultan de la aplicación del proceso de abstracción, son formas de pensamiento y elaboración, que juegan un rol fundamental en la resolución de los problemas de procesamiento de datos. Para muchos autores, el dominio de estas habilidades marca la diferencia entre un programador novato y un experto, quién dispone y domina estas habilidades.

Así entonces, nos parece que el aspecto más crítico en el cual debe enfocarse el diseño del proceso de enseñanza y aprendizaje de los FP de computadores, es precisamente, en la adquisición y el desarrollo de las habilidades que hemos señalado.

Las habilidades para llevar a cabo estos procesos de abstracción, deben estar complementadas con las capacidades para identificar, integrar y organizar diversos tipos de conocimientos.

Así, para elaborar el modelo del espacio del problema se requiere la identificación de los conocimientos declarativos que son propios del tipo de problema con que se trata. Para el modelo de solución del problema, se requiere identificar los conocimientos procedimentales que son atingentes a dicho espacio del problema, a partir de lo cual se podrá hacer la integración y organización de ellos, para construir el algoritmo de solución.

Luego, el modelo de programa de solución requiere de la identificación, integración y organización de los conocimientos declarativos y procedimentales del lenguaje de programación, para construir el programa.

Y finalmente para el modelo de prueba y depuración del programa, se requiere la integración y organización de conocimientos declarativos y procedimentales, del propio problema y los referidos a la realización de la prueba y depuración.

En estos términos, la actividad intelectual de resolver problemas de programación, demanda del resolutor habilidades de pensamiento abstracto dirigidas a la construcción de distintos modelos, y de las habilidades para identificar, integrar y organizar, conocimientos de tipo metodológicos y los conocimientos que son propios de tipo de problema con que trata.

Para nosotros, el desarrollo de estas habilidades está fuertemente determinada por la experiencia que tiene el resolutor en el desarrollo de esta actividad intelectual, por lo que creemos que la enseñanza y el aprendizaje de la programación, debe estar

fuertemente mediada por la práctica intencionada, organizada y permanente en la ejecución implicativa del proceso de resolución de los problemas de programación.

Esta práctica ha de ser gradual, en términos de la complejidad de los conocimientos y las estrategias que se van demandando, ya que la idea es ir desarrollando estrategias de resolución, cada vez más completas y efectivas, como es el caso de aquellas estrategias que le permiten al resolutor, evaluar la efectividad de cada parte del proceso de resolución, en otras palabras, desarrollar la capacidad para determinar, si lo que se está haciendo para resolver un problema, realmente conduce a la solución deseada.

En síntesis, los dos aspectos de mayor complejidad que se encuentran implicados en la resolución de problemas de procesamiento de datos son, las estrategias de pensamiento abstracto, para la elaboración de los distintos modelos de representación que se requieren en el proceso de resolución, y la identificación e integración de diversos tipos de conocimientos, como los declarativos y procedimentales, referidos al propio problema que se intenta resolver, y los conocimientos declarativos y procedimentales referidos a la programación de computadores.

Además de estos conocimientos, también presenta una complejidad importante el desarrollo del conocimiento estratégico, el cual como hemos señalado, es el conocimiento que se construye, como resultado de la utilización de los conocimientos anteriores, cuando se logra resolver de manera exitosa un cierto problema. En estos términos, este conocimiento es fundamental, ya que es el que pasa a conformar el conocimiento de experto del programador.

1.3.2. Enfoques para la enseñanza de la PC

Robins et al. (2003), sugiere que la clave para que los novicios aprendan programación, es que estos sean puestos en situaciones efectivas de resolución de problemas de programación, por sobre situaciones en las que sólo se privilegie, la presentación de los conocimientos y la ejercitación repetitiva de problemas de un mismo tipo. Esto significa que, lo fundamental es trabajar las estrategias de programación y la aplicación de los conocimientos, a lo cual nosotros agregamos el trabajo sobre las estrategias de pensamiento abstracto dirigidas a la construcción de los modelos.

Para Clancy y Linn (1999), aprender programación significa aprender a construir modelos y estrategias de pensamiento de orden superior, esto debido a que las experiencias muestran que los programadores novicios, no infieren los modelos de manera natural, y por lo tanto, es responsabilidad de los profesores, crear problemas y situaciones de aprendizaje apropiados, y por sobre todo, condiciones y ambientes

de enseñanza, que faciliten y apoyen a los estudiantes en la comprensión, elaboración y uso de estos modelos.

Varios autores destacan la necesidad de que en los procesos formativos en CcC se trabaje de manera especial sobre el desarrollo de las habilidades de pensamiento abstracto y la construcción de modelos, como una cuestión que ha de tener carácter transversal a todo el currículo de formación profesional.

Conforme a lo planteado y analizado hasta ahora, y basados también en nuestra experiencia docente, consideramos que existen dos enfoques que son fundamentales, para tratar con el desarrollo de esta asignatura: uno basado en el uso de situaciones de aprendizaje, diseñadas para lograr el desarrollo de las capacidades y habilidades para la PC; y el segundo, basado en el enfoque constructivista-social, como facilitador del aprendizaje colectivo e individual de las capacidades y habilidades.

1.3.2.1. Enfoque basado en las situaciones de aprendizaje.

Uno de los aspectos de importancia que nos parece fundamental de considerar a la hora pensar en el diseño de un modelo de enseñanza de la PC, está referido a la necesidad de crear condiciones para el desarrollo de aquel tipo especial de conocimiento que hemos referido como estratégico.

Para Monereo *et al.* (1998), el conocimiento estratégico es el que construye el resolutor experto, para una situación determinada de resolución de problema, y que puede reactualizar parcialmente, si la situación problema que enfrenta, tiene elementos similares a los de otra situación en la que haya utilizado eficazmente su conocimiento estratégico.

Por tanto, este es un tipo de conocimiento que pasa a formar parte de la estructura lógico-cognitiva del resolutor, que se va construyendo como resultado de sus propias experiencias en las tareas de resolución de los problemas, y que se enriquece y perfecciona, en la medida en que se reactualiza con el trabajo en situaciones en las cuales hace uso de parte de su acervo de conocimiento estratégico.

Así, este tipo de conocimiento se va construyendo a partir de cada experiencia exitosa de resolución de estos problemas. Para Monereo et al. (1998), el conocimiento estratégico se va construyendo en la medida que el resolutor va desarrollando, combinaciones de conocimientos conceptuales y procedimentales cuando trabaja en la resolución de un cierto problema.

Este mismo autor destaca que el desarrollo de este tipo de conocimiento, es esencial para alcanzar la capacidad de ser un buen resolutor de problemas, entendiendo que

un buen resolutor de problemas, es un individuo que no solo dispone de una buena base de conocimientos conceptuales y procedimentales, sino que además, dispone de una buena base de conocimiento estratégico, y dispone también de la capacidad de usar este conocimiento estratégico aplicándolo a la resolución de problemas diferentes, es decir, cuando ha desarrollado la capacidad de transferir su conocimiento estratégico, para aplicarlo a la resolución de otros problemas.

Puesto esto en términos de las demandas formativas de los estudiantes de CcC, el desarrollo del pensamiento estratégico es una cuestión fundamental, ya que prácticamente todas las asignaturas del currículo de formación profesional tratan con el diseño de soluciones a problemas.

Así por ejemplo, la asignatura de Ingeniería de software trata con el diseño de sistemas de software para el tratamiento de datos; la asignatura de Redes de computadores, trata con el diseño de sistemas de comunicación basados en computadores, para soportar diversos servicios de comunicación, etc.

Si lo vemos ahora desde la perspectiva del perfil profesional, en las recomendaciones curricular CC2001 (CC2001, (2001)) se señala expresamente que un profesional de CcC, debe disponer de suficientes habilidades para la resolución de problemas, ya que por su naturaleza, son profesionales capaces de diseñar sistemas computacionales, que resuelvan diversos tipos de problemas referidos al tratamiento de la información.

Conforme a nuestra experiencia, la generación de conocimiento estratégico es posible cuando, el estudiante es puesto en situaciones que favorecen el desarrollo de un sistema de regulación y de utilización consciente, reflexiva y eficaz de los conocimientos. Para Monereo *et al.* (1998), esto supone crear situaciones de enseñanza y aprendizaje, que obliguen a un constante ajuste de la actividad cognitiva del sujeto, para adaptarla a los cambios y variaciones que presentan las situaciones problemáticas que se le plantean.

Esto implica la toma decisiones de cuáles conocimientos conceptuales y procedimentales, se deben recuperar y cómo hay que utilizarlos para construir la respuesta a una situación específica. Pero en un estado de dominio de esta práctica, también implica la toma de decisiones respecto de cuáles conocimientos estratégicos de los que ya se tienen, son de utilidad para abordar la solución de un nuevo problema.

Complementariamente, cuando un estudiante de un curso de FP consigue generar este tercer tipo de conocimiento, ha logrado desarrollar también estrategias de aprendizaje que, son definidas por Monereo *et al.* (1998), como “procesos de toma de decisiones, conscientes e intencionales, en los cuales elige, recupera y aplica, de

manera coordinada, los conocimientos que necesita para cumplimentar una determinada demanda u objetivo, dependiendo de las características de la situación educativa en que se produce la acción".

De lo que hemos analizado hasta este punto podemos establecer que los aspectos claves para el aprendizaje de la PC son:

- La capacidad de utilizar conocimientos declarativos y procedimentales que generalmente corresponden a conocimientos de tipo matemáticos,
- El aprendizaje de los conocimientos declarativos y procedimentales de la disciplina,
- La capacidad de aplicar y usar los conocimientos declarativos y procedimentales de la disciplina,
- La capacidad para integrar y organizar los conocimientos matemáticos y los conocimientos de la disciplina,
- La adquisición y el desarrollo de habilidades para el pensamiento abstracto y la construcción de modelos,
- La adquisición y desarrollo del conocimiento estratégico, y
- La capacidad de utilizar y manejar el conocimiento estratégico.

En atención a estos aspectos del aprendizaje, consideramos que el enfoque que es pertinente para abordar el proceso de enseñanza y aprendizaje de los fundamentos de programación de computadores, debe orientarse a la creación de situaciones que sean efectivas para:

- Adquirir y desarrollar las habilidades de pensamiento abstracto y para la construcción de modelos,
- Adquirir y desarrollar la capacidad de integrar los diversos tipos conocimientos implicados, y
- Adquirir y desarrollar la capacidad para construir el conocimiento estratégico.

1.3.2.2. Enfoque basado en el constructivismo y el constructivismo social

Para estos efectos, la propuesta didáctica que hemos elaborado y puesto en práctica en el presente trabajo investigativo, se sustenta en el enfoque de enseñanza y aprendizaje constructivista, pero más específicamente en el constructivismo social.

Basados en la experiencia que hemos recogido durante varios años dictando cursos de programación de computadores, consideramos que este es un enfoque apropiado para tratar con el aprendizaje de la programación, y para desarrollar las estrategias, las capacidades y habilidades que hemos descrito.

El constructivismo social es un enfoque que se sustenta en la idea de que los aprendizajes se construyen socialmente, mediante un proceso de naturaleza dinámica, que se produce como resultado de las interacciones y actuaciones del

sujeto con el medio, y/o con situaciones que se producen en el medio. Este enfoque, considera que la realidad se construye socialmente, que esta realidad no es externa al sujeto, sino que es una construcción individual que se elabora a partir de las relaciones interactivas entre los sujetos y el entorno.

En nuestro ámbito esto significa que, los conocimientos, las capacidades, las habilidades de pensamiento y las estrategias para resolver los problemas de procesamiento de datos, pueden ser aprendidas y desarrolladas mediante la implicación directa de los estudiantes en las propias tareas de resolución de los problemas, y favorecidas por la co-construcción con otros.

Robins et al. (2003), afirman que si bien en mucha de la literatura referida a los aspectos instruccionales en CcC, no se hace una referencia explícita a la teoría educativa del constructivismo, el modelo constructivista se da como supuesto, o a lo menos, se parte de los modelos cognitivos del constructivismo, para establecer las características de los aprendizaje en este ámbito.

Basándonos en nuestra experiencia docente en la asignatura, consideramos que el constructivismo, es un enfoque cognitivo natural para abordar el aprendizaje de la resolución de problemas de procesamiento de datos, puesto que las propias soluciones a los problemas resultan de una actividad intelectual constructiva, donde cada solución, debe ser creada a partir de la identificación y la integración de los distintos tipos de conocimientos que han de intervenir en la solución del problema, y los elementos de la metodología que se usan y aplican para diseñar e implementar la solución.

En este contexto vemos el constructivismo, como un enfoque de enseñanza y aprendizaje mediante el cual buscamos ayudar a los estudiantes a internalizar, reacomodar, y/o transformar la nueva información que va recibiendo durante su proceso de aprendizaje.

Para Grennon y Brooks (1999), esta transformación tendría lugar a través de la creación de nuevos aprendizajes, lo que resulta del surgimiento de nuevas estructuras cognitivas, que permiten enfrentarse a situaciones iguales o parecidas en la realidad. Bajo esta perspectiva, el aprendizaje lo entendemos como una actividad personal, enmarcada en contextos funcionales, significativos y auténticos, pero que es potenciada por la interacción social.

Siendo más específicos y desde la perspectiva de la actividad intelectual, cuando el resolutor trata con la búsqueda de la solución a un problema, parte de los elementos de conocimientos que posee, como el conocimiento matemático respecto de las ecuaciones de segundo grado en el caso del problema ejemplo que usamos anteriormente. A partir de este conocimiento de base, necesita hacer acomodaciones

y adaptaciones para enfrentar la generalización del problema, alcanzándose entonces un primer estadio del proceso constructivo hacia la solución del problema.

Para nosotros, estas acomodaciones y adaptaciones son parte del aprendizaje que se requieren para llevar a cabo el proceso de análisis del problema, ámbito en el cual entran en juego además el uso de estrategias de pensamiento abstracto, que permiten la definición del espacio del problema, y la elaboración del modelo de problema que se quiere resolver.

En otras palabras, estamos hablando de un proceso constructivo, que ha de producir aprendizaje, el que resulta de las acomodaciones y adaptaciones de los conocimientos, lo que además producirá nuevos conocimientos. Este conjunto de conocimientos permite la construcción del espacio del problema, lo que en el ámbito metodológico es esencial para la elaboración de la especificación del problema.

Si nos ubicamos en la etapa de diseño de la solución, la construcción de conocimientos se orienta al diseño de la solución algorítmica. Acá debe darse una nueva acomodación, en el sentido de que si bien los estudiantes conocen el algoritmo básico para resolver un determinado problema, requieren hacer ciertas acomodaciones para elaborar un nuevo algoritmo, que responda a las nuevas condiciones de un cierto problema y a los grados de generalización del mismo.

Más específicamente, si pensamos en el problema que hemos utilizado como ejemplo, los estudiantes conocen el procedimiento general (o fórmula) para encontrar las raíces que son solución de una ecuación de segundo grado, pero ahora se demanda la acomodación de este conocimiento para elaborar un algoritmo que permita resolver cualquier ecuación de segundo grado, y que además, permita obtener resultados esperables para ciertas condiciones de borde, como por ejemplo, el caso de las soluciones imaginarias. Estas acomodaciones permitirán la construcción de nuevos conocimientos.

En el desarrollo de estas dos etapas, la construcción de los nuevos conocimientos está determinada por acomodaciones de conocimientos que los estudiantes ya poseen, a los cuales se adicionan los conocimientos que son propios de la metodología de resolución de estos problemas.

Para el desarrollo de la etapa de implementación, estos nuevos conocimientos se han de integrar con los conocimientos referidos a los propósitos de esta etapa, y con los referidos al lenguaje de programación, con lo cual se construyen nuevos conocimientos.

De manera similar, para el desarrollo de la etapa de prueba y depuración del programa, se construyen nuevos conocimientos que resultan de la integración de los

ya construidos, con los que son propios de los propósitos de esta etapa, y los referidos a las técnicas para elaborar los casos de prueba que se requieren para verificar la correctitud del programa resultante.

En esta perspectiva, el aprendizaje de los conocimientos y el desarrollo de las capacidades para resolver problemas de procesamiento de datos, lo concebimos como un proceso constructivo, que se da en términos de un “aprender haciendo”, en el cual, cada estudiante, a partir de ciertos conocimientos que posee y maneja, debe ir acomodándolos, para construir nuevos conocimientos los que integra con los conocimientos de la metodología, los del lenguaje de programación y los de las técnicas y procedimientos de prueba de programa.

Junto con ello, en este proceso va también desarrollando y perfeccionando las estrategias para utilizar estos conocimientos, y las capacidades y habilidades más específicas, que demanda la solución de estos problemas.

Para nosotros este proceso de construcción tiene dos instancias que durante el proceso de aprendizaje han de estar en permanente interacción de ocurrencia:

- La instancia de construcción personal y
- La instancia de construcción colectiva,

donde entendemos que esta última actúa como facilitadora y potenciadora de la primera.

Así también, consideramos que el aprendizaje y el desarrollo de las estrategias de pensamiento que se requieren para la elaboración, y el trabajo con los distintos modelos, es también un proceso de carácter constructivo que se debe dar en estas instancias.

Para nosotros, cada estudiante debe construir sus habilidades para tratar con el pensamiento abstracto y con la construcción de los modelos de representación, pero esto puede ser favorecido y potenciado por la interacción con los otros, que se encuentran abocados a la misma tarea.

Y por último, cada estudiante debe construir, desarrollar y perfeccionar las capacidades para elaborar el conocimiento estratégico, que como hemos indicado, es el conocimiento más importante en el proceso de resolución de estos problemas, ya que no sólo sirve para resolver un determinado problema de procesamiento de datos, sino que es el conocimiento de base que es de utilidad para resolver problemas más complejos.

Todo este proceso constructivo, el referido al manejo y uso de los conocimientos, y el referido al conjunto de capacidades y habilidades que destacamos, configuran el

desarrollo de las estrategias de resolución de los problemas de procesamiento de datos, que es el objetivo fundamental de los primeros cursos de PC.

Vygotsky fue un auténtico pionero al formular algunos postulados que han sido retomados por la psicología, varias décadas más tarde, y han dado lugar a importantes hallazgos sobre el funcionamiento de los procesos cognitivos. Quizá uno de los más importantes, es el que señala que todos los procesos psicológicos superiores (comunicación, lenguaje, razonamiento, etc.) se adquieren primero en un contexto social y luego se internalizan.

Pero precisamente, esta internalización es un producto del uso de un determinado comportamiento cognitivo en un contexto social. Para este autor, con el cual concordamos, todas las funciones psicológicas superiores se originan, como relaciones entre seres humanos. (Vygotsky, 1978).

Así, basados en nuestra experiencia, creemos firmemente que el constructivismo social, es una poderosa herramienta teórica para fundamentar el diseño de una metodología de enseñanza y aprendizaje para el desarrollo de la asignatura de FP, ya que, consideramos que constituye un referente teórico adecuado para la definición de modelos de enseñanza y aprendizaje, que apunten a la mejora y el potenciamiento de los aprendizajes de nuestros estudiantes, en especial, para abordar aquellos aspectos de la enseñanza y el aprendizaje que hemos señalado como críticos.

Complementado esta idea, creemos también que, la mejor forma de aprender sobre la resolución de problemas de programación, es trabajando sobre el propio proceso constructivo de las soluciones y en la medida que ello ocurra, el estudiante tendrá las mejores oportunidades para construir los diversos tipos de conocimientos capacidades y habilidades que se requieren.

Es un aprendizaje que se desarrolla haciendo, modificando y construyendo sobre los conocimientos que el estudiante tiene, e integrando con ellos los nuevos conocimientos que son propios de las nuevas tareas que se abordan, y esta modificación, construcción y aprendizaje de nuevos conocimientos, se favorece y se potencia con la mediación social, ya que el constructivismo social genera oportunidades para desarrollar y perfeccionar las herramientas cognitivas.

Otro aspecto que nos parece importante es que, el constructivismo social favorece las acciones de regulación de lo que se aprende y por tanto, para nosotros, al ser aplicado a la resolución de los problemas de procesamiento de datos y la programación, nos permitirá favorecer el desarrollo y el perfeccionamiento de la capacidad de regular el desarrollo, y la ejecución del propio proceso de resolución de los problemas, principalmente en lo referido a la aplicación de los conocimientos, y

las estrategias de pensamiento que se requieren para llevar a cabo este proceso de manera exitosa.

Como resumen de lo que hemos expuesto, se deduce que en la resolución de problemas de programación, intervienen conocimientos de tipo declarativos, de tipo procedimentales y estrategias de pensamiento, siendo el aprendizaje de estas últimas, lo que representa las mayores dificultades para los estudiantes.

Para varios autores, tanto las habilidades de pensamiento abstracto y la construcción e interpretación de modelos, son fundamentales para el abordaje de las tareas que son propias de la resolución de estos problemas, habilidades que en un ámbito más general, debiera desarrollar y poseer cualquier estudiantes de CcC.

En estos términos, la actividad intelectual que se necesita para resolver problemas de programación, demanda el uso de estrategias de pensamiento, que permitan la construcción y el manejo de variados modelos, junto con habilidades para identificar, integrar y organizar conocimientos de diversos tipos, que es lo que hemos señalado como el pensamiento de tipo estratégico.

Conforme a lo que hemos expuesto, el desarrollo de estas habilidades está fuertemente determinado y mediado, por la práctica directa en las tareas de resolución de los problemas.

Como lo que interesa es el aprendizaje, el desarrollo y perfeccionamiento de estas habilidades, la práctica debe ser intencionada y de complejidad gradualmente creciente.

Conforme a los detalles que hemos presentado respecto de la actividad intelectual para la resolución de estos problemas, hemos planteado que el aprendizaje de las habilidades para resolver problemas de programación debe orientarse a:

1.- El desarrollo de la capacidad para utilizar conocimientos declarativos y procedimentales de tipo matemáticos,
2.- El aprendizaje de los conocimientos declarativos y procedimentales de la disciplina,
3.- El desarrollo de la capacidad para aplicar y usar los conocimientos declarativos y procedimentales de la disciplina,
4.- El desarrollo de la capacidad para integrar y organizar los conocimientos matemáticos y los de la disciplina,
5.- La adquisición y el desarrollo de habilidades de pensamiento abstracto y la construcción y el manejo de modelos,
6.- La adquisición, desarrollo y perfeccionamiento del conocimiento estratégico, y
7.- El desarrollo de la capacidad de utilizar y manejar el conocimiento estratégico

Conforme argumentamos, nuestra propuesta didáctica para el logro de estos propósitos debe sustentarse en un enfoque de enseñanza y aprendizaje de tipo socio-constructivista, ya que de esta forma no solo favorecemos el acto personal de construcción de conocimientos y el desarrollo de habilidades, sino que además lo potenciamos mediante la interacción social.

SÍNTESIS GENERAL DEL CAPÍTULO 1

La PC es una disciplina que posee una estructura lógico-conceptual particularmente compleja, como consecuencia de los tipos de conocimientos que la conforman.

Su cuerpo de conocimientos evoluciona de manera permanente, con el objeto de ir respondiendo a la diversificación de las demandas que se van produciendo en el ámbito de nuevas necesidades de procesamiento de datos, que es su campo aplicativo.

Los tres elementos que componen su estructura lógico-conceptual son:

1.- El concepto de algoritmo,
2.- El modelo de computador y
3.- El concepto de lenguaje de programación.

El primero es la base para el diseño de las soluciones. El modelo de computador es el sustento para pensar en una máquina que puede ser programada para ejecutar los procesamientos de datos, en tanto el lenguaje de programación constituye la conceptualización de la forma de comunicar los diseños de las soluciones y las acciones de procesamiento de datos al computador, con el objeto de que este pueda ejecutarlas.

La PC es reconocida como un cuerpo disciplinar, que se orienta al estudio sistemático de los procesos algorítmicos que describen y transforman información y sus correspondientes implementaciones como programas computacionales, los que pueden ser efectivamente ejecutados por un computador real. Por tanto, es un cuerpo de conocimientos de orientación tecnológica, cuyos fundamentos se encuentran en la matemática, en la ingeniería, y en la tecnología computacional.

Al aplicar el modelo de Heckhausen (1975), con la intención de clarificar los elementos característicos del cuerpo de conocimientos, es posible establecer que la PC, trata con cuatro objetos: (1) la teoría de la computabilidad, (2) el concepto de algoritmo, (3) los lenguajes de programación, y (4) la metodología de resolución de problemas de procesamiento de datos.

El primero sirve como marco de referencia formal, para establecer el concepto de función calculable, concepto que es fundamental para pensar en el diseño y la representación de procesos de cálculo sobre los datos. El segundo, está fuertemente

relacionado con la teoría de la computabilidad y es la base formal para tratar con el modelamiento de las soluciones a los problemas. Posee dos dimensiones, la formal que tiene que ver con la caracterización de lo que es un algoritmo, y la aplicativa que está relacionada con la representación y expresión de estos los algoritmos.

El lenguaje de programación, es el formalismo que permite disponer de una construcción de carácter sintáctico y semántica, de tipo artificial que sirve para expresar las computaciones, es la herramienta que permite que ellas puedan ser interpretadas y ejecutadas por un computador real.

El cuarto elemento, tienen un carácter central y nuclear para la PC, ya que con la metodología de resolución de problemas, ella se establece el marco de referencia formal y procedimental, de la técnica que se aplica para buscar e implementar las soluciones a los problemas de procesamiento de datos, por tanto es el objeto que actúa como integrador de los demás objetos con que trata la disciplina.

En la dimensión aplicativa, la disciplina de la PC conforma un conjunto de conocimientos cuya orientación fundamental está dirigida el tratamiento de los problemas de procesamiento de datos, específicamente, para tratar con las tareas de diseño e implementación de soluciones computacionales para esta categoría de problemas.

Para Barchini, et. al. (1998), este es un cuerpo de conocimientos de carácter científico tecnológico, y que está compuesto por elementos multidisciplinares que se integran e interrelacionan.

Se configura a partir de un cuerpo de conocimientos que sirve a la búsqueda y diseño de soluciones a los problemas, y otro que sirve para realizar los procesos de implementación de los programas de solución, los que como conjunto se integran en la Metodología de programación de computadores. Para el trabajo en esta dimensión es fundamental el pensamiento abstracto dirigido a la elaboración de distintos modelos.

Cuando la PC se aplica para llevar a cabo tarea de diseñar e implementar soluciones a problemas de procesamiento de datos, tanto los conocimientos, como las capacidades para aplicarlos, se organizan en lo que hemos llamado la estructura lógico-cognitiva, la que resulta de la integración de conocimientos, de tipo declarativos, procedimentales y estratégicos, siendo estos últimos, los más críticos para la ejecución de la tarea intelectual de aplicar los primeros, por tanto esta es un cuestión que debe tenerse en cuenta de manera especial a la hora de diseñar una metodología enseñanza para los cursos de programación.

El conocimiento estratégico, es el conocimiento en el cual se apoya el resolutor para realizar las tareas intelectuales dirigidas a identificar, seleccionar y organizar los conocimientos declarativos y procedimentales que son de utilidad para el diseño de la solución a un problema particular de procesamiento de datos. Es un tipo de conocimiento que además le ha de permitir evaluar la eficacia del propio proceso de resolución y sus resultados, de tal forma de corregirlo cuando estime necesario.

Por tanto se considera que el desarrollo y perfeccionamiento del conocimiento estratégico es un aspecto fundamental sobre el cual deben enfocarse el proceso de enseñanza de la PC, ámbito en el cual, creemos que la práctica directa con actividades que apunten al desarrollo y perfeccionamiento del conocimiento estratégico, pueden favorecer el logro de la capacidad de resolver este tipo de problemas.

En esta perspectiva creemos también, que el constructivismo social puede ser un enfoque para diseñar y desarrollar actividades de enseñanza y aprendizaje para el primer curso de programación, ello basados en nuestra experiencia, según la cual, hemos constatado que cuando creamos condiciones y ambientes, en los cuales conseguimos que nuestros estudiantes se involucren de manera directa e intencionada, en las tareas de resolución de los problemas de programación, se producen mejores oportunidades para que, aprendan tanto los conocimientos de PC, como también, aprendan, desarrollen y perfeccionen las capacidades y habilidades que les permitan utilizarlos de manera efectiva, en los procesos de resolución de estos problemas.

CAPITULO 2

EL APRENDIZAJE DE LA DISICIPLINA DE LA PC

2. EL APRENDIZAJE DE LA DISCIPLINA DE LA PC

INTRODUCCIÓN

Como hemos analizado en el capítulo anterior, la PC posee una estructura lógico-conceptual compleja, lo que se refleja en los tipos de conocimientos que la conforman, los que provienen de distintas disciplinas.

En este mismo capítulo analizamos además, los elementos de la estructura lógico-cognitiva que entra en juego cando los conocimientos de la estructura lógico-conceptual se aplican en la tarea de diseñar las soluciones a los problemas de procesamiento de datos, contexto en el cual señalábamos que para realizar esto, además de los conocimientos, es necesario disponer de la capacidad de construir y tratar con varios modelos de representación, para lo cual se requiere del pensamiento abstracto.

Del análisis de estos elementos, es posible visualizar que el aprendizaje de los conocimientos disciplinares, así como el desarrollo de las capacidades para aplicarlos, y las habilidades de pensamiento abstracto, son cuestiones difíciles para la mayoría de los estudiantes de un primer curso de FP.

Dado este contexto, en el presente capítulo, nos abocamos a analizar el aprendizaje de la disciplina de la PC desde la perspectiva de la propia asignatura de FP, y desde tres ámbitos que nos parecen fundamentales:

1.- Las demandas curriculares, ya que la asignatura debe responder a la definición del tipo de profesional que se quiere formar,
2.- Las exigencias cognitivas, y
3.- La forma en que se desarrolla la docencia en la misma.

Para estos efectos, en primer lugar, analizamos los elementos que caracterizan al área de conocimientos a la cual pertenece esta asignatura, teniendo como referente,

el último informe de recomendaciones curriculares desarrollado por la ACM y la IEEE (CSC2008).

En este informe, junto con presentar los elementos que caracterizan al área de conocimientos de los Fundamentos de Programación, se presentan y justifican los elementos curriculares que se recomiendan tener en cuenta a la hora de diseñar un plan de formación profesional en CcC. En esta misma perspectiva, se entregan poderosos argumentos para respaldar por qué el área de conocimientos de Fundamentos de Programación debe ser considerada como nuclear, central e ineludible para estos planes de formación profesional.

Para complementar la caracterización de la asignatura, analizamos los propósitos que se persiguen con su desarrollo, y los contenidos que en ella se tratan para alcanzar los objetivos de la misma.

A continuación analizamos los elementos que son propios del desarrollo de la asignatura, en términos de su número de créditos y su carga horaria. En este análisis contrastamos estos aspectos para varias universidades nacionales y extranjeras, junto con el caso de la Universidad de Playa Ancha en Chile, que es dónde hemos realizado la intervención. Luego tratamos con la caracterización general de los docentes que desarrollan la asignatura, principalmente en el contexto de nuestro país.

Este capítulo lo cerramos con el análisis de la problemática del aprendizaje de la asignatura en estudio, para lo cual nos apoyamos tanto en las investigaciones que han sido publicadas en torno al tema, como en nuestra propia experiencia docente en el desarrollo de la asignatura.

2.1. El área de conocimientos de FP

Esta área de conocimientos, es una de las catorce áreas disciplinares que, de acuerdo con las recomendaciones curriculares contenidas en el informe CC2001 (CC2001 (2001)), ha de considerarse para la definición de una estructura curricular para planes de formación profesional en CcC.

En este informe, se considera que FP es un área de conocimientos troncal e ineludible para los planes de formación profesional en CcC, requerimiento que ha sido consensuado, por la comunidad internacional de académicos vinculados a estos planes de formación, puesto que se reconoce que la capacidad para programar computadores, constituye una habilidad que debe adquirir y poseer cualquier estudiante de CcC.

La condición de esencial y troncal de esta área de conocimientos, se ha fundamentado y reiterado en todos los informes de las recomendaciones curriculares, que han venido desarrollando y presentando a la comunidad académica, la ACM (Association for Computing Machinery) y el IEEE (Institute for Electrical and Electronic Engineers), instituciones que cuentan con un amplio reconocimiento en el mundo académico.

La primera de estas recomendaciones fue presentada por la ACM en el año 1968 (ACM68), a ellas le siguieron las propuestas por el IEEE en el año 1977 (IEEE77). A partir de estos dos primeros informes, se han desarrollado y presentado varias revisiones y actualizaciones, siendo el último el que se conoce en el ámbito académico como Computing Curricula 2001 (CC2001), que es el resultado de un trabajo en conjunto entre la ACM y el IEEE.

El espíritu de las comisiones que elaboraron los documentos de esta última versión, ha sido llevar a cabo una revisión de las recomendaciones anteriores, y además tomar en consideración los últimos cambios y avances ocurridos en el área de la tecnología de la computación e informática, así como también, prestar atención a las innovaciones que se prevén que podrían producirse en los próximos años, todo ello con el objeto de garantizar que las recomendaciones presentadas, sean un referente adecuado y actualizado, para la definición de los planes de formación profesional en Ingeniería en Computación e Informática.

En virtud de esto, las comisiones que elaboran estos informes han consensuado que para asegurar una adecuada actualización de ellas, a la luz de la velocidad con que avanzan los cambios e innovaciones en esta área de la tecnología, dichas recomendaciones han de ser revisadas y actualizadas a lo menos cada 10 años. Conforme a las revisiones que hemos hecho, no se ha presentado en esta década un nuevo documento de recomendaciones.

Para disponer de los mejores referentes, y de esa forma cumplir de manera cabal con estos propósitos, estas organizaciones mantienen comisiones de trabajo permanentes, constituidas por expertos de prestigiosas universidades de todo el mundo, quienes se encuentran vinculados directamente con el trabajo académico en estos planes de formación profesional.

Las recomendaciones curriculares se han formulado con un carácter general, ya que se espera que cada institución formadora, pueda adaptarlas al momento de implementar su plan de estudios particular, con lo cual se asume que partiendo de estas recomendaciones generales, cada institución formadora defina planes de estudios con matices curriculares diferente, de manera tal de orientar su plan de estudios, a una sub-área de la formación profesional en computación e informática.

A este respecto, en el documento The Guide to Undergraduate Degree Programs in Computing (CC2005), se establece que conforme a las orientaciones de los distintos programas de formación profesional en computación e informática que hoy en día se imparten, estos se clasifican en las siguientes cinco categorías:

1.- Ingeniería en Computación (IC); considera a los programas que tratan con la formación de profesionales que sean capaces de, diseñar e implementar sistemas que impliquen la integración del software y los dispositivos de hardware.
2.- Ciencias de la Computación (CC); trata con los programas cuyo propósito es formar profesionales que a partir de una sólida base teórica, sean capaces de desarrollar software.
3.- Sistemas de Información (SI); referida a los programas orientados a la formación de profesionales que tengan la capacidad para analizar requerimientos de información y procesos de negocios, y que cuenten con la habilidad para especificar y diseñar sistemas acordes, con los objetivos de una organización.
4.- Tecnología de la Información (TI); considera a los programas que persiguen la formación de especialistas capaces de planificar, aplicar, configurar, y mantener la infraestructura computacional de una organización.
5.- Ingeniería de Software (IS); trata con los programas que tienen el propósito de formar profesionales capaces de, diseñar e implementar sistemas de software a gran escala.

Uno de los referentes fundamentales para el diseño de un plan de formación profesional en alguna de estas cinco categorías, es la definición de las catorce áreas de conocimientos contenidas en el informe CC2001. En ellas se establecen los contenidos temáticos, en la idea de que esta definición sirva de guía para la organización de las asignaturas de un determinado plan curricular.

En estos términos, la manera como se aborda una determinada área de conocimientos, la profundidad y énfasis con que sean tratados los contenidos temáticos, la organización de los mismos dentro de las asignaturas, y en general, la forma de organizar y estructurar un determinado plan curricular, se debería encuadrar en alguna de las cinco categorías descritas anteriormente (IC, CC, SI, TI, o IS).

Junto con establecer y definir los contenidos temáticos de cada área, en dicho informe se señalan las razones y los argumentos para establecer cuáles de estos contenidos temáticos han de ser considerados como esenciales e ineludibles, en cualquier plan de formación profesional en Computación e Informática.

Respecto de esto, consideramos que para el contexto de esta investigación, existen dos aspectos centrales de la definición de las catorce áreas de conocimientos, que

son de alta pertinencia y relevancia. El primero es que FP, es una de las catorce áreas de conocimientos que ha sido definida, correspondiendo específicamente a la segunda de ellas, y lo segundo, es que todos los contenidos temáticos que han sido definidos para ésta área, han sido catalogados como esenciales e ineludibles para cualquier plan de formación profesional.

Lo anterior está refrendado en las conclusiones que resultan del estudio presentado en el documento The Guide to Undergraduate Degree Programs in Computing (CC2005), en el cual se analiza el peso que tiene cada una de las catorce áreas de conocimientos, para cada una de las cinco categorías de programas de formación profesional que hemos descrito en los párrafos anteriores. Para llevar a cabo este análisis, se tomaron en cuenta un número importante de programas de formación profesional en distintas universidades del mundo.

Los pesos fueron evaluados en una escala de 0 a 5, sin embargo, asumiendo que pueden existir rangos de variabilidad entre las distintas universidades analizadas, se han registrado los valores mínimos y máximos que se han encontrado. Para a los intereses de la presente investigación, hemos extractado sólo los valores relativos al área de conocimientos de FP, los que presentamos en la siguiente tabla.

IC		CC		SI		TI		IS	
Min.	Máx.	Min.	Máx.	Min.	Máx.	Min.	Máx.	Min.	Máx.
4	4	4	5	2	4	2	4	5	5

Tabla 1: Pesos del área de conocimientos de FP para varias universidades del mundo.
(Extracto de la tabla publicada en CC2005)

A nuestro juicio, los resultados que se muestran, reiteran y confirman la relevancia e importancia que tiene el área disciplinar de FP, en el contexto de cualquiera de las cinco categorías de planes de formación profesional.

Complementariamente, en el mismo documento al cual hacemos mención, se presenta la definición del conjunto de requisitos comunes que debieran alcanzarse, con el desarrollo del área de conocimientos que es de nuestro interés, los cuales se han clasificado en cuatro ámbitos:

1.- Construir una sólida base de conocimientos disciplinares, en lo referido a los conceptos y los fundamentos teórico-formales que dan el sustento científico a la disciplina.
2.- Desarrollar las habilidades para programar computadores, lo que implica comprender el rol central de los algoritmos y las habilidades para comprenderlos, interpretarlos y construirlos.
3.- Desarrollar la capacidad para utilizar y aplicar un lenguaje de programación y un entorno de programación para llevar a cabo las tareas de implementación de algoritmos, y para ejecutar y depurar los programas implementados. Esto

implica además una adecuada compresión del concepto de proceso y su relación con la computación, especialmente con la ejecución de un programa y el funcionamiento del sistema.

4.- Desarrollar habilidades interpersonales y de comunicación con miras a optimizar el desempeño del trabajo en equipos disciplinares e inter-disciplinares.

En atención a la caracterización de esenciales e ineludibles que se ha dado a estos contenidos temáticos, así como también en virtud de la extensión y complejidad de los mismos, es que para cubrirlos de manera adecuada dentro del desarrollo de cualquiera de estos planes de formación profesional, estos comúnmente se organizan en dos o más asignaturas del plan de estudios.

La primera de ellas se encuentra ya sea en el primer o segundo semestre, esto con el propósito de abordar el aprendizaje de estos contenidos, y el desarrollo de las capacidades y habilidades requeridas, en etapas tempranas del plan de estudios, como una forma de asegurar que los conocimientos, capacidades y habilidades estén disponibles al momento de abordar las asignaturas de nivel avanzado e intermedio del plan de formación profesional.

Para confirmar lo anterior, hemos analizado los programas de estudio de once universidades que ofrecen estos planes de formación profesional, en distintas partes del mundo, con la intención en primer término, de verificar en que semestre del plan de estudios se ubica la primera asignatura, y en segundo término, establecer con cuales otras asignaturas de dicho plan se relaciona esta primera asignatura. La información está contenida en las tablas 2 y 3 de los anexos.

Para una mejor visualización y análisis, la información las hemos organizados en las dos tablas. En la primera, se presenta el nombre de la primera asignatura del plan de estudios con que se aborda el área disciplinar, el nombre de la carrera, de la universidad que la dicta y el país al cual pertenece la universidad.

En la segunda tabla se presenta la ubicación de la asignatura dentro del plan de estudios, y los nombres de las distintas asignaturas del mismo plan, con las cuales se relaciona esta primera asignatura, para las once instituciones formadoras analizadas.

Finalmente, con el objeto de reducir el tamaño de la segunda tabla y así hacerla más legible y fácil de analizar, se han agrupado las asignaturas que aunque tienen nombres diferentes, tratan en general con contenidos similares.

Por tanto, FP corresponde a un área de conocimientos que forma parte del ciclo de formación básica, y que es una fuente de conocimientos predominantes en estos

planes de formación, lo se justifica porque aborda el conocimiento de un conjunto de conceptos fundamentales, que son necesarios para el tratamiento y la comprensión de otras áreas de conocimiento de la CcC.

Esto es porque cualquier área de estudio relacionada con esta ciencia, pone al computador, y la posibilidad de que éste sea programado, como un elemento central de ella, así por ejemplo, independiente de las aplicaciones que puedan ser desarrolladas y de las soluciones que se creen a determinados tipos de problemas, siempre está presente la condición de que estas soluciones sean automatizadas mediante un programa computacional.

Conforme a esto, podemos afirmar que la lógica de la estructura de la disciplina, el conjunto de conceptos fundamentales, los procesos de comprensión de la disciplina, y su aplicación en sus distintas áreas de competencia, justifican la existencia de una asignatura que trate con los aspectos formales y las capacidades de base de la disciplina, en etapas tempranas del desarrollo del currículo de estos planes de formación.

Estas razones las podemos sintetizar en las siguientes consideraciones:

- El computador es una máquina de cálculo y que puede ser programada, lo que la convierte en un elemento central para todos los campos disciplinares de la CcC.
- La habilidad para programar un computador es una capacidad fundamental para el abordaje exitoso de muchas asignaturas del plan de formación profesional en CcC.
- La programación de computadores, es una capacidad inherente al tipo de formación profesional; cualquier ingeniero informático debe tener dominio de ella.

Por tanto, la ubicación de una asignatura en los primeros semestres del plan de formación, como organización curricular, responde a una intención curricular según la cual, se quiere privilegiar el desarrollo de los procesos cognitivos específicos para la adquisición de los conocimientos y las competencias que se encuentran implicadas en el desarrollo de la tarea de programar computadores, entendiendo que estos constituyen un núcleo esencial de conocimientos y competencias para estos planes de formación profesional, y que además son necesarios para el abordaje y desarrollo de otros campos disciplinares de la CcC.

Conforme a esto, con el desarrollo de la primera asignatura se punta a la adquisición de conocimientos y el desarrollo de competencias referidas a:

- Comprender y explicar los fundamentos teórico-formales de la PC.
- Aplicar la metodología de la programación a la resolución de problemas de procesamiento de datos.

- Construir e implementar programas computacionales.

2.1.1. Propósitos de la asignatura inicial

Respondiendo a la caracterización del área de conocimientos, el propósito fundamental de una asignatura inicial es abordar el desarrollo de los conocimientos que conforman la estructura lógico-cognitiva que se requiere para PC.

De manera más precisa, podemos afirmar que con ella se intenta dotar a los estudiantes, de los elementos teóricos, metodológicos, procedimentales y estratégicos, que les permitan diseñar e implementar programas computacionales, para resolver problemas de procesamiento de datos, y que estos conocimientos sirvan de base, para el aprendizaje de las demás asignaturas con las se complementa el desarrollo del área disciplinar de FP, así como de otras asignaturas que tienen una relación directa con la programación de computadores.

Deek, Kimmel, and McHugh (1998), señalan que el primer curso de programación debe tratar fundamentalmente con el modelo de resolución de problemas, donde las características del lenguaje de programación, sean introducidas como parte del desarrollo de las soluciones específicas, para lo cual, deben enseñárseles los formalismos, conceptos, métodos y las técnicas, que les habilitan para resolver los problemas, mediante la aplicación de un método riguroso, que parte de la especificación del problema hasta la construcción del programa computacional de solución.

Para nosotros, el desarrollo de la asignatura, se debe enfocar en que los estudiantes consigan un aprendizaje de tipo práctico respecto de la solución de los problemas, ya que aunque estos traten con situaciones simples y donde se requiera de la implementación de unos pocos procesos, sus enunciados deben plantear situaciones reales y no ficticias.

En las recomendaciones curriculares contenidas en el informe CC2001 (CC2001 (2001)), se precisa que el desarrollo de la primera asignatura de FP, debe responder a tres principios básicos.

1.- Introducir a los estudiantes en el conocimiento y la aplicación de los FP desde la perspectiva disciplinar.
2.- Que los estudiantes adquieran una parte importante y sustantiva de las habilidades de programación, en especial, de las que se requieren para el tratamiento de los contenidos de otras asignaturas del plan de formación profesional, y
3.- Que los estudiantes adquieran la base conceptual y metodológica, que les permita abordar el aprendizaje de otros lenguajes y paradigmas de programación en cursos superiores.

Complementariamente, nosotros consideramos que uno de los aspectos más relevante de los propósitos de esta asignatura, es el desarrollo y el perfeccionamiento de las habilidades y estrategias para la resolución de los problemas de procesamiento de datos.

Brooks (1990), señala que el uso de buenas habilidades y estrategias de resolución de problemas, tienen un fuerte impacto en la eficacia y eficiencia del programa final que se produce. Una conclusión importante que se puede obtener de esto es que, la enseñanza de estas habilidades debe hacerse fundamentalmente, mediante la práctica directa e intensiva de la resolución de los problemas, ya que si se basa sólo en la presentación de ejemplos de programas resueltos, no es posible evidenciar ni explicitar el uso de las habilidades y estrategias que han intervenido para alcanzar la solución.

Como lo hemos señalado, la enseñanza de las habilidades y estrategias debe hacerse mediante la práctica activa e intencionada en las tareas de resolución de los problemas.

En esta misma perspectiva Anderson (1976, 1983, 1993), señala que las representaciones abstractas del conocimiento, la construcción de modelos y las estrategias de resolución, no pueden ser aprendidas de manera directa, como es el caso del conocimiento conceptual. Recalca que el aprendizaje de estos elementos solo se consigue *"haciendo"*, con lo cual el autor quiere dar a entender que este aprendizaje, exige la práctica directa con las habilidades y las estrategias.

Para el logro de sus propósitos la asignatura trata con el desarrollo de cuatro áreas temáticas:

1.- Metodología de la programación. Trata con los formalismos y los procedimientos metodológicos de la PC. La metodología da el carácter disciplinar a la programación, haciendo de esta una actividad sistemática, ordenada, y aplicable a una amplia variedad de problema de procesamiento de datos.

    Para estos efectos, su desarrollo se debe focalizar hacia dos aspectos. El primero dirigido a que los estudiantes comprendan el rigor disciplinar con que se debe desarrollar la actividad de la programación, y el segundo, dirigido a la comprensión y aplicación de los procedimientos de la metodología de programación. Sin embargo es importante enfatizar que el tratamiento de estas dos dimensiones ha de hacerse de manera integral, de forma que se comprenda y aplique, el rigor disciplinar a través de las prácticas de aplicación de los procedimientos de la metodología.

2.- Resolución de problemas; Aborda los elementos de orden teórico-formal y procedimentales, relacionados con el diseño de las soluciones a los problemas

de procesamiento de datos, enfatizando por sobre todo, los aspectos aplicativos de los conceptos y los procedimientos. También trata con el desarrollo y el uso de estrategias de pensamiento abstracto y de elaboración de modelos.

El principal propósito aquí, es lograr aprendizaje, manejo y aplicación de las estrategias. Como hemos señalado, la habilidad de manejar y aplicar estas estrategias es absolutamente esencial para la elaboración de los modelos de representación que requiere el diseño de las soluciones. Por tanto, al desarrollar esta área temática, se debe poner énfasis en la práctica aplicativa de los conocimientos, como una cuestión fundamental para lograr la adquisición, el desarrollo y el perfeccionamiento de los procesos y las estrategias de resolución de estos problemas.

3.- Algoritmos y formas de representación algorítmica. El desarrollo de esta área se orienta al logro de la capacidad para utilizar el lenguaje algorítmico.

Dado el carácter aplicativo de esta área temática, es esencial que su desarrollo se ponga énfasis en los aspectos prácticos de utilización de las representaciones algorítmicas y del lenguaje para las interpretaciones de las soluciones.

4.- Lenguaje de programación. Esta área aborda el desarrollo de las capacidades para utilizar un lenguaje de programación para implementar los algoritmos de solución. Nuevamente aquí es importante enfatizar los aspectos aplicativos de los conocimientos del área. Es importante también, asegurar la comprensión de la relación que existe entre la representación algorítmica de una solución y su correspondiente implementación como programa computacional.

Aunque estas cuatro áreas temáticas provienen de fuentes de conocimientos distintas, en el tratamiento y uso de ellas, se debe enfatizar tanto sus relaciones, como su integración, con el objeto de que se reconozca y comprenda tanto la importancia de cada área, como el carácter integrador y disciplinar con que deben utilizarse los conocimientos, cuando se aplican a la resolución de los problemas de procesamiento de datos, con forma de asegurar que con esta primera asignatura, se aprendan de manera adecuada los fundamentos de la disciplina.

Respecto de esto, en el informe de las recomendaciones curricular CC2001 (CC2001 (2001)), se hace especial mención a que en la enseñanza de los FP, debe enfatizarse el carácter disciplinar y metodológico de la PC.

Como una forma de garantizar la integración de las cuatro áreas de conocimientos, se recomienda organizar la asignatura en dos momentos; el primero enfocado al tratamiento de los temas introductorios de la programación, como sus fundamentos teórico-formales, y el segundo enfocado al uso y aplicación de los FP en el diseño e implementación de las soluciones.

Desde nuestra perspectiva, en la fase inicial del desarrollo de este segundo momento, se debe prestar especial atención al uso y aplicación del lenguaje algorítmico y al diseño de ellos, intentando que reconozca que el desarrollo riguroso de estas fases es la base fundamental para elaborar las soluciones.

Luego se debe introducir el concepto de lenguaje de programación, abordando el problema de la correspondencia entre la notación algorítmica y el lenguaje. Junto con ello, se debe tratar el manejo de un entorno de programación, y trabajar en el proceso de codificar los algoritmos de solución que han sido desarrollados.

Con el tratamiento de los FP se debe perseguir el propósito de tratar el conjunto de conceptos que conforman la base teórica de la disciplina, trabajando por sobre todo, en la comprensión de esta ella y su importancia para dar rigor a la disciplina de la PC.

Este conjunto de conceptos deben desarrollarse en estrecha relación con la resolución de algunos problemas específicos, según los cuales se haga evidente la aplicación de ellos.

Y reiteramos que en el desarrollo de la asignatura, se debe prestar especial atención a la enseñanza y el aprendizaje de las habilidades y las estrategias de pensamiento para resolución de estos problemas, para lo cual se han de utilizar los problemas como un medio para transmitir y aprender el uso y aplicación de las habilidades y estrategias de pensamiento y de resolución.

2.1.2. Los contenidos de la asignatura de FP.

Los contenidos y conocimientos que han de ser enseñados, para lograr el desarrollo de la competencia general para la resolución de problemas de procesamiento de datos, los podemos agrupar en las siguientes seis categorías:

A.- La resolución de problemas de procesamiento de datos y la PC; aborda la relación entre el proceso mismo de resolución de problemas y la posibilidad de automatizar las soluciones, mediante un programa computacional. Se estudia la definición de lo que es un problema de procesamiento de datos, el concepto de máquina de Turing, las teorías de la computabilidad y de la computación, y el concepto de computador como máquina de cálculo que puede ser programada.

Son conocimientos esenciales para entender la relación conceptual entre, el proceso mismo de resolución de los problemas, y la programación de computadores.

B.- El concepto de algoritmo y el lenguaje algorítmico; trata con el concepto y la caracterización de algoritmo como recurso para, describir las acciones de un proceso, y con las formas del lenguaje algorítmico para expresar procesos de

computación. Se estudian los aspectos formales y aplicativos de los algoritmos, como recurso para expresar y analizar conjuntos de acciones de procesamiento de datos.

Estos conocimientos son la base para entender el concepto de algoritmo como fundamento para la definición de conjuntos de acciones y operaciones para el procesamiento de datos, es decir las computaciones. Los conocimientos del lenguaje algorítmico tratan con los formalismos para expresar los procesamientos de datos o computaciones, y para analizarlas.

C.- La Metodología de resolución de problemas de procesamiento de datos; trata con la definición de la metodología, como técnica y procedimiento integral y sistemático, para realizar el proceso completo de resolución de problemas. Se estudian los fundamentos teóricos de la metodología, los propósitos de cada etapa del proceso de resolución, y la relación de dependencia entre cada una y los procedimientos para aplicarlas a la resolución de los problemas.

Estos conocimientos son el fundamento para entender el proceso de resolución de los problemas, sus etapas, la integridad y sistematicidad del mismo. Se estudian los fundamentos de cada etapa, y los procedimientos para realizarlas cuando se trata con la resolución de los problemas.

D.- El lenguaje de programación; trata con el concepto de lenguaje de programación, y con los detalles de la escritura de un algoritmo como programa computacional.

Se estudia el concepto de lenguaje de programación y su caracterización, y a partir de un lenguaje particular, se estudian sus formas declarativas y los aspectos aplicativos de la implementación de un algoritmo como programa computacional y su ejecución.

Estos son los conocimientos que se requieren para se trata con la fase final de la resolución de problemas de programación, que es la implementación del programa computacional.

Como lo hemos reiterado en varios de nuestros análisis, una característica fundamental del manejo de estos conocimientos, es su componente práctica. Estamos hablando de conocimientos que conforme a su dominio, han de permitirle al resolutor ser capaz de desarrollar el conjunto de actividades que implica la resolución de problemas de procesamiento de datos, y que como analizaremos más adelante, es precisamente esta habilidad práctica del dominio de los conocimientos, lo que conlleva las mayores dificultades para nuestros estudiantes.

Así, la primera asignatura con que se aborda el área de conocimientos de FP, se debe orientar al desarrollo del conjunto de saberes y capacidades, que están determinados por las estructuras de conocimientos a las que hemos hecho mención.

Desde una perspectiva más específica, son los conocimientos que se encuentran involucrados con los aspectos teórico-formales, en que se sustenta la metodología de resolución de problemas de procesamiento de datos, y con los elementos de carácter práctico, referidos al desarrollo de capacidades para aplicar la metodología al análisis y diseño de soluciones de problemas de esta categoría, así como para la adquisición de las competencias y capacidades que son requeridas para, aplicar los recursos de un lenguaje de programación de computadores, para la implementación de los programas computacionales mediante los cuales se automatizan las soluciones diseñadas.

Por tanto, esta primera asignatura constituye el instrumento curricular, mediante el cual se debe lograr el aprendizaje y el desarrollo de los elementos que conforman la estructura lógico-cognitiva, y el desarrollo de las capacidades, habilidades y destrezas que son requeridas para la PC en una etapa inicial.

Con el objeto de precisar con el mayor detalle posible, el manejo de conocimientos que se intentan alcanzar con el desarrollo de estos contenidos, en la tabla 95 del anexo presentamos un desglose de ellos en 12 ámbitos de conocimientos. En la primera columna de la izquierda, detallamos los tipos de conocimientos para cada uno de los 12 ámbitos, y en la columna de la derecha, detallamos los contenidos que se encuentran implicados en cada una de los ámbitos.

Como planteamos al comienzo de este apartado, el área de conocimientos de FP es considerada como troncal y nuclear, ya que el conjunto de contenidos que se tratan y las capacidades que en ella se desarrollan, constituyen conocimientos que son fundamentales no solo para el aprendizaje de la PC, como capacidad que cualquier profesional de CcC debe poseer, sino que además, por la relevancia que estos tienen, como cocimientos previos que se requieren para tratar con los contenidos y aprendizajes, de varias asignaturas de nivel intermedio y avanzado que se encuentran en el currículo de formación de este tipo de profesionales.

Con el propósito de clarificar esto, en la tabla 95 del anexo, se muestran en una primera columna las asignaturas de nivel intermedio y avanzado para las cuales se requieren los conocimientos de FP. En la primera columna de la izquierda se encuentra el nombre genérico de la asignatura, en la segunda columna, se muestra una breve descripción de cada una de ellas, en tanto en la tercera columna, se detallan los conocimientos previos que se requieren para el tratamiento de cada una de ellas. Al revisarlos, se aprecia la fuerte relación de ellos con los conocimientos indicados en la tabla 95.

Para la confección de esta tabla, se han considerado las asignaturas que se encuentran en plan de formación profesional de la carrera de Ingeniería informática de la Universidad de Playa Ancha, que es donde hemos realizado esta investigación.

En síntesis, FP es un área de conocimientos que es troncal y nuclear para cualquier plan de formación profesional en CcC, y corresponde a uno de los catorce que se señalan en las recomendaciones curriculares contenidas en el informe CC2001.

El propósito esencial de una primera asignatura de programación, es abordar el aprendizaje de los conocimientos que se requieren para resolver e implementar soluciones computacionales a problemas de procesamiento de datos, con lo cual se debe intentar que los estudiantes aprendan los elementos teóricos, metodológicos, procedimentales y estratégicos que les permitan diseñar e implementar soluciones para los problemas.

Para el logro de estos propósitos, en una primera asignatura se debe tratar con el desarrollo de cuatro áreas temáticas que son: Metodología de la programación, Resolución de problemas, Algoritmos y sus formas de representación, y Lenguaje de programación.

Un aspecto importante del tratamiento de los contenidos, es que debe reflejarse un tratamiento integrado de estas cuatro áreas temáticas, clarificando y explicitando el aporte que cada área tiene en la propia tarea de resolución de los problemas, y destacando el rigor disciplinar con que se debe realizar el proceso de resolución de los problemas.

Finalmente, con el desarrollo de una primera asignatura se debe tratar de desarrollar los conocimientos que se detallan en la tabla 95 del anexo.

2.2. Variables que comprometen el desarrollo de la asignatura de FP

2.2.1. Distribución de los tiempos en el desarrollo de la asignatura.

Para tener un referente para el análisis de la distribución de los tiempos, en la siguiente tabla comparamos cuatro aspectos de ella, entre lo que ocurre para el caso de la carrera de Ingeniería en informática de la Universidad de Playa Ancha (UPLA), que es donde hemos realizado esta investigación, y otras instituciones que hemos revisado.

Descriptores	UPLA	Otras Instituciones
N° de Sesiones de clases semanales	4,5 horas	4,5 a 6,0 horas
N° de Semanas del semestre	17 semanas	17 a 20 semanas
Proporción horas teóricas/horas de práctica de laboratorio	2/3 dedicadas al tratamiento de contenidos teóricos 1/3 dedicado a la practica en laboratorio	2/3 dedicadas al tratamiento de contenidos teóricos 1/3 dedicado a la practica en laboratorio
N° de Sesiones de ayudantía por semana	1 sesión por semana, sin exigencia de asistencia y sin evaluaciones	1 a 2 sesiones por semana, en algunos casos obligatorias y con evaluaciones

Tabla 2: Elementos generales de desarrollo de la asignatura de FP

De la tabla se observa que el número de horas presenciales que se consideran para el desarrollo de la primera asignatura, tanto en el contexto de las universidades nacionales como extranjeras que hemos revisado, fluctúan entre 4,5 y 6 horas de clases preséncíales a la semana, durante un semestre académico, con una duración de entre 17 y 20 semanas. Para el caso de la asignatura en estudio (UPLA), se tienen 4,5 horas semanales y 17 semanas de clases lectivas.

Como se observa en la misma tabla, tanto para el caso de UPLA, como para las otras instituciones, generalmente este número de horas se distribuye conforme a una proporcionalidad de 2/3 del tiempo dedicado al tratamiento de contenidos teóricos y sesiones de ejercicios en una sala de clases, y un tercio del tiempo dedicado a sesiones de práctica en un laboratorio de computación.

Además de las sesiones de clases, en la mayoría de los planes de clases de esta primera asignatura que hemos revisado, tanto en instituciones nacionales como extranjeras, se verifica la existencia de a lo menos 1 sesión semanal de ayudantía, ofrecida a los estudiantes como una instancia más de aprendizaje y practica en la aplicación de contenidos. Como se observa en el caso de Ia UPLA se considera una sesión de ayudantía sin exigencia de asistencia.

Para el caso de Ia UPLA, consideramos que las sesiones ayudantía tienen un nulo impacto sobre las mejoras en el aprendizaje de los estudiantes. Creemos que las razones que determinan este nulo impacto son entre otras, la no existencia de planificación de las mismas, la participación en ellas por parte de los estudiantes es muy baja ya que no es calificada y no se exige asistencia a las mismas, sólo se observa una mayor asistencia cuando se aproxima una prueba, y en las actividades de ayudantía no se promueve el aprendizaje y desarrollo de las capacidades que hemos señalado como fundamentales para un curso de este tipo, ya que el ayudante sólo se dedica a presentar ejercicios resueltos por él mismo.

Conforme a la información que hemos recopilado, en el caso de las universidades de nuestro país, estas sesiones de ayudantía son conducidas generalmente por estudiantes de cursos superiores de la misma carrera, o carreras afines, los que son seleccionados fundamentalmente en base a la calificación que el estudiante haya obtenido al aprobar el curso de FP. Para el caso de instituciones extranjeras, no disponemos de información respecto de si las sesiones de ayudantía son conducidas por docentes o por estudiantes.

En teoría, se supone que la programación de las sesiones de ayudantía se debería realizar de manera conjunta entre el profesor de la asignatura y el ayudante, instancia en la cual se establece la secuencia de contenidos, los tipos de ejercicios que se

desarrollaran, la metodología según la cual se conducirán las clases de ejercicios, etc. A este respecto, y conforme a los antecedentes que manejamos, en la práctica esta programación rara vez se realiza de manera conjunta, con lo cual los ayudantes tienen autonomía para organizar las sesiones de ayudantía.

Como ya indicamos, esta situación tiene efectos negativos en términos del aprovechamiento que debieran hacer los estudiantes, de esta instancia de aprendizaje, entre ellos está la frecuente improvisación de los ayudantes a la hora de decidir los contenidos y actividades a tratar en las sesiones de ayudantía, la baja asistencia de los estudiantes, el discutible efecto que estas sesiones pueden tener para el aprendizaje de los estudiantes ya que no se conocen estudios serios sobre la efectividad de las ayudantía.

Sin embargo, en algunos casos, la participación de los estudiantes en estas sesiones es calificada con una ponderación que no supera el 10% de la nota final de la asignatura. A este respecto es necesario destacar que para el caso de las 9 universidades de nuestro país, de las cuales disponemos de información, en 5 de ellas la actividad de ayudantía tienen calificación para los estudiantes.

Para el caso de las restantes 4 universidades, el reglamento de evaluación no permite que los ayudantes apliquen evaluaciones, porque se considera que no tienen la calidad de responsables del curso, situación que determina que el interés de los estudiantes por tomar parte en las sesiones de ayudantía sea baja.

Para el caso particular de la UPLA, de los tres períodos de clases a la semana (4,5 horas), dos de ellos (3,0 horas) se desarrollan en una sala de clases y son dedicados a sesiones de clases expositivas y de ejercicios.

En las sesiones de clases expositivas, se presentan y analizan los elementos teórico-formales tanto de la metodología de resolución de problemas de procesamiento de datos, como los referidos al lenguaje de programación usado para la implementación de las soluciones. Durante las sesiones de clases de ejercicios, se presentan a los estudiantes ejercicios tipo, sobre los cuales se presentan y trabajan los aspectos procedimentales, relacionados con la aplicación práctica de los elementos teórico-formales de la metodología y el lenguaje de programación. Conforme a la programación de estas sesiones de clases expositivas y de ejercicios, estas se desarrollan en días diferentes de la semana.

El tercer período (1,5 horas), se desarrolla en el laboratorio de computadores, y tiene como propósito tratar con los aspectos práctico relacionados con el manejo del entorno de programación y del lenguaje de programación para la implementación de las soluciones. Con esto se persigue un primer objetivo que es el de verificar y probar los programas que se han implementado a partir del diseño de las soluciones a los

problemas, y como segundo objetivo, adquirir el dominio en el manejo de las sentencias y recursos para la implementación de programas con que cuenta el lenguaje de programación. Conforme a esto, estas sesiones de laboratorio se desarrollan bajo el esquema de taller.

Como señalamos, estos 3 períodos semanales se complementan con una sesión de un período semanal de ayudantía (1,5 horas), la cual es conducida por un estudiante que haya aprobado previamente la asignatura con una calificación no inferior a 5,0 (en la escala de 1,0 a 7,0 y con una nota mínima de aprobación de 4,0).

Esta sesión de ayudantía se desarrolla en el laboratorio de computación, y en ella el ayudante presenta problemas, desarrolla las soluciones e implementa los programas respectivos, al tiempo que atiende preguntas y consultas de los estudiantes.

Para estas sesiones no se consideran evaluaciones, ya que el reglamento de evaluación de la universidad no permite que los ayudantes se involucren en el proceso de evaluación, lo que produce una baja tasa de asistencia de los estudiantes a estas sesiones.

Sin embargo, a pesar de la baja tasa de asistencia a las sesiones de ayudantía, la experiencia muestra que regularmente en la sesión inmediatamente anterior a una evaluación, se da un incremento en la asistencia de los estudiantes.

En estas sesiones, los estudiantes actúan fundamentalmente de manera pasiva, y en la mayoría de los casos, sólo se dedican a copiar los ejercicios resueltos por el ayudante. En raras ocasiones adoptan un rol más activo, haciendo preguntas respecto de una parte o todo el proceso de resolución, y en casos muy excepcionales, hacen propuestas al ayudante, respecto de cómo resolver alguno de los problemas planteados, y/o muestran o discuten en la ayudantía, una solución que ellos hayan realizado en su estudio personal previo a la sesión.

2.2.2. Caracterización general de los estudiantes que cursan la primera asignatura de FP.

De la relación que se tiene con los estudiantes y de los diálogos informales que se ha tenido con ellos, se ha recogido la opinión de que en su gran mayoría, consideran que el aprendizaje de la programación de computadores es una tarea compleja, y la mayoría de los alumnos que han cursado varias asignaturas de programación señalan que algunas de ellas han debido cursarlas dos y hasta tres veces, y no son pocos los casos en que estudiantes han debido abandonar la carrera debido a sus fracasos reiterados en una o más asignaturas de la línea de programación, lo cual determina un alto grado de predisposición al fracaso.

Desde otra perspectiva, Byrney & Lyons (2001), afirman que una característica común de los estudiantes de un primer curso de programación, es que carecen de los conocimientos y las habilidades de programación de los expertos.

Por su parte, Robins et. al. (2003), señalan que los programadores novicios poseen un conocimiento superficial respecto de la resolución de problemas de programación y que es un conocimiento organizado en base a similitudes también superficiales, lo que se manifiesta en un enfoque de la programación, conforme a una visión del código del tipo línea a línea, y no un enfoque orientado a la visualización de la estructura completa del programa.

Estos mismos autores han observado que los estudiantes en general, dedican muy poco tiempo a los procesos de análisis, planeación de la solución, así como también, a las tareas de prueba del programa, y cuando requieren realizar cambios en el código del programa, tratan con modificaciones pequeñas y localizadas, sin tratar con una visión completa del programa al desarrollar las modificaciones, lo que generalmente se traduce en que el programa modificado tenga más errores que en su estado inicial.

Los autores Kessler & Anderson (1989), han constatado que cuando los programadores novicio trabajan en la resolución de un problema, hacen uso de los conocimientos en un ámbito muy específico, con lo cual no atienden a las características de generalidad que requieren las soluciones a estos problemas. También han observado que los novatos, fallan al aplicar el conocimiento que poseen, lo que hace que el promedio de los estudiantes, no logren avances significativos en el desarrollo del conocimiento de programación en los cursos introductorios, lo que también ha sido observado por McCracken et. al. (2001), quienes dan cuenta de que los estudiantes muestran serias deficiencias en las habilidades de programación en estos cursos.

Con el objeto de diagnosticar cuales son las dificultades que tienen durante el aprendizaje en un primer curso de programación, Milne and Rowe (2002) han aplicado un cuestionario a los estudiantes, y en una de sus conclusiones relevantes señalan que los propios estudiantes no saben reconocer sus dificultades.

En una perspectiva diferente de análisis de las actuaciones de los estudiantes en los primeros cursos de programación, Perkins et. al. (1989), indican que éstos tienen diferentes comportamientos cuando son enfrentados a las situaciones problemas, caracterizándolos como “stoppers” y “movers”.

Estos autores precisan que los “stoppers”, son aquellos que cuando se enfrentan a la resolución de un problema, y les surge la primera dificultad, simplemente se detienen y abandonan toda intención de resolver el problema por si solos, en tanto los “movers”, actúan de manera persistente en sus esfuerzos por resolverlo, intentan modifican sus programas, y generalmente tratan de retroalimentarse de sus errores mediante análisis de los mismos.

Estos mismos autores precisan que también existen los “extreme overs”, a los que denominan también como los “tinkerers”, que los definen como aquellos estudiantes que tienen dificultades para hacer seguimientos o ruteo a sus programas, por lo que cuando encuentran errores e ellos, proceden a hacer cambios al azar, con lo cual no resolverán los errores, y por tanto, al igual que los “stoppers” no consiguen avanzar en la tarea de resolver efectivamente el problema.

Como en cualquier aprendizaje de una capacidad de carácter compleja como es la programación, hay estudiantes que son efectivos y otros que no lo son (Robins et. al. (2003)). Lo que significa que, existen estudiantes para quienes aprender programación no representa muchos esfuerzos, y aquellos que no aprenden, si no se les brinda una atención especial y personal. En este contexto, estos autores afirman que, las estrategias personales de aprendizaje y la motivación, afectan de manera importante el éxito en este tipo de aprendizaje.

Así también, estos mismos autores afirman que, tanto los conocimientos, como las experiencias previas que poseen los estudiantes, pueden ser factores importantes que hacen que los ellos cometan errores durante los procesos de resolución de los problemas, especialmente cuando intentan traducir cada paso de una secuencia de acciones escritas en un lenguaje natural, a un programa. Robins (Robins et. al. (2003)) y sus colaboradores, creen que la dificultad está en que los estudiantes no son capaces de trata con las diferencias entre el lenguaje natural y un lenguaje de programación.

Estos mismos autores, en un contexto más específico han observado varios tipos de errores en los programadores novicios, dentro de los que destacan que, en general

no poseen modelos mentales detallados, fallan en la aplicación de los conocimientos que son relevantes para la solución de un problema, usan estrategias generales de resolución de los problemas y no estrategias específicas o estrategias de programación, y utilizan un enfoque de programación línea por línea y no un enfoque basado en la significancia de las estructuras de un programa.

Davies (1993) por su parte señala que, el conocimiento de los estudiantes puede adolecer de varias fallas que hacen que sea un conocimiento frágil, ya sea porque es un conocimiento que ha sido olvidado, o porque es inerte en el sentido de que, es un conocimiento que si bien han aprendido, pero no tienen la capacidad para usarlo, o que lo usan de manera inapropiada.

A esto nosotros agregamos que los estudiantes poseen estrategias de pensamiento frágiles, en el sentido de que son estrategias muy básicas, que no permiten trabajar apropiadamente con las abstracciones, sumado a ello la poca habilidad para trabajar con modelos, que les permitan el manejo de las abstracciones, y también, un manejo deficitario de estrategias cognitivas de orden superior, como por ejemplo la capacidad para establecer relaciones entre distintos conocimientos y para transferirlos a situaciones nuevas, y por añadidura, un deficiente o nulo manejo de estrategias de control y regulación sobre lo que se aprende, y de cómo aplicar lo aprendido.

Conforme a nuestra experiencia, los elementos que se señalan en los párrafos anteriores, constituyen las características que describen el comportamiento general de los estudiantes de programación. Sin embargo, para el caso de nuestra Universidad, debemos mencionar además que la mayoría de los estudiantes se caracterizan por presentar, una actitud pasiva tanto durante las clases expositivas, como en las clases de ejercicios.

Se dedican fundamentalmente a tomar apuntes, siendo renuentes a realizar aportes, mostrar sus soluciones, discutir sobre ellas, y muestran una nula participación en los análisis y discusiones colectivas que intenta llevar a cabo el profesor durante la resolución de los problemas.

Se observa también que ellos realizan un estudio de los contenidos y del desarrollo de los ejercicios principalmente de manera individual. Por otra parte, tienen una baja asistencia a las ayudantías y utilizan de manera escasa los horarios de consultas y tutorías ofrecidos por el docente.

En general, son estudiantes que enfrentan sus momentos de estudio personal, trabajando sobre la reproducción de los ejemplos y ejercicios desarrollados en clases, en la idea de retener en sus mentes los pasos de resolución de cada ejercicio, mediante la memorización de los mismos, y en la esperanza de que los mismos ejercicios, o ejercicios muy similares, sean preguntados en la evaluaciones.

El desánimo y la sensación de fracaso, es otro de los aspectos que caracteriza a estos estudiantes, situación que hace que un número significativo de ellos, abandonen la asignatura antes de finalizar el semestre, empujados principalmente por la sensación de que consiguen pocos avances en sus aprendizaje. En esta perspectiva, muchos de ellos manifiestan que no saben cómo abordar el aprendizaje de la asignatura.

En contraposición, aquellos estudiantes que van obteniendo buenos resultados académicos, evitan el trabajo con los estudiantes que tienen menos éxito en el aprendizaje, ya que consideran que estos les entorpecen sus avances.

En la siguiente imagen, presentamos una síntesis de las características generales de los estudiantes que hemos estado señalando en este apartado.

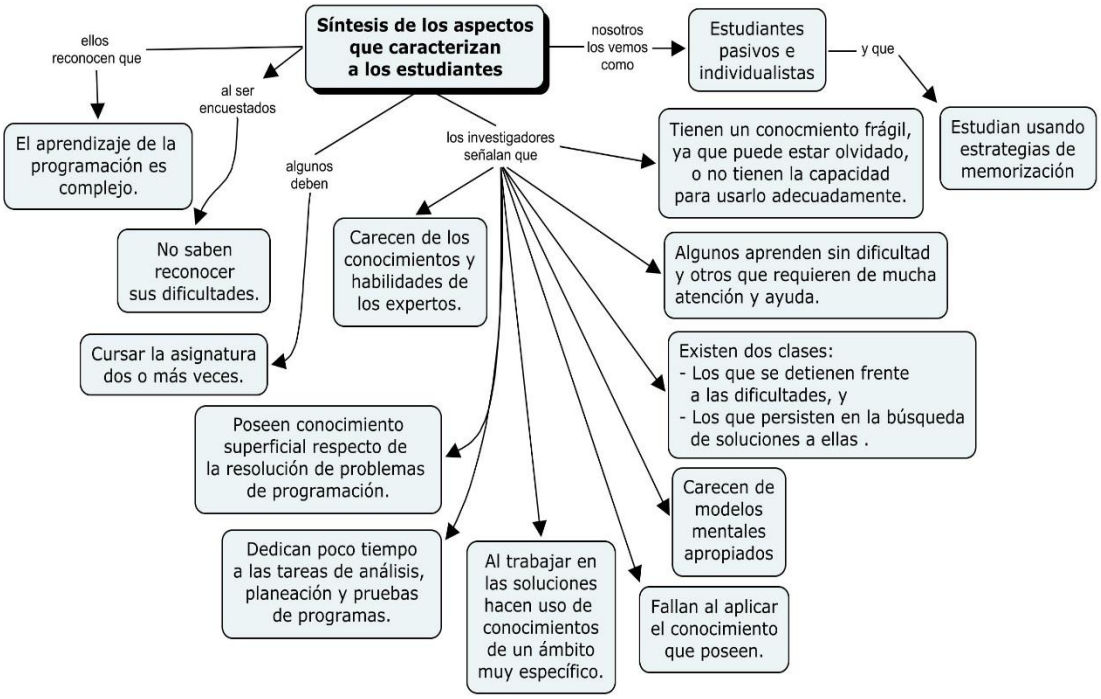

Figura 12: Aspectos que caracterizan a los estudiantes (Caneo, 2009)

2.2.3. Características generales de los docentes que dictan la primera asignatura de FP

De acuerdo al conocimiento general que se tiene respecto de los docentes que dictan la primera asignatura de FP, en las instituciones de educación superior de nuestro país, estos son generalmente profesionales con formación de ingenieros o técnicos del área de la computación o de áreas técnicas de cercanía disciplinar como la electrónica o la electricidad.

Si consideramos que los primeros planes de formación en CcC se iniciaron en Chile en los años 80, hoy en día existen docentes que llevan dictado esta asignatura por más de 30 años.

Estos profesores, tanto los más antiguos, como aquellos que han comenzado a dictar la asignatura en los últimos años, reconocen que la metodología de enseñanza que utilizan en sus prácticas docentes, son las mismas que utilizaron con ellos cuando fueron estudiantes, esto es, la típica clase magistral o de cátedra, conducida de forma frontal y expositiva, dejando pocos o nulos espacios para el análisis, la reflexión y la discusión de los contenidos presentados, y con escazas o nulas posibilidades de que se dieran situaciones de aprendizaje, para abordar los aspectos aplicativos de lo que se aprende.

Bajo esta forma de instrucción, los estudiantes eran meros receptores de los contenidos que se les exponían.

En la actualidad, esta forma de conducción de la asignatura no ha cambiado significativamente, incluso se dan muchos casos, en que el docente responsable de la cátedra, sólo realiza las clases expositivas en las que se abordan los contenidos de carácter teórico-formales, y uno o más estudiantes, son los responsables de realizar las clases de ejercicios, en las cuales se abordaban los aspectos de orden práctico aplicativo de los conocimientos.

Aunque de manera no sistemática, algunos de los actuales docentes han desarrollado apuntes y guías de ejercicios como material complementario a sus clases expositivas.

Generalmente, entre ellos existe la creencia de que al desarrollar estilos de enseñanza que promuevan una participación más activa de sus estudiantes, les significarán pérdidas importante de tiempo en el desarrollo de las sesiones de clases, lo que pondrá en serio riesgo el cumplimiento del programa de contenidos de la asignatura.

Algunos de estos docentes están convencidos, que es sólo responsabilidad de los estudiantes, realizar las acciones que sean necesarias para asegurar el aprendizaje de los contenidos, las capacidades y habilidades de programación, es decir, son ellos los que deben asistir a clases, utilizar los apuntes y resolver los problemas que se les entregan en las guías de ejercicios, resolver los ejercicios de pruebas anteriores, así como asistir a las ayudantía y utilizar esta instancia para aclarar las dudas con los ayudantes.

Sin embargo, existe un número menor de docentes que manifiestan su preocupación, por desarrollar acciones tendientes a disminuir las altas tasas de reprobación de

estudiantes que tiene la asignatura, sin embargo, reconocen que al no poseer una formación pedagógica, carecen del manejo de técnicas o métodos que al ser aplicados de manera sistemática y organizada, puedan permitirles mejorar los índices de aprobación.

En este sentido, estos docentes manifiestan que es urgente conocer experiencias serias de mejora en estos procesos de enseñanza y aprendizaje, y capacitarse de manera adecuada para desarrollar y poner en práctica iniciativas que permitan mejorarlos.

De lo que hemos presentado en este apartado se puede resumir que generalmente para la primera asignatura del área de conocimientos de FP, se considera una carga horaria que varía entre 4,5 y 6,0 horas semanales para cada semestre académico que tiene una duración de entre 17 a 20 semanas.

De este tiempo, los 2/3 se dedican al tratamiento de los contenidos teóricos en la sala de clases, y 1/3 del tiempo, se dedica al trabajo en tareas prácticas de resolución de problemas en el laboratorio de computadores.

Además, puede existir hasta dos sesiones de ayudantía semanales, las cuales en algunos casos son de carácter obligatorio para los estudiantes, y en otros casos, son sin obligación de asistencia.

Para el caso de la Universidad de Playa Ancha, que es donde hemos llevado a cabo esta investigación, la asignatura cuenta con 4,5 horas semanales, de las cuales, 3 se dedican al trabajo en la sala de clases y 1,5 al trabajo en un laboratorio de computadores. Complementariamente, existe una sesión de 1,5 horas para ayudantía, que no es de asistencia obligatoria y que tampoco es calificada. El semestre académico tiene una duración de 17 semanas.

Hemos argumentado también que la generalidad de los estudiantes, tiene la percepción de que aprender programación de computadores es un tarea altamente compleja, y no tienen claridad respecto de cuál o cuáles, deberían ser las estrategias de estudio que les podrían ayudar a superar la dificultades propias del aprendizaje en la asignatura.

Para el caso de las universidades de Playa Ancha, Valparaíso, y Federico Santa María, los resultados históricos de rendimiento muestran, una tendencia general en el sentido de que un número significativo de estudiantes, aprueban la primera asignatura, después de haberla cursado dos o más veces, y existe un número no menor de ellos, que han debido abandonar la carrera por no haber conseguido aprobarla.

Respecto de las dificultades que presentan los estudiantes con esta asignatura, los investigadores han reportado las siguientes:

a) Los estudiantes de un primer curso de programación, carecen de los conocimientos y las habilidades de programación de los expertos. (Byrney & Lyons (2001)),
b) Una característica general de los programadores novicios es que poseen un conocimiento respecto de la programación que es superficial, y que está organizado en base a similitudes superficiales, por lo al abordar la resolución de los problemas, usan un enfoque de programación línea por línea y no un enfoque orientado a la estructura del programa. (Robins et. al. (2003)),
c) Dedican tiempo mínimos a los procesos de análisis, planeación de soluciones y a las tareas de prueba de programa, y cuando le realizan cambios, tratan con modificaciones pequeñas y localizadas, sin trabajar con una visión completa del programa, lo que generalmente se traduce en que la su modificación, tenga más errores que en su estado inicial. (Robins et. al. (2003)),
d) Los programadores nóveles, al trabajar en la resolución de un problema, hacen uso de conocimientos en un ámbito muy específico, sin atender a las características de generalidad que requieren las soluciones a estos problemas, y fallan al aplicar correctamente el conocimiento que poseen. (Kessler & Anderson (1989), McCracken et. al. (2001)),
e) Los propios estudiantes no saben reconocer cuales son las dificultades que tienen para aprender programación. (Milne and Rowe (2002)),
f) Existen estudiantes que son persistentes en sus intentos por resolver los problemas, tratando de resolver las dificultades que tienen en el aprendizaje, en cambio hay otros que abandonan sus intentos cuando se encuentra con las primeras dificultades. (Perkins et. al. (1989)),
g) Existen estudiantes que aprenden programación si muchos esfuerzos, y otros que no lo harán si no se les brinda una atención personal. Además, las estrategias de aprendizaje personales y la motivación afectan el éxito del aprendizaje. (Robins et. al. (2003)),
h) Los programadores novatos, no tienen modelos mentales detallados, y fallan en la aplicación de los conocimientos que son relevantes para la solución de un problema. Además usan estrategias generales de resolución de los problemas y no estrategias específicas o estrategias de programación. (Robins et. al. (2003)),
i) En general novicios, disponen de conocimientos débiles, incompletos, y/o tienen poca capacidad para usarlos, y no poseen y/o manejan las estrategias de pensamiento que se requieren para resolver problemas programación. (Davies (1993))

Conforme a los antecedentes que tenemos para el caso de nuestro país, la generalidad de los docentes que dictan esta primera asignatura, son profesionales con formación de ingenieros o técnicos del área de la computación o de áreas técnicas afines, y varios de ellos cuentan con 30 o más años de experiencia docente en la asignatura.

Tanto los docentes nuevos como los antiguos, reconocen que la forma pedagógica que utilizan en sus prácticas docentes, son las mismas que se utilizaban cuando ellos aprendieron programación, esto es, una forma de instrucción frontal, basada sólo en la transmisión y presentación de contenidos, con algunos momentos en que los estudiantes trabajan de manera individual, en el uso y aplicación de los contenidos para la resolución de problemas que les son propuestos. Bajo esta modalidad pedagógica, los estudiantes son meros receptores de los contenidos.

La mayoría de estos docentes, creen que al utilizar formas de instrucción que permitan una participación más activa de los estudiantes, generará pérdidas importantes de tiempo, que harán que no puedan tratar todos los contenidos del programa de la asignatura.

En opinión general de estos docentes, ellos consideran que es responsabilidad directa y única de los estudiantes, realizar las acciones que les permitan alcanzar niveles adecuados de aprendizaje para aprobar la asignatura.

Sin embargo, es posible apreciar que un número reducidos de estos docentes, consideran que es urgente estudiar la problemática del aprendizaje y desarrollar iniciativas que permitan mejorar los resultados del aprendizaje.

2.3. Lo que se sabe respecto de la problemática del aprendizaje en la primera asignatura de FP

Como hemos argumentado en apartados anteriores, para la generalidad de los estudiantes, aprender a programar es un proceso difícil, ya que demanda no solo el aprendizaje de variados conocimientos, muchos de ellos con un alto nivel de abstracción, sino que además exige disponer de la capacidad de utilizar estos conocimientos, y de usar ciertas estrategias específicas para resolver los problemas de procesamiento de datos.

Para ello, es necesario aprender, manejar y utilizar los conocimientos teórico-formales que constituyen la base de sustentación disciplinar, y aprender los conocimientos de tipo procedimental, que son propios de la metodología que se aplica para la resolver este tipo de problemas, y los de un lenguaje de programación de computadores con el cual implementar los programas computacionales que automatizan las soluciones.

En términos generales, la adquisición y el dominio de estos conocimientos son los que permitirían a un estudiante, desarrollar las competencias necesarias para constituirse en un resolutor efectivo de problemas de procesamiento de datos y en un programador.

Como hemos señalado, adquirir, desarrollar y aplicar los conocimientos para la resolución de problemas y la PC, es un proceso altamente complejo, ya que involucra una variedad de actividades cognitivas, y representaciones mentales relacionadas por ejemplo, con el análisis y el diseño de los programas, la comprensión de los mismos, la modificación y la depuración de ellos, entre otras tareas intelectuales, además del dominio de determinados conocimientos conceptuales referidos al manejo de las estructuras básicas de programación tales como ciclos, sentencias condicionales, etc.

La experiencia y los resultados académicos, muestran que la generalidad de los estudiantes tiene dificultades para aprender estos conocimientos, y dificultades para aplicarlos de manera efectiva. A este respecto, los autores Gomes & Mendes (1999) identifican las siguientes:

- La PC es una disciplina que trata con el manejo de contenidos de alto nivel de abstracción, como por ejemplo los conceptos de, algoritmo; computador como máquina de cálculo; y lenguaje de programación.
- Muchos estudiantes creen que el aprendizaje de los conceptos no son importantes, y que para ser un buen programador, basta sólo con dominar un lenguaje de programación.
- La disciplina trata con problemas que para su solución, se necesita del manejo de conocimientos y de la práctica en las técnicas para resolverlos, y además, el dominio de estrategias generales y específicas para la resolución de ellos.
- En los cursos con muchos estudiantes, es imposible lograr un acoplamiento individualizado, en especial, respecto de los aspectos prácticos y aplicativos de los conocimientos.
- Generalmente los grupos curso están formados por estudiantes con niveles de conocimientos muy diferentes, lo que se traduce en ritmos de aprendizaje muy variados y tal vez descompensados, condición que las metodologías de enseñanza actualmente no logran manejar.

A esto, basados en nuestra experiencia, agregamos que:

- La tarea de resolver estos problemas implica el manejo de conocimientos declarativos, procedimentales, y lo más complejo, de conocimiento de tipo estratégico.

  Este último tipo de conocimiento, involucra dos aspectos, el conocimiento estratégico que sirve a la tarea de identificar, seleccionar, y organizar los conocimientos declarativos y procedimentales que se requieren para diseñar e implementar la solución de un cierto problema de procesamiento de datos,

y aquellos conocimientos estratégicos que se requieren para evaluar y corregir el uso y aplicación que se está haciendo de los conocimientos.
Creemos que estas dos dimensiones del conocimiento estratégico, constituyen las dificultades más críticas para el aprendizaje de los FP.

- El manejo del lenguaje de programación, exige el dominio de la sintaxis, la semántica y las estructuras de programación, lo que en sí es una dificultad para algunos pocos estudiantes.
- La asignatura exige una visión práctica del manejo y uso de los contenidos, lo que difiere del aprendizaje de otras disciplinas que se basan en nociones más bien teóricas, cuyo aprendizaje depende de las habilidades de comprensión lectora y memorización.
- En el desarrollo de la asignatura, se hace uso intensivo de metodologías de enseñanza tradicionales, lo que no facilita el aprendizaje de conceptos que tienen un carácter dinámico.
  Estas metodologías no favorecen la adquisición y el desarrollo de las estrategias cognitivas y de pensamiento, que se requieren para la puesta en práctica de los conocimientos en la propia tarea de resolución de los propios problemas.

2.3.1. Dificultades cognitivas de la PC

Green (1990), plantea que uno de los aspectos más importantes a considerar al momento de analizar las dificultades que tienen los estudiantes al resolver problemas de programación, es que ésta no es una tarea intelectual que se basa en una “simple transcripción de representaciones internas que el estudiante ya posee”, sino que, demanda de ellos el desarrollo de un proceso exploratorio, donde las soluciones y los programas son creados para abordar un problema particular, y obedecen a un proceso de aprendizaje que debe llevarse a cabo de forma incremental. Un planteamiento similar ha sido expresado por Visser (1990).

En esta misma línea de análisis, Davies (1993), establece que para programar computadores, no basta con poseer una buena provisión de conocimiento declarativo, sino que además se requiere de la capacidad para determinar de qué forma se debe aplicar este conocimiento para la creación e implementación de programas.

Nosotros compartimos esta visión, sin embargo, la hacemos más complementaria y extensiva, en el sentido de que para llevar a cabo el proceso completo de resolución de problemas, se requiere tanto del conocimiento declarativo y el procedimental, y a partir de ellos, se debe tener la habilidad para combinarlos y organizarlos, para producir el conocimiento estratégico, que es el que en definitiva permite crear la solución a un problema particular.

Como hemos argumentado anteriormente, en este conocimiento estratégico también se encuentran implicados, los procesos cognitivos que permiten al resolutor, evaluar la efectividad de su propio proceso de resolución, con lo cual evalúa la aplicación y uso que está haciendo de los conocimientos, y eventualmente, le permitirá determinar y realizar los cambios que sean necesarios para que el proceso conduzca a una solución que se efectiva.

Muestra experiencia nos permite asegurar que la construcción de este tercer tipo de conocimiento, el estratégico, se consigue mediante una práctica intensiva, organizada e intencionada en la resolución de problemas.

Desde la perspectiva operativa, Brooks (1990) indica, que cuando un programador experto intenta construir una solución, parte de una representación central del problema, lo que este autor denomina la “estructura del conocimiento relacionada con la situación que se le plantea”, para lo cual hace uso de conocimiento declarativo y procedimental, y luego a partir de esto, elabora un plan para crear la solución, lo que correspondería a la construcción del conocimiento que llamamos estratégico.

Para referirse a esto mismo Ormerod (1990), señala que un programador experto dispone de “esquemas”, los cuales consisten en conjunto de proposiciones que se encuentran organizadas por su contenido semántico. Estos esquemas le conducen a un plan distinguible, y consecuentemente, a un procedimiento para realizar la tarea que la solución demanda. Análisis similares pueden encontrarse en Anderson (2000).

Brooks (1977, 1983) precisa que las soluciones a los problemas y el respectivo programa, son creaciones desarrolladas para un propósito, con respecto a alguna tarea, problema, o especificación. Así entonces, quién intenta resolver un problema de este tipo, debe disponer en primer término, de un modelo mental que le permita la comprensión del dominio del problema, el cual es previo a cualquier intento por diseñar la solución y el acto mismo de escribir el programa correspondiente.

Juicios similares se pueden encontrar en Spohrer, Soloway, and Pope (1989), Davies (1993), Rist (1995), quienes además, recomiendan que el primer curso de programación para carreras de CcC, debe basarse en modelos de resolución de problemas, donde las características de los lenguajes de programación, sean introducidos, sólo en el contexto de las soluciones que el estudiante vaya creando y desarrollando para problemas específicos.

De lo expuesto podemos sintetizar que, los elementos cognitivos que intervienen en el acto intelectual de resolver problemas de programación, se encuentran implicados tanto los conocimientos declarativos, como los procedimentales, y la capacidad para, a partir de estos, construir el conocimiento estratégico.

En esta capacidad interviene, la habilidad para determinar la forma en que los dos primeros tipos de conocimientos deberían ser usados en una situación concreta de resolución de un problema.

Junto con lo anterior, se requiere también de la disponibilidad de adecuados modelos de representación de estos conocimientos, y del manejo de ellos.

2.3.2. Las habilidades que debe aprender un novato para ser un experto en PC.

En estos últimos 22 años, se han realizado varios estudios orientados, en primer término, a determinar cuáles son las características de actuación que tiene un experto en la resolución de problemas y la PC, esto con el objeto de establecer las diferencias entre un experto y un novato. En segundo término, estos estudios se han dirigido a establecer en qué ámbitos se encuentran los principales problemas que presentan los novatos.

Dentro de los estudios sobre expertos se encuentra los realizados por Dreyfus y Dreyfus (1986), quienes establecieron una clasificación de etapas para determinar el paso de un novicio a experto. En este tránsito, estos autores han identificado las etapas de: novicio, principiante avanzado, competente, perito, y experto.

Luego de esto, los estudios sobre expertos se centraron en establecer las complejidades de las representaciones del conocimiento, para la programación y las estrategias de resolución de problemas que éstos pueden utilizar. Ejemplos de este tipo de estudios se encuentran en los trabajos de Détienne (1990); Gilmore, (1990); Visser & Hoc (1990).

Uno de los aspectos centrales relacionado con la representación del conocimiento se encuentra en la capacidad para llevar a cabo el análisis de un problema, respecto de lo cual Spohrer y Soloway (1989), identificaron nueve clases de errores que cometen frecuentemente los novatos al abordar un problema, y que describen como:

1.- Problema de interpretación; se relaciona con una inadecuada interpretación de las especificaciones implícitas en el enunciado del problema, ello como consecuencias de un fallido desarrollo del proceso de análisis.
2.- Problema del lenguaje natural; se refiere a los errores generados por una mala interpretación del enunciado de los problemas.
3.- Análisis incompleto; ocurre cuando realizan un análisis incompleto del enunciado del problema, lo que hace que los aspectos esenciales del mismo no sean considerados.
4.- Problemas de fronteras o límites; al igual que el anterior, resulta de un proceso de análisis mal realizado, lo que se traduce en una determinación incorrecta de los límites del problema.

5.- Problema de casos inesperados; la describen como otra consecuencia de un análisis defectuoso, el que ocurre cuando no se han considerado todos los casos posibles que puede arrojar la solución.
6.- Problema de especialización; se produce también por un análisis que defectuoso, que hace que se presenten errores cuando éste se adapta a una situación más específica.
7.- Problema de optimización; que resulta de un diseño que produce una solución que no es óptima.
8.- Problema de experiencias previas; ocurre cuando el resolutor hace un mal uso de ellas o las aplica de manera inapropiada.
9.- Problemas de carga cognitiva; que ocurre cuando partes pequeñas pero significativas de la tarea de análisis, se omiten o son realizadas de manera incompleta.

Conforme a esta identificación, estos autores reconocen fundamentalmente, problemas asociados con la tarea de análisis del problema, que en términos prácticos, corresponde a la primera etapa de la metodología de resolución de problemas.

Desde nuestra experiencia, si bien reconocemos la existencia de estas dificultades, también se observan dificultades importantes en la etapa de diseño de las soluciones, principalmente en lo referido a la tarea de elaborar la organización lógica de las secuencias de acciones y operaciones, que expresan la solución a un problema, es decir, en el diseño del algoritmo.

Nuestros juicios están avalados por los hallazgos de von Mayrhauser & Vans (1994) quienes en un estudio sobre compresión de programas, observaron que los expertos, a diferencia de los novatos:

- Poseen una organización eficiente de esquemas y de conocimiento especializado, y
- Tienen una buena capacidad para organizar sus conocimientos de acuerdo a las características funcionales, tales como la naturaleza del algoritmo implicado en la solución, en cambio los novatos los organizan en base a detalles superficiales, como la sintaxis del lenguaje de programación que usaran para implementar el programa.
- Para el diseño de las soluciones, usan estrategias generales de resolución de problemas, tales como “divide y vencerás”, junto con estrategias especializadas que son específicas para cada problema.

Así, los expertos cuando resuelven problemas de programación, usan esquemas especializados y un enfoque “top-down” para descomponer el problema eficientemente, como forma alcanzar una mejor comprensión de los programas.

Por otra parte, estos autores indican que disponen de evidencias que señalan que los expertos son flexibles en sus enfoques, tanto cuando tratan con la compresión del programa, como cuando deciden abandonar las hipótesis que les parecen cuestionables.

Para nosotros estas habilidades que evidencian los expertos a diferencia de los novatos, son consecuencia directa de sus experiencias prácticas en las tareas de resoluciones de problemas de procesamiento de datos. Con esto queremos enfatizar que, un experto no es sólo aquel que dispone de una buena provisión de conocimientos declarativos y procedimentales, sino que por sobre todo, es un resolutor que tiene una práctica extensiva en tareas de resolución de problemas de procesamiento de datos.

Consideramos que esta práctica directa es fundamental para el desarrollo y el perfeccionamiento del conocimiento estratégico, que es la base del conocimiento experto.

En esta misma línea de manera complementaria, Linn & Dalbey (1989) concluyeron que los expertos, son capaces de asociar a sus esquemas de solución, estrategias de prueba y depuración que les permiten validar la eficacia de ellas.

En el segundo ámbito de estos estudios, están aquellos que se orientan a establecer las dificultades que presentan los novatos. En esta perspectiva Winslow (1996), reporta que una de las principales dificultades que presentan los estudiantes que se inician en el aprendizaje de la programación, es la carencia de conocimientos y habilidades específicas.

Para este autor los estudiantes novatos en programación, poseen un conocimiento de tipo superficial, en el sentido de que lo tienen organizado en base a similitudes superficiales, y no disponen de modelos mentales detallados, lo que los hace cometer errores, cuando deben aplicar el conocimiento que es de relevancia para construir una solución para un cierto problema. Además, son estudiantes que aplican estrategias generales de resolución de problemas y no estrategias específicas.

Las estrategias específicas de resolución de problemas de procesamiento de datos, son aquellas que están determinadas por la propia metodología de resolución, y por las técnicas y las estrategias de pensamiento, entre las se encuentran el pensamiento abstracto, y la elaboración y el trabajo con distintos modelos, que como se recordará, son elementos esenciales de las estrategias de resolución de este tipo de problemas.

Los novatos tienen un enfoque de análisis de los programas del tipo “línea por línea”, y no disponen de habilidad para poner en práctica un enfoque orientado a la significación de fragmentos o estructuras de del mismo. Respecto de esto mismo, en

un estudio realizado por Wiedenbeck y sus colaboradores (Wiedenbeck et. al., 1999) se concluye que los novicios son muy locales y concretos en la comprensión de los programas.

A esto nosotros agregaríamos, las dificultades que tienen nuestros estudiantes para comprender el significado y el funcionamiento de ciertas estructuras que son comunes en los programas, como por ejemplo las estructuras de control anidadas.

En el ámbito de las actividades que son más críticas para el éxito de la tarea de resolver esta clase de problemas, Linn & Dalbey (1989) encontraron evidencias de que, de que a diferencia de un experto, los novicios dedican muy poco tiempo a las tareas de análisis y prueba de código, y abordan los errores de manera muy localizada, en lugar de tratar con la reformulación significativa para corregir los errores.

Esto es concordante con nuestra experiencia, ya que es frecuente observar que cuando los estudiantes son enfrentados a la resolución de un problema, ellos tratan de iniciar la escritura del código del programa lo más pronto posible, con lo que obtienen programas deficientes, que no responden a las especificaciones del problema, ello como resultado de un pobre o nulo análisis del enunciado del problema, y de los resultados que debe producir la solución al mismo.

Además, llevan a cabo prueba de código superficiales, lo cual no les permite realmente encontrar los errores, y tratan de realizar corrección al programa intentando “parchar” el código, sin antes realizar un análisis detallado de los errores, para así corregirlos de manera más efectiva.

En términos prácticos, estas actuaciones que son el resultado de un análisis y una prueba de código realizados muy a la ligera, se traduce en un mayor consumo de tiempo para resolver el problema, lo que además generalmente se refleja en la elaboración de soluciones incorrectas o poco eficientes, sobre las cuales tampoco disponen de mucha claridad respecto de cómo corregirlas o mejorarlas.

En el contexto de las dificultades más específicas que presentan los estudiantes novatos en programación, se encuentra el estudio realizado por Soloway & Spohrer (1989) según el cual se verifica, que estos tienen limitaciones para comprender varios de los constructos específicos de los lenguajes de programación, tales como el manejo de variables, ciclos, funciones y las estructuras recursivas.

Respecto de esto, Hoc (1989), estableció que los novicios presentan dificultades en el manejo de ciertas clases de abstracciones, como es el caso de las pruebas condicionales y el manejo de ciclos. En esta misma categoría de dificultades, Spohrer et. al. (1989), mostraron que los novatos cometen errores asociados con la

comprensión y el manejo de ciclos y condicionales, en el caso de un programa simple en Pascal, el cual consistía en la lectura de algunos datos y la realización de un proceso iterativo sencillo.

Así también du Boulay (1989), observó que los ciclos *for* son problemáticos para los novicios, ya que ellos fallan en la comprensión de que es lo que ocurrirá a continuación de que la variable de control del ciclo, haya sido actualizada o modificada.

Conforme a nuestra experiencia, si bien en un principio, los estudiantes presentan dificultades en el manejo de las pruebas condicionales, en especial en el manejo de condicionales dobles y anidadas, estas dificultades se resuelven rápidamente, conforme trabajamos en ejercicios que hemos diseñado específicamente para trabajar estas estructuras.

Winslow (1996), señala que en un número importante de estudios respecto de las actuaciones de los programadores novatos, se concluye que estos, aun conociendo la sintaxis y la semántica de varias de las sentencias de un lenguaje de programación, no son capaces de combinar las características de ellas, dentro de un programa coherentemente válido, lo que para este autor significa que, aun cuando ellos saben cómo resolver un problema, tienen dificultades para trasladar la solución algorítmica a un programa computacional equivalente.

Es necesario precisar que este autor, se focalizó especialmente en la creación de un programa, más que en la resolución del problema mismo, observando por ejemplo, que la mayoría de los estudiantes son capaces de establecer un procedimiento para promediar una lista de números, pero tienen dificultades para implementar un fragmento de código que considere un ciclo para realizar la misma operación.

Estas dificultades nosotros las tratamos, trabajando de manera intensiva en la construcción y comprensión de las representaciones algorítmicas, y estableciendo una fuerte relación entre los elementos del procedimiento expresado en la representación algorítmica, y los recursos del lenguaje de programación para implementar dicho procedimiento. Para estos efectos, utilizamos preferentemente representaciones algorítmicas en seudocódigo, por su cercanía con las estructuras de programación de los lenguajes imperativos y estructurados.

En esta misma perspectiva, en nuestra práctica docente hemos constatado las dificultades que presentan los estudiantes en la comprensión del funcionamiento de los ciclos, en la capacidad para distinguir sus diferentes tipos, cómo se controlan cada uno de ellos, y cómo aplicarlos a situaciones específicas, y como señalamos anteriormente, estas dificultades también se relacionan con la comprensión de ciclos anidados.

Teniendo en cuenta que los ciclos constituyen un recurso de programación que es fundamental, entre otras cosas, para manejar los arreglos y las operaciones sobre ellos, lo que se hace muy evidente, cuando se trata con operaciones sobre arreglos vi y tridimensionales.

En un ámbito de mayor complejidad, están las dificultades en el uso y aplicación del concepto de recursión. El propio concepto de recursión, es reconocido por muchos autores, como, altamente abstracto (ver por ejemplo a Dijkstra (1989); Burkhardt, Détienne & Wiedenbeck (1997); Kay et. al. (2000); Proulx (2000); Rist (1996); Vilalta & Drissi (2002); Wiedenbeck & Ramalingam (1999)).

Respecto de esto, en nuestra práctica docente hemos observado que los estudiantes tienen dificultades, para comprender el mecanismo de iteración en la recursión, así también tienen dificultades para entender, cómo hacer la llamada recursiva, y en cómo establecer y definir de la condición para finalizar las llamadas recursivas.

Para nosotros, todas estas dificultades tienen un denominador común en el sentido que son conceptos muy abstractos, de los cuales se demanda una visión práctica, ya sea en términos de su uso y aplicación, como su compresión cuando forman parte de un programa.

Nuestra propuesta para abordar estas dificultades, es que durante el tratamiento y aplicación de los contenidos de la asignatura, se debe privilegiar la práctica directa, intencionada, organizada y sistemática, en las que los estudiantes sean puestos a trabajar en la resolución de situaciones problemas que involucren estos conceptos y contenidos.

Según Brooks (1977, 1983), Spohrer, Soloway, y Pope (1989), Davies (1993) y Rist (1995), para lograr el manejo de estos constructos específicos, los novicios deben disponer primero de un modelo mental que les permita la comprensión del dominio del problema, para lo cual ellos señalan, que el primer curso de programación para CcC, debe estar basado en modelos de resolución de problemas, donde las características de los lenguajes de programación, se introduzcan sólo en el contexto de las soluciones que el estudiante consiga crear y desarrollar para problemas específicos.

Desde el punto de vista curricular, esta indicación es consecuente con las recomendaciones contenidas en el CC2001 (CC2001), en lo referido al desarrollo del área de conocimientos de FP, en las que se señala que lo importante en un primer de programación, es que los estudiante aprendan los fundamentos de la programación de computadores, más que aprender a resolver problemas de programación de alta complejidad.

Sin embargo para que esto sea efectivo, Cañas, Bajo y Gonzalvo (1994); du Boulay (1989); du Boulay, O'Shea y Monk (1989); Hoc & Nguyen-Xuan (1990); Mayer (1989); y Menselsohn, Green y Brna (1990), han planteado que es necesario que los novicios también dispongan de un adecuado modelo de computador, esto es, que manejen la abstracción de lo que comúnmente se llama "concepto de máquina", el que debe estar definido con respecto a un lenguaje de programación, ya que el concepto de máquina bajo el lenguaje de programación Pascal, o C es muy diferente a uno bajo el lenguaje de programación Prolog o Lisp. Este modelo es fundamental para comprender el comportamiento de un programa en ejecución.

Para du Boulay (1989), el dominio del concepto de máquina no es fácil de alcanzar para los estudiantes, y generalmente toma mucho tiempo y maduración del mismo, ya que demanda aprender la relación entre el programa escrito en un papel, y el procedimiento que el programa realiza al ser ejecutado por un computador real.

Además, se debe considerar la complejidad que implica ver un programa de diferentes formas, ya sea mediante una visión de orden lineal (sentencias ejecutándose una a una), o la visión de su flujo de control, o la visión de las transformaciones de los datos, o la visión de la estructura de sus módulos o de las estructuras de sus objetos.

Estas afirmaciones son consecuentes con nuestra experiencia en el sentido de que, cuando enfrentamos a nuestros estudiantes al análisis y la predicción de los resultados, que producirá la ejecución en un computador de un determinado fragmento de código, ellos cometen errores importantes.

Las dificultades son aún mayores, cuando en el fragmento de código se incluyen ciclos, estructuras de datos del tipo arreglos, y/o llamadas y funciones. Sin embargo hemos podido observar que en la medida en que los ponemos en situaciones intencionadas de práctica, en las cuales ellos asumen un rol activo en las tareas de compilación, ejecución y verificación de los resultados de un programa completo o de fragmentos del mismo, esta dificultad se supera rápidamente, mejor aún, si tienen oportunidades de discutir y cuestionar mientras realizan estas tareas.

Relacionado con esto, Corritore & Wiedenbeck (1991), encontraron que los novicios tienen más dificultades con el flujo de datos y con las cuestiones del propósito de las funciones, que con el control del flujo del programa, y tienen menos problemas con las operaciones elementales, como por ejemplo con la asignación de una variable.

Nuestra experiencia nos muestra que este escenario se complica aún más, cuando se necesita tener clara la distinción entre el modelo de programa que se está intentando construir, y el modelo de programa que realmente se ha obtenido, ya que

esto es el primer paso para determinar qué errores tiene el programa de solución que se ha obtenido.

El diseño de la solución que realiza el estudiante puede ser incorrecto por varias razones, como por ejemplo, pueden ocurrir iteraciones no predecibles en un ciclo, o la variable de control del mismo no está bien manejada, etc.. Estos son algunos de errores asociados con el manejo de ciclos, que frecuentemente los estudiantes no son capaces de detectar.

Para manejar correctamente estos casos, el estudiante debe tener la capacidad para detectar porqué el programa se está ejecutando de una forma no esperada, usando por ejemplo, la técnica de rutear a mano del código con el objeto de construir un modelo del programa y predecir su comportamiento, lo que Perkins et. al. (1989), denominan la capacidad para describir un programa tomando el punto de vista del computador.

Para se requiere que el estudiante maneje e integre dos modelos, el modelo del lenguaje y el modelo del comportamiento de la máquina, elementos que son esenciales para la comprensión del programa, para el análisis del problema, y para ejecutar correctamente los procesos de prueba y depuración del mismo.

Pennington & Grabowski (1990), han establecido que los programadores novicios tienen dificultades importantes en estas tareas, lo que se refleja también en las dificultades que tienen para realizar el proceso de depuración, y las correcciones que ello puede implicar.

Respecto de esto mismo, Perkins et. al. (1989), encontraron que los novicios tienen dificultades para realizar ruteo de programas, y tienen una pobre capacidad para capturar la naturaleza de la secuencia básica de ejecución del mismo, lo que confirma las limitaciones para llevar adelante los procesos que se señalan.

En esta perspectiva, en nuestra práctica docente hemos constatado que los estudiantes tienen dificultades para elaborar casos de prueba, que permitan verificar de manera efectiva la correctitud de la solución alcanzada. En los ejercicios que desarrollamos en clases, enfrentamos a los estudiantes al diseño de casos de prueba, tanto de soluciones construidas por ellos, como a soluciones pre-establecidas, y en ambas situaciones, presentan problemas para diseñar los casos de pruebas que sean adecuados para verificar la correctitud de la solución.

Nuevamente enfatizamos la importancia que tiene la práctica directa de los estudiantes, en la ejecución de estas tareas para la mejora de estas dificultades, especialmente si la práctica va acompañada de instancias, en las cuales los estudiantes sean puestos en situaciones que les inviten a discutir, cuestionar y

defender sus percepciones y conclusiones, mientras van desarrollando las tareas que hemos descrito.

En el ámbito de otras dificultades de carácter más específico que presentan los estudiantes novatos, Samurçay (1989) exploró el manejo del concepto de variable que ellos poseen, mostrando que la inicialización resulta ser una operación cognitiva compleja, junto con la lectura y/o la entrada externa de datos. Así también la actualización y la prueba de variables resultan ser de complejidad equivalente.

Nosotros no concordamos con esta apreciación. Para la mayoría de nuestros estudiantes, estos conceptos quedan suficientemente claros usando la estrategia de explicar el concepto de memoria, localización de memoria, y la relación entre estos conceptos y el concepto de variable. Además mediante estas explicaciones, los estudiantes alcanzan una comprensión clara del concepto de tipo de dato, lo que se puede almacenar en un cierto tipo, y porqué cada tipo de dato tiene especificado un tamaño en byte para su almacenamiento.

De lo expuesto en este apartado es posible concluir, que las dificultades que diferencian la forma en que actúa un novato y un experto al momento de llevar a cabo las tareas de resolución de problemas de procesamientos de datos, se pueden clasificar en dos categorías:

A.-Las dificultades de tipo generales; dentro de las que consideramos más relevantes, están por ejemplo:
- ♦ Uso de estrategias generales y no específicas,
- ♦ Organización deficiente de esquemas de conocimiento especializado,
- ♦ No disponibilidad de conocimientos y habilidades específicas y de modelo mentales detallados,
- ♦ Carencia de habilidades de pensamiento y de manejo de conceptos abstractos, y
- ♦ Dificultades para la elaboración de modelos de representación que integren el pensamiento y los conceptos abstractos.

B.-Las dificultades de tipo específicas; ámbito en el cual consideramos de importancia las referidas a:
- ♦ Uso de los aspectos semánticos del lenguaje de programación y de las estructuras de programación,
- ♦ Dificultades en el manejo del concepto de computador como máquina de cálculo y programable,
- ♦ Dificultades en la interpretación de algoritmos y programas, y
- ♦ Dificultades para realizar las tareas de prueba y depuración de programas.

En síntesis, son dificultades referidas a los aspectos de carácter práctico de la propia tarea de resolución de los problemas, las que creemos son dificultades que pueden

superarse con la practica organizada, intencional y socializada en las tareas de resolución de problemas.

2.3.3. Estilos cognitivos predominantes de los estudiantes en los cursos de FP

Como se sabe en términos generales, los estilos cognitivos o también llamados estilos de aprendizaje, se refieren tanto a aspectos externos que son visibles cuando un aprendiz se enfrenta a una cierta tarea de aprendizaje, como también, a aspectos más internos, referidos a cómo intelectualmente se enfrenta a dicha tarea.

En este contexto, a nosotros nos interesa además, cómo los estudiantes deberían enfrentarse a la tarea de aplicar los conocimientos aprendidos, cuando trata con la resolución de problemas, ya que entendemos que al realizar este tipo de tareas de manera exitosa, conlleva también un aprendizaje, ya que mediante su realización, el estudiante está construyendo conocimiento estratégico, que como hemos señalado, es el conocimiento más difícil de construir, y que en términos prácticos, es el tipo de conocimiento que posee el experto de resolución de problemas de este tipo.

Existen muy pocas investigaciones que traten con la identificación de los estilos cognitivos de los estudiantes en los cursos de PC. Dentro de ellas se encuentra el trabajo desarrollado por Bishop-Clark (1995), quién en un conjunto de estudios, reportó que no existen diferencias entre los estudiantes, en relación con los estilos cognitivos y la personalidad al momento de enfrentarse a un curso de PC.

Por su parte en un estudio más reciente, Rountree, Rountree & Robins (2002) investigaron la relación entre factores como, conocimientos previos, intencionalidad y carga de trabajo esperada, de estudiantes en un curso introductorio de PC, de lo que concluyeron que, el predictor más confiable de éxito de los estudiantes, fue el grado de expectativas de éxito que los movilizaba.

Desde la perspectiva del comportamiento externo, nuestra experiencia docente nos muestra que los estudiantes en este tipo de asignaturas, realizan sus procesos de aprendizaje en solitario, y rara vez lo hacen de manera grupal, incluso al ser consultados respecto de esto, señalan que el estudio en grupo les parece menos efectivo que el estudio individual. Esta característica también la observamos cuando les pedimos que resuelvan algunos ejercicios en las sesiones de práctica en el laboratorio de computadores.

Ahora bien, conforme a la caracterización que hemos hecho del proceso de resolución de los problemas, y al tipo de actividad cognitiva que debe darse durante el trabajo intelectual de la resolución, hemos resaltado la importancia que tienen los procesos de selección e integración de conocimientos (declarativos y procedimentales), así como también la importancia que tiene el trabajo con

abstracciones y modelos, y por último, la importancia que tiene el uso y el desarrollo del conocimiento estratégico.

Estos aspectos del proceso de pensamiento durante la resolución de los problemas, se ven favorecidos y potenciados, si los estudiantes trabajan de manera colectiva en las tareas de resolución, ya que fundamentalmente, existen mejores oportunidades para compartir, discutir, reflexionar, defender y consensuar las visiones y construcciones particulares que van desarrollando, y de esta manera enriquecer sus propios procesos de aprendizaje, recogiendo las formas de realizar la tarea, y los productos que sus pares van elaborando y presentando al grupo.

Desde nuestra visión consideramos que para alcanzar el aprendizaje de los conocimientos declarativos y procedimentales, y para desarrollar la capacidad de aplicarlos durante el desarrollo de la tarea de resolución de los problemas, y junto con ello, desarrollar el conocimiento estratégico que permita alcanzar un cierto nivel de experto en la resolución de estos problemas, los estilos cognitivos que los estudiantes deberían usar, se corresponden con los definidos por Honey y Mumford (1986) como:

- Activo; ya que se espera que los estudiantes actúen como individuos abiertos, entusiastas, sin perjuicios ante las nuevas experiencias de aprendizaje de contenidos y el desarrollo de capacidades.
- Reflexivo; puesto que se requiere de estudiantes que tengan la capacidad para observar y analizar los conocimientos que aprenden, y de la forma de usarlos y aplicarlos.
- Teórico; ya que el aprendizaje de la disciplina de la programación, requiere del uso de un pensamiento lógico, en el que se integran los elementos de tipo práctico de aplicación de los conocimientos disciplinares, dando origen a estructuras lógicas complejas, tanto por sus elementos, como por las relaciones que se dan entre ellos, durante el proceso de resolución de los problemas.
- Pragmático; ya que son estudiantes que deben estar en disposición de poner en práctica sus conocimientos e ideas, en la propia tarea de resolución de los problemas, buscando la eficacia en sus acciones y decisiones, durante el desarrollo de la tarea.

Para estos efectos, consideramos entonces, que es fundamental que esta asignatura sea conducida conforme a una metodología que promueva un rol más activo de parte de los estudiantes, en las tareas de aprendizaje, las cuales han de ser conducidas por las actividades prácticas referidas al uso y aplicación de los conocimientos para el diseño de las soluciones de los problemas. Hablamos de actividades donde los estudiantes se involucren de manera directa en las discusiones y defensas de las posturas particulares respecto de las soluciones a un problema.

Creemos que mediante esta metodología los estudiantes tienen mejores condiciones para poner a prueba sus conocimientos y la aplicación que están haciendo de ellos, para corregir la forma en que usan sus conocimientos, lo que en definitiva ha de transformar en una mejor construcción del conocimiento de experto para la PC.

2.3.4. Dominios de conocimientos que presenta logros deficitarios

Las características de los estudiantes y las formas clásicas de desarrollar esta asignatura, son determinantes en el bajo rendimiento académico y en la alta tasa de reprobación que tienen los primeros cursos de programación, situación que se evidencia en Universidades nacionales y extranjeras, así como también en la Universidad de Playa Ancha.

Tomando como referencia los resultados de rendimiento académico y nieles de aprobación de los últimos cinco años en la Universidad Técnica Federico Santa María, Pontificia Universidad Católica de Valparaíso, Universidad de Valparaíso, y Universidad de Playa Ancha, todas universidades de la quinta región de Chile, se verifica que el rendimiento promedio de los estudiantes que aprueban el curso, no supera la nota 5.0 (en la escala de notas de 1.0 a 7.0, con nota mínima de aprobación 4.0), y que los niveles de reprobación fluctúa entre un 35% y un 70%.

Resultados similares de bajo rendimiento han sido informados también en ciertas universidad extranjeras por: Alonso, García y Mollá (2005); Bravo, Redondo y Ortega (2004); Brusilovsky, Kouchnirenko, Miller and Tomek (1994); Fernández, Trella, Galindo y Aranda (2006); van Merrienboer (1990); y Yu-Fen and Stephen (1993).

En conversaciones informales que hemos sostenido con académicos que dictan la asignatura en las Universidades de la quinta región antes citadas, es posible apreciar que existe unanimidad en reconocer que los estudiantes tienen importantes dificultades de rendimiento académico en esta asignatura. Existen coincidencias en señalar que las causas más probables del fracaso son:

- Que se trata de una asignatura que demanda el aprendizaje y aplicación de conocimientos de carácter complejo y con un alto nivel de abstracción,
- Heterogeneidad en los niveles de conocimientos previos de los estudiantes que ingresan a estas carreras,
- En algunos casos, cursos muy numerosos,
- Escasos o nulos conocimientos previos en PC de los estudiantes,
- Las preconcepciones de los estudiantes, en el sentido de que ellos creen que para programar computadores, sólo basta con dominar un lenguaje de programación,
- Carencia de estrategias cognitivas adecuadas para el aprendizaje de los contenidos y las capacidades que demanda la asignatura,

- Conocimiento y práctica escasa o nula en estrategias de resolución de problemas,
- Académicos titulados en Ingeniería Informática o áreas afines, y que aunque en gran número tienen estudios de post-grados en el área de CcC, así como una experiencia docente que fluctúa entre los 5 y 20 años, carecen de capacitación pedagógica sistemática y formal.

Estos antecedentes los hemos compartido y reflexionado en dos instancias de desarrollo del Congreso Internacional de Educación en Ingeniería, que organiza la Sociedad Chilena de Educación en Ingeniería, con el patrocinio del Colegio de Ingenieros de Chile, la primera corresponde al XVII Congreso Chileno de Educación en Ingeniería, realizado en la Universidad Católica del Norte, Antofagasta, y segundo al XVIII Congreso Chileno de Educación en Ingeniería, realizado en la Universidad del Bío-Bío, Concepción.

Conforme a nuestra experiencia docente de estos últimos 8 años dictando la asignatura de FP en la carrera de Ingeniería Informática, tanto en la Universidad de Playa Ancha como en la Universidad de Valparaíso, así como en asignaturas similares en planes de formación de otras carreras de ingeniería, como en otras carreras que no pertenecen al área de formación en ingeniería, nos permite identificar un conjunto de dificultades que presentan los estudiantes en los cursos de FP, las cuales se relacionan principalmente con dificultades para realizar las actividades de orden práctico que están implicadas en la resolución de problemas y la PC.

Estas dificultades se reflejan en la débil o nula competencia que consiguen los estudiantes en varios ámbitos, los cuales listamos a continuación relacionándolos con la especificación de dominios de conocimientos que presentamos la Tabla del anexo:

- Diseño e implementación de la solución al problema, relacionado con las competencias 1.1., 1.2., 1.3., 1.4., y 1.5.
- Utilización del lenguaje algorítmico para representar la solución, relacionado con las competencias 1.1., 1.2., 1.3., 1.4., 1.5., 2.1., 2.2., 2.3., 2.4., 2.5., 2.6., 2.7., 2.8., y 2.9.
- Diseño de casos de prueba, relacionado con las competencias 7.1., 7.2., 7.3., y 7.4.
- Elección de la estructura iterativa apropiada, relacionado con las competencias 8.1., 8.2., 8.3., 8.4., 8.5., y 8.6.
- Elección de la estructura de datos apropiada para un conjunto de procesos, relacionado con las competencias 9.1., 9.2., 9.3., 9.4., 9.5., 9.6., y 9.7.
- Descripción del funcionamiento de los algoritmos de ordenamiento, relacionado con las competencias 2.4., 2.7., 2.8., 2.9., 10.1., y 10.2.

- Elección del algoritmo de ordenamiento más apropiado para un procesamiento determinado, relacionado con las competencias 2.4., 2.7., 2.8., 2.9., 10.1., 10.2., y 10.3.
- Descripción del funcionamiento de los algoritmos de búsqueda, relacionado con las competencias 2.4., 2.7., 2.8., 2.9., y 11.1.
- Discriminar entre uso de procedimientos y funciones, relacionado con las competencias 12.1. y 12.2.
- Organizar programas modularmente, relacionado con las competencias 12.1., 12.2., 12.3., 12.4., 12.5., y 12.6.

Si analizamos estas dificultades en términos de las tres categorías de conocimientos que deben ponerse en juego para la resolución de problemas de procesamiento de datos, a las cuales hicimos mención en el apartado 1.2. y que corresponden a los conocimientos de tipo declarativos, procedimental y estratégico, podemos sostener que nuestros estudiantes en general, no evidencian dificultades importantes en el ámbito de los conocimientos declarativos y procedimental, pero si en el ámbito del conocimiento de tipo estratégico, es decir, en aquel conocimiento que el estudiante debe construir y aplicar en una situación particular de resolución de problemas, y en cuya construcción y aplicación, debe estratégicamente seleccionar y organizar los conocimientos declarativos y procedimentales que le serán de utilidad para la tarea.

Estamos hablando aquí de un conocimiento de carácter práctico, no memorístico, que debe ser construido por el propio estudiante, y para lo cual se requiere que él adopte un rol eminentemente activo y participativo durante su proceso de aprendizaje.

Como se ha señalado, la construcción de este tercer tipo de conocimiento demanda la puesta en práctica de estrategias cognitivas de orden superior, estrategias donde entren en juego procesos de análisis, reflexión, selección, organización, verificación y contrastación, para establecer cuales conocimientos declarativos y procedimentales son los que deberán utilizarse para construir la solución al problema.

Conforme a nuestra experiencia, claramente el aprendizaje y la puesta en práctica de estas capacidades y habilidades de orden superior, son las que constituyen las mayores dificultades que presentan los estudiantes.

Como elemento complementario a los antecedentes y argumentos anteriores, hemos realizado un análisis del rendimiento promedio de los estudiantes en el curso de FP, conforme a los resultados de las evaluaciones, relacionando cada evaluación con las competencias que se intentan medir (tabla 3 en la página siguiente).

Este análisis lo hemos desarrollado a partir de la información histórica de las evaluaciones con que contamos, la cual está conformada por los registros de notas

y el desarrollo en el tiempo de las actividades de clases y contenidos de la asignatura, para los últimos 3 años, tiempo en el cual se han dictado 4 cursos de FP para la carrera de Ingeniería en Informática de la Universidad de Playa Ancha de Valparaíso, Chile, y en los cuales han tomado parte un total de 223 alumnos.

Como muestran los resultados generales, las mayores dificultades de los estudiantes se encuentran relacionadas con el conjunto de competencias asociadas con los objetivos generales 1 y 2, los cuales corresponden a los contenidos de uso y aplicación de metodologías de resolución de problemas de procesamiento de datos, y al contenido de diseño de la solución.

Estos dos contenidos se relacionan directamente con las dos primeras etapas del proceso de resolución de problemas de procesamiento de datos, esto es, las etapas de análisis del problema y la fase del diseño de la solución.

Para desarrollar estas dos etapas, no basta con conocer la metodología de resolución de problemas, sino que además, se debe contar con la capacidad para, en un acto fundamentalmente creativo, diseñar la solución al problema.

Evaluación	Dominios implicados, referidos a la definición hecha en la tabla 4 en el Anexo	Nota promedio de rendimiento
Primera	1.1., 1.2., 1.3., 1.4., 1.5, 2.1., 2.2., 2.3., 2.4., 2.5., 2.6., 2.7., 2.8., y 2.9.	3.5
Segunda	1.5., 2.3., 2.4., 2.6., 2.7., 2.8., 3.1., 3.2., 3.3., 3.4., 4.1., 4.2., 4.3., 4.4., 4.5., 4.6., 4.7., 4.8., 4.9., 4.10., 5.1., 5.2., 5.3., 6.1., 6.2., 6.3., 6.4., 6.5., 6.6., 7.1., 7.2., 7.3., y 7.4.	4.3
Tercera	1.4., 1.5., 2.3., 2.4., 2.5., 2.6., 2.7., 2.8., 3.4., 4.5., 4.7., 4.8., 4.9., 4.10., 5.2., 5.5., 6.2., 6.3., 6.4., 6.5, 6.6., 7.2., 7.3., 7.4., 8.1., 8.2., 8.3., 8.4., 9.1., 9.2., 9.3., 9.4., 9.5., 9.6., y 9.7.	4.7
Cuarta	1.4., 1.5., 2.3., 2.4., 2.6., 2.7., 2.8., 2.9., 3.3., 3.4., 4.4., 4.5., 4.7., 4.8., 4.9., 4.10., 5.2., 5.5., 6.2., 6.3., 6.4., 6.5., 6.6., 7.2., 7.3., 7.4., 8.2., 8.3., 8.4., 8.5., 8.6., 9.4., 9.5., 9.6., 9.7., 10.1., 10.2., 10.3., 10.4., 10.5., 11.1., 11.2., 11.3., 11.4., 11.5., 12.1., 12.2., 12.3., 12.4., 12.5., 12.6., 12.7., y 12.8.	4.5

Tabla 3: Resumen de rendimiento en evaluaciones para los últimos 3 años

Teniendo en cuenta, que la etapa de diseño de la solución considera dos aspectos, el primero, relacionado con el diseño de la solución misma, y el segundo vinculado con la representación de la solución mediante un algoritmo, nuestra experiencia indica que el primero de estos dos aspectos es el más dificultoso para los estudiantes.

Apreciaciones similares a las que hemos expresado en los párrafos precedentes se pueden encontrar en: Boada, Soler, Prados y Poch (2006); Bravo, Redondo y Ortega (2004); Brusilovsky, Kouchnirenko, Miller and Tomek (1994); Fernández, Trella, Galindo y Aranda (2006); Salomon and Perkins (1987); Sánchez (2006); Van Merrienboer (1990).

En términos más específicos, las mayores dificultades de los estudiantes se encuentran en el desarrollo de los actos de pensamiento y en el uso de las estrategias cognitivas de orden superior que se requieren para el diseño de las soluciones. Como hemos señalado, en estos actos de pensamiento se encuentra implicada la capacidad de pensamiento abstracto y la creación y el manejo de los distintos modelos. En el ámbito de las estrategias cognitivas, se encuentran las estrategias para identificar, organizar y utilizar los distintos tipos de conocimientos.

Para apoyarles en estos aspectos, creemos que es absolutamente necesario la creación de ambientes que permitan trabajar de manera colectiva e implicativa en actividades de aprendizaje que, favorezcan el desarrollo y el perfeccionamiento del pensamiento abstracto específico para la resolución de problemas de procesamiento de datos, y complementariamente, que favorezcan el desarrollo y el perfeccionamiento de las estrategias cognitivas que hemos señalado.

Consideramos que si estas tareas de aprendizaje se abordan de manera colectiva, cada estudiante puede potenciar y enriquecer su aprendizaje individual, mediante la socialización e implicación en las tareas de resolución de los problemas.

En síntesis, conforme a nuestra experiencia y los resultados de las investigaciones, para la mayoría de los estudiantes, aprender programación es un proceso difícil, ya que por lo que se sabe, no sólo demanda el aprendizaje de variados conocimientos, muchos de ellos muy abstractos, para lo que se exige disponer de la capacidad de utilizar estos conocimientos, ámbito en el cual, es importante disponer de ciertas estrategias de pensamiento.

Gomes & Mendez (1999) señalan que las principales dificultades para aprender programación son:

- Contenidos de alto nivel de abstracción
- Se exige un buen dominio de los conocimientos disciplinares y de la práctica para aplicarlos y utilizarlos.
- Los lenguajes de programación poseen una sintaxis y una semántica que es compleja.
- Se demanda una visión práctica en el manejo de los contenidos.
- Las metodologías de enseñanza no son apropiadas para el tipo de aprendizaje que se demandan.
- Grupos de estudiantes muy heterogéneos en términos de conocimientos y habilidades previas.

Green (1999), Visser (1990) y Davies (1993), señalan que una de las dificultades importantes, es que en el aprendizaje de la PC, no opera sobre una simple transcripción de conocimientos, sino que se requieren de actos creativos de organización y aplicación de conocimientos a situaciones problemas nuevos, en otras

palabras, no es la simple aplicación de un procedimiento único y conocido previamente.

Para nosotros esto significa que para resolver problemas de este tipo, se requiere de una buena provisión de conocimientos declarativo y procedimental, y a partir de ellos, tener la capacidad para generar el conocimiento estratégico.

En un ámbito más específico Spohrer y Soloway (1989) identifican nueve dificultades que tienen los programadores noveles cuando abordan la resolución de problemas de programación, los cuales corresponden a: (1) Problema de interpretación, (2) Problema del lenguaje natural, (3) Análisis incompleto, (4) Problema de fronteras o límites, (5) Problema de casos inesperados, (6) Problema de la especialización, (7) Problema de optimización, (8) Problema de experiencias previas, y (9) Problema de carga cognitiva.

Nosotros agregamos a estos, los problemas de diseño de soluciones. Complementando lo anterior Winslow (1976) señala como problema, la carencia de conocimientos y habilidades específicas.

Otros autores como Soloway & Spohrer (1989), Hoc (1989), Spohrer et. al. (1989), du Boulay (1989), Brooks (1977, 1983), Rist (1995), Cañas, Bajo y Gonzalvo (1994), hacen mención a las dificultades que tienen los estudiantes en relación con el manejo de abstracciones y modelos.

Otros estudios hacen referencia a la identificación de los tipos de estudiantes. Rountree & Robins (2002) respecto de esto concluyen que factores como conocimientos previos, intencionalidad y carga de trabajo esperada, constituyen predictores confiables de éxito de los estudiantes en un curso de programación.

Finalmente, y basados en los resultados de las evaluaciones que disponemos respecto de cursos anteriores que hemos dictado, las mayores dificultades de rendimiento se aprecian en el desarrollo de las tareas de análisis y diseño de soluciones.

SÍNTESIS GENERAL DEL CAPÍTULO 2

La asignatura de FP es la primera asignatura del área de programación de los planes de estudio en CcC, y dada la importancia y trascendencia que tienen los contenidos que en ella se tratan, se ubica ya sea en el primer o segundo semestre de estos planes de estudio, respondiendo además de esta manera, a las recomendaciones curriculares contenidas en el informe CC2001, las cuales han sido reiteradas en el informe CSC2008.

En estos informes se dan los argumentos para establecer que el área de conocimientos de FP tiene un carácter troncal y nuclear para los planes de formación profesional en CcC y por tanto es de importancia esencial e ineludible para este tipo de currículos, ya que se considera que la capacidad para programar computadores, es una habilidad que debe poseer cualquier profesional de la CcC. Además, dada la composición y organización de estos currículos, se verifica que esta capacidad es un pre-requisito para abordar varias asignaturas de nivel medio y avanzado, contenidas en estos planes de estudios, que tienen relación directa con programación.

En atención a estos argumentos, con el desarrollo de esta primera asignatura se pretende la adquisición de los conocimientos teórico-formales y procedimentales de los fundamentos de programación de computadores, y junto con ello, la adquisición, el desarrollo y perfeccionamiento, de un conjunto de conocimientos que se pueden sintetizar en, la habilidad para comprender y explicitar los fundamentos teórico-formales de la PC y para aplicar estos conocimientos, y la metodología de la programación, para la resolución de problemas de procesamiento de datos y la implementación de los programas computacionales de solución a los problemas.

Conforme a estos propósitos, la asignatura trata con el desarrollo de cuatro áreas temáticas que son: Metodología de la programación; Resolución de problemas; Algoritmos y formas de representación algorítmica; y Lenguajes de programación.

Con el objeto de cubrir de manera adecuada estas áreas, los contenidos de la asignatura han sido agrupados en seis categorías que son: (1) Resolución de problemas de procesamiento de datos y la programación de computadores; (2) La metodología de resolución de problemas de procesamiento de datos; (3) Análisis del problema; (4) Diseño y representación algorítmica de la solución; (5) Transformación del algoritmo en un programa; y (6) Prueba y verificación del programa.

La asignatura de FP, se ubica en el primer o segundo semestre de las mallas curriculares de las carreras de Ingeniería en Computación y sus similares, con una carga horaria de entre 4,5 a 6 horas de clases semanales del semestre académico, que tiene una duración de entre 17 a 20 semanas.

Los docentes que dictan esta asignatura son en su gran mayoría profesionales ingenieros o técnicos del área de la computación e informática, con escasa o nula formación pedagógica, por lo cual para la presentación y transmisión de los contenidos de la asignatura, utilizan exclusivamente las clases expositivas.

Aunque la mayoría de ellos están preocupados por los altos niveles de fracaso que presenta la asignatura, no disponen de recursos metodológicos que les permitan diseñar y poner en práctica formas de conducir el proceso de enseñanza y aprendizaje, que apunten a mejorar los malos resultados. Sólo algunos de ellos han

intentado minimizar los malos resultados, por la vía de desarrollar apuntes de clases y guías de ejercicios, las cuales ponen a disposición de sus estudiantes.

Hasta hora, se han desarrollado varias investigaciones en el intento por identificar las dificultades que tienen los estudiantes cuando aprenden programación, las cuales podemos sintetizar en:

- Los contenidos de un curso de programación son de elevado nivel de abstracción y su aprendizaje exige de la práctica referida a su aplicación y al uso de técnicas de resolución de problemas de programación.
- Las dificultades de aprender los aspectos sintácticos y semánticos de los lenguajes de programación.
- La actividad de resolver problemas y la programación de computadores demanda la activación de procesos cognitivos de carácter exploratorio.
- La ejecución de la actividad intelectual requiere del manejo de variados modelos de representación, muchos de ellos de orden abstracto.
- Los estilos cognitivos y los aspectos de la personalidad de los estudiantes.

Aun teniendo en cuenta estos aspectos, basados en nuestra experiencia consideramos que el manejo y dominio de ciertas estrategias cognitivas por parte de los estudiantes, constituye un componente que es fundamental para el aprendizaje exitoso en esta asignatura.

En este contexto, hemos señalado que los estilos cognitivos (conforme a los definidos por Honey y Mumford (1986)) que los estudiantes deberían poner en acción en este tipo de cursos son: Activo, Reflexivo Teórico y Pragmático

Nos parece que las estrategias cognitivas específicas para resolver los problemas de programación tales como, las dirigidas a la identificación, selección, y organización de los conocimientos declarativos y procedimentales, son parte de los recursos que se requieren para construir el conocimiento estratégico que permite diseñar las soluciones.

Otra de las estrategias requeridas son las de control y regulación, las cuales, conforme a nuestra experiencia, son necesarias para que el resolutor pueda ir evaluando y corrigendo la propia aplicación de los conocimientos declarativos, procedimentales y estratégicos que va realizando mientras lleva a cabo las tareas de resolución de los problemas.

Así entonces, si bien existen dificultades que son propias de la complejidad de los contenidos con que trata la asignatura, existen también dificultades importante referidas a la disponibilidad y el manejo de ciertas estrategias de pensamiento que nos parecen vitales para el aprendizaje de los contenidos y para la aplicación de los

mismos en el contexto de la ejecución de las tareas intelectuales de resolución de los problemas de programación.

CAPITULO 3

PRINCIPIOS QUE HAN DE INFLUIR EN EL PROCESO DE APRENDIZAJE

3. PRINCIPIOS QUE HAN DE INFLUIR EN EL PROCESO DE APRENDIZAJE

INTRODUCCIÓN

En este apartado se expone el análisis de los principios que debieran influir en el proceso de aprendizaje de los fundamentos de la programación de computadores.

Como antecedente de referencia, se analiza en primer término, la importancia y relevancia que tiene la asignatura de FP, dentro de los cinco programas de formación profesional en computación e informática que hoy son reconocidos por la ACM. Usando como referencia el documento, The Guide to Undergraduate Degree Programs in Computing (CC2005), se presenta la definición de los propósitos que deberían alcanzarse al desarrollar el área de conocimientos de FP.

Basado en este documento, se revisan los objetivos y los contenidos de la asignatura FP, en el contexto del plan de formación profesional de la Carrera de Ingeniería en Informática de la Universidad de Playa Ancha, que es donde se ha llevado a cabo la intervención. Junto con ello, se analizan los bloques temáticos que aborda la asignatura, precisando las intencionalidades que se persiguen con el desarrollo de cada uno de ellos.

Además, se revisan los elementos ha de tener en cuenta para el aprendizaje de los conocimientos y las capacidades que se abordan con la asignatura, y los factores que inciden en la complejidad de la asignatura, y las dificultades que la misma representa para los estudiantes.

En la idea de tratar de manera más detallada las dificultades, se hace uso de un problema ejemplo, para analizar las características del tipo de actividad cognitiva que

tiene lugar, cuando se trata con la actividad intelectual de resolver problemas de procesamiento de datos.

A partir de lo anterior, establecemos los principios básicos para el aprendizaje de las capacidades que se requieren para resolver problemas de procesamiento de datos.

El capítulo finaliza con el análisis de las estrategias metodológicas, que deberían tenerse en cuenta para lograr el aprendizaje de las capacidades que se señalan

3.1. Perspectiva del desarrollo de la asignatura de FP

Como se ha planteado en apartados anteriores, la asignatura de FP es la primera asignatura con la que se aborda el área de conocimientos de PC, área que conforma uno de los núcleos esenciales y troncales, dentro de todos los planes de formación profesional en Ingeniería Informática, tanto en la UPLA, como también en universidades chilenas y extranjeras.

Conforme al análisis presentado en apartados anteriores, respecto de la importancia que tiene el área disciplinar de FP, para cualquier plan de formación profesional en Computación e Informática, así como la relación que tiene la primera asignatura de esta área, con otras asignaturas de estos planes, queremos ahora concentrarnos en analizar aspectos más específicos de esta primera asignatura, para el caso particular del plan de formación profesional en Ingeniería Informática de la UPLA en Chile, asignatura sobre la cual hemos llevado a cabo la intervención en el marco del desarrollo de la presente investigación.

Para este caso particular, la descripción general de la asignatura establece que ésta es una asignatura de tipo teórico-práctica, que persigue el propósito de que los estudiantes adquieran los conocimientos y las competencias fundamentales, que les permitan disponer de la capacidad para resolver problemas de procesamiento de datos y para programar computadores.

En conformidad con lo anterior, se declaran los objetivos generales y específicos que se presentan en la tabla 8 en la página siguiente:

A partir de esta formulación, es posible identificar dos bloques temáticos en el desarrollo de la asignatura, el primero relativo a la metodología de resolución de problemas de procesamiento de datos y los algoritmos, y el segundo relacionado con el computador como máquina de cálculo que puede ser programada, y el lenguaje de programación como recurso para la construcción de los programas.

Sin embargo es necesario enfatizar, que para alcanzar los propósitos que se persiguen con la asignatura, los estudiantes deben lograr el dominio integrado de estos dos bloques temáticos, es decir, deben ser capaces de aplicar de manera

correcta la metodología de resolución de problemas, y complementariamente, deben ser capaces de utilizar de manera adecuada los recursos del lenguaje de programación, con el objeto de implementar y probar el programa computacional que resulta del diseño de la solución.

OBJETIVOS GENERALES	OBJETIVOS ESPECÍFICOS
• Plantear sistemática y ordenadamente soluciones a problemas susceptibles de ser resueltos mediante la programación de un computador.	• Comprender y aplicar el concepto de algoritmo. • Diseñar algoritmos de solución. • Comprender y utilizar formas de representación algorítmica. • Interpretar y analizar algoritmos. • Comprender y aplicar el paradigma de programación estructurada. • Especificar, diseñar, analizar y representar algoritmos a partir del planteamiento de un problema.
• Hacer uso de un lenguaje de programación, de sus recursos y de un entorno de programación, para implementar y probar soluciones a problemas de procesamiento de datos.	• Comprender y aplicar distintas estructuras de programación. • Traducir representaciones algorítmicas a programa. • Comprender y aplicar los elementos básicos de un lenguaje de programación. • Comprender y utilizar las funcionalidades de un entorno de programación, para realizar tareas de escritura, edición, grabación, recuperación, compilación y depuración de programas.

Tabla 4: Objetivos Generales y Específicos de la Asignatura

Ahora bien, cada bloque temático puede ser descompuesto en términos de lo que los estudiantes debieran conocer, comprender, identificar, utilizar y aplicar. Con el objeto de aproximarnos a la identificación de estas capacidades, hemos hecho un desglose de ellas para cada uno de los dos bloques temáticos, los que ilustramos en el esquema que presentamos en la figura 13 en siguiente página:

Del esquema es posible deducir que en el logro de las capacidades que son propias de esta primera asignatura, se encuentran implicados tanto la adquisición de conocimientos teórico-formales, como también la adquisición de los conocimientos referidos a la forma de aplicar y utilizar estos conocimientos teórico-formales.

Como se ha señalado y justificado, la aplicación y utilización de estos conocimientos está fuertemente condicionada por el desarrollo y el perfeccionamiento de un conjunto de estrategias de pensamiento, de regulación y control, que garanticen la correcta aplicación y uso de estos conocimientos, en el contexto del desarrollo de la tarea intelectual de resolución de problemas de procesamiento de datos.

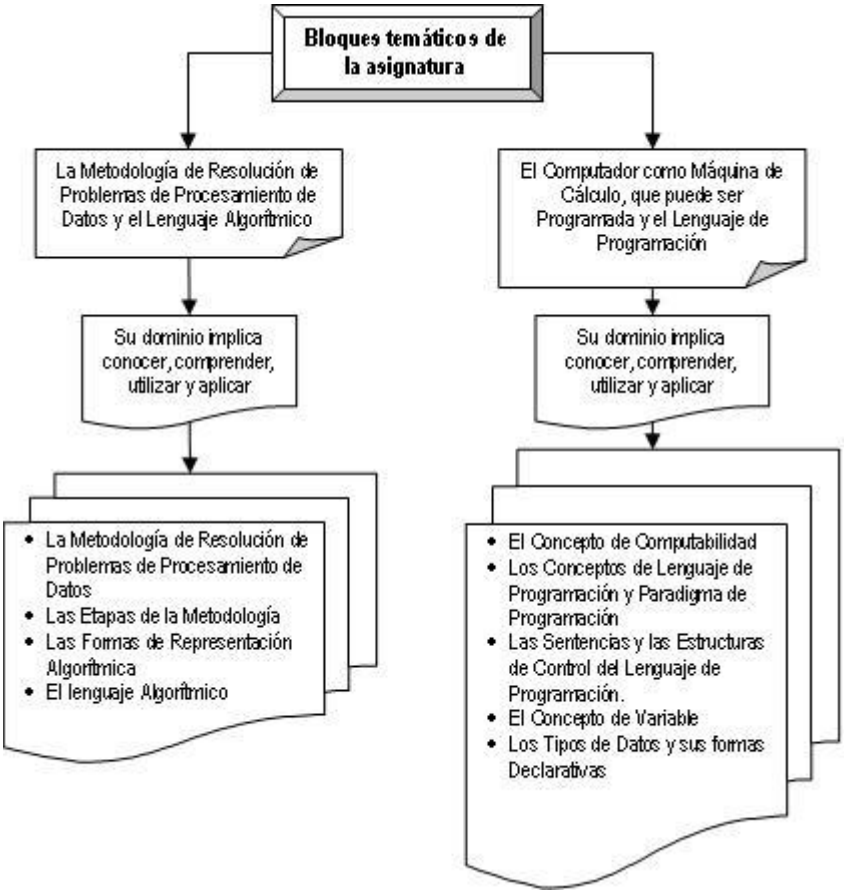

Figura 13: Detalles de la Estructura de Bloques Temáticos de la Asignatura (Caneo,2008)

Dicho en otros términos, no se debe olvidar que uno de los propósitos fundamentales de este primer curso, es que los estudiantes desarrollen las competencias necesarias para resolver problemas de procesamiento de datos, con apego disciplinar, cuestión que conforme a nuestra experiencia y a los resultados de los logros académicos de nuestros estudiantes, es complejo de alcanzar.

Es importante poner énfasis en esto, ya que en el desarrollo de esta primera asignatura, con mucha frecuencia, ya sea por el enfoque que el docente le da a la asignatura, y/o por la percepción o enfoque de aprendizaje que usan preferentemente los estudiantes, se tiende a la instrumentalización del acto mismo de programar computadores, con lo cual el aprendizaje de la asignatura se focaliza sólo en el uso del lenguaje de programación, desconocimiento con ello la importancia y relevancia que tienen los fundamentos teórico-formales y la estrategias de resolución de problemas en las cuales se sustenta la disciplina, cuestión que nos parece de gran importancia, toda vez que como se ha argumentado, esta área disciplinar tiene un carácter nuclear para estos planes de formación, y por tanto, los fundamentos teórico-formales y las estrategias de las cuales hablamos, tienen un rol preponderante dentro de todo el currículum de estos planes de formación profesional.

Ahora bien, en la intención de profundizar en la compresión de lo que implica aprender los fundamentos de la programación de computadores, y desarrollar las capacidades y habilidades para aplicar el proceso completo de resolución de los problemas de programación, en el siguiente apartado se analizan los elementos que se deben considerar, a la hora de tratar con el aprendizaje y el desarrollo de las capacidades que esto implica. Con ello se profundiza también, en la forma cómo estos elementos deben conjugarse para el logro de los propósitos que se han señalado.

En resumen, esta es la primera asignatura con que se aborda el área de conocimientos de FP. Para el caso de la UPLA, es una asignatura de tipo teórico-práctica, que persigue el propósito de que los estudiantes adquieran los conocimientos y las competencias fundamentales, que les permitan disponer de la capacidad para resolver problemas de procesamiento de datos y para programar computadores.

A partir de los objetivos declarados, se identifican dos bloques temáticos, uno relativo a la metodología de resolución de problemas de procesamiento de datos y los algoritmos, y el otro, relacionado con el computador como máquina de cálculo que puede ser programada y el lenguaje de programación como recurso para la construcción de los programas.

Un aspecto importante en el tratamiento de estos contenidos, es que los estudiantes deben lograr el dominio integrado de los conocimientos de los bloques temáticos, lo que significa, que han de lograr la capacidad para aplicar de manera correcta la metodología y el concepto de algoritmo, y complementariamente, deben ser capaces de usar de manera eficiente, los recursos del lenguaje de programación.

Hemos enfatizado también que los estudiantes deben adquirir y utilizar los conocimientos teórico-formales y los procedimentales con apego al carácter disciplinar de la programación de computadores, cuestión que es fundamental de lograr en esta primera asignatura, ya que con ella se están desarrollando los conocimientos y capacidades de base, que se requieren para otras asignaturas de programación que los estudiantes deben cursar posteriormente.

3.2. Elementos a considerar para el aprendizaje de los conocimientos y el desarrollo de las capacidad

Como hemos señalado en apartados anteriores, para la generalidad de los estudiantes, el aprendizaje y el desarrollo de las capacidades para resolver problemas de procesamiento de datos, es complejo y por tanto, difícil de alcanzar. Conforme a nuestra experiencia, creemos que existen cuatro factores que determinan esta complejidad y dificultad:

1.- La forma particular cómo evoluciona el tratamiento de los contenidos que deben ser aprendidos,
2.- Las características de las capacidades que se deben aprender y desarrollar,
3.- Los recursos cognitivos que deben ponerse en juego y,
4.- Las características del propio proceso de enseñanza y aprendizaje.

En el desarrollo de la asignatura y conforme se abordan los contenidos, estos evolucionan desde contenidos que son de un alto nivel de abstracción, hacia contenidos que tienen un alto nivel de concreción y verificabilidad práctica. Conforme a esta perspectiva, los conocimientos que han de aprenderse, transitan desde el aprendizaje de los fundamentos y la base teórico-formal de la disciplina, hacia el aprendizaje de los conocimientos y el desarrollo de capacidades que se relacionan con el uso y aplicación de los fundamentos y la base teórico-formal, a los procesos de resolución de los problemas.

Por tanto, decimos que estos últimos conocimientos son de carácter concreto y verificable en la medida que, se aplican y usan para resolver problemas concretos, y su adecuada y correcta aplicación, puede ser verificada a través de la constatación de que se ha logrado diseñar e implementar como programa, la solución a un determinado problema.

Ahora bien, como se ha señalado y descrito en apartados anteriores, para alcanzar el dominio de estos conocimientos, se requiere aprender y perfeccionar determinadas capacidades y habilidades. Desde una perspectiva general, hablamos de las capacidades y habilidades para comprender los fundamentos teórico-formales y el método de resolución de estos problemas, pero también hablamos de las capacidades y habilidades para aplicar los fundamentos y el método, a los procesos de diseño e implementación de soluciones.

Nos referimos entonces a capacidades y habilidades que en términos de su dominio, transitan desde capacidades y habilidades que se expresan mediante el conocimiento y comprensión de ellos, pero que han de materializarse en capacidades y habilidades referidas a un saber hacer, es decir, capacidades y habilidades para usar y aplicar los conocimientos a la resolución de los problemas.

Para ser más específicos, para resolver estos problemas no basta con que el resolutor conozca y comprenda los fundamentos teórico-formales de la programación, que sepa de los propósitos de cada etapa, y de los conceptos en que se sustentan estos fundamentos, como el concepto de algoritmo, el de lenguaje de programación, y el concepto de computador como máquina programable, sino que además, debe contar con la capacidad y habilidad de usar e integrar estos elementos cuando los aplica en las tareas de resolución de los problemas.

Pero como se ha señalado, esto no es lo único que se requiere. También es necesario que ciertos conocimientos declarativos y procedimentales, generalmente del ámbito de la matemática, puedan ser también integrados con los conocimientos de los fundamentos y la metodología, dando origen al conocimiento estratégico, que es el conocimiento esencial para resolver problemas de este tipo.

Así, para diseñar la solución al problemas de encontrar las raíces que son solución de la ecuación de segundo grado, que citamos como ejemplo en el apartado 1.2.1, se requiere la activación de los conocimientos declarativos y procedimentales que son propios del espacio de este problema, ello fundamentalmente con el propósito de abordar las condiciones de generalización a las cuales deben responder los problemas de procesamiento de datos.

Para una parte de la realización de la tarea de resolución del problema, estos conocimientos se deben integrar con los conocimientos de los fundamentos y la metodología. Pero para realizar esta integración, se requiere que en una instancia anterior, se hayan identificado, seleccionado y organizado el conjunto de conocimientos que son pertinentes al problema particular que se está tratando de resolver.

Estos actos de identificación, selección, organización e integración, continúan siendo demandados en otras partes del proceso de resolución, así por ejemplo cuando se diseña la solución, intervienen los conocimientos del espacio del problema, los fundamentos del concepto de algoritmo y los de la metodología, teniendo en consideración, que la intervención de estos conocimientos tiene que llevarse igualmente al plano del uso y aplicación de ellos.

Consecuentemente, cuando se trabaja en la implementación del programa, nuevamente se requiere de la identificación, selección, organización e integración de conocimientos, en particular los referidos al lenguaje de programación y del concepto de computador como máquina de cálculo.

En estos términos el proceso completo de resolución de estos problemas es un constante devenir de identificación, selección, organización e integración de variados conocimientos, tareas que desde la perspectiva cognitiva, irán generando nuevos conocimientos, aquellos que son propios de la resolución del problema particular, es decir el estratégico, con lo cual, cada problema que se resuelve generará un nuevo conocimiento. La generación de estos nuevos conocimientos estratégicos, es lo que va permitiendo construir el conocimiento de experto del programador, al que se ha hablado en apartados anteriores.

Claramente, estamos hablando de un aprendizaje complejo, que demanda una actividad cognitiva reconocidamente compleja, y que requiere el uso de ciertas capacidades y habilidades intelectuales que son también de carácter complejo.

3.2.1. La cognición en la resolución de los problemas de programación

Los conocimientos de base que se requieren para tratar con la resolución de los problemas de procesamiento de datos, son los declarativos y los procedimentales. Hablamos de dos categorías de conocimientos que el estudiante debe poseer de manera previa, los cuales deberá integrar con conocimientos que ha de aprender en la propia asignatura.

Así por ejemplo, saber qué es un triángulo, qué es una ecuación de segundo grado, qué es un operador lógico, etc., corresponden a conocimientos declarativos que el estudiante debe tener incorporado en su estructura cognitiva.

De igual forma, conocer cómo se calcula un porcentaje, cómo operar con expresiones algebraica y con expresiones lógicas, etc., son ejemplos de conocimientos procedimentales que también deben estar incorporados en la estructura cognitiva del estudiante.

Por su parte, el concepto de algoritmo, el de computador, y los propósitos de cada etapa de la metodología de resolución de los problemas de procesamiento de datos, son ejemplos de conocimientos declarativos que debería adquirir e integrar con el desarrollo de la asignatura. De modo similar, las técnicas de prueba de un programa, las formas de usar un ciclo, y la forma de seguir la traza de un algoritmo, son ejemplos de conocimientos procedimentales que también debería adquirir e integrar con la asignatura.

Aparece aquí un primer elemento que caracteriza la actividad cognitiva que debe darse en este contexto. Nos referimos al aprendizaje de nuevos conocimientos declarativos y procedimentales, pero que junto con ser aprendidos, deben ser integrados con los conocimientos previos que se poseen. Para nosotros, esta integración ha de resultar en una nueva estructura de conocimientos de carácter más específico, en el sentido que se aprenden e integran para la tarea específica de resolver esta clase de problemas.

Ahora bien, el aspecto fundamental de la actividad cognitiva para la resolución de los problemas, es que los recursos cognitivos de los estudiantes han de estar dirigidos al desarrollo de las capacidades y habilidades, para usarlos y aplicarlos, es decir, no basta con que sólo posean estos conocimientos, sino que deben ser capaces de movilizarlos durante la realización de la propia tarea de resolución de los problemas.

Para ello, creemos que el estudiante debe poner en juego un conjunto de recursos cognitivos en cada momento del proceso de aprendizaje. Así, conforme a cómo los contenidos transitan desde lo teórico-formal hacia los procedimientos que definen la aplicación y uso de estos, las estrategias cognitivas que se activan en un primer momento deberán corresponder a aquellas orientadas a la retención y comprensión de contenidos, y hacia el manejo de los procedimientos, en tanto en la medida que los contenidos avanzan hacia los aspectos referidos a la aplicación y uso de los mismos, las estrategias cognitivas que han de activar son de carácter más reflexivo, ámbito en el cual se requiere poner en juego el pensamiento inductivo, deductivo, la creatividad y la intuición.

Descrito esto con mayor detalle, queremos decir que cuando se trabaja en el desarrollo de la etapa de análisis, la capacidad para integrar los conocimientos previos y los nuevos, ha de permitir definir el espacio del problema, y consecuentemente conseguir construir el modelo del problema que se intenta resolver.

Para esto se deben identificar y seleccionar los conocimientos previos que se encuentran implicados en el área del problema, integrándolos con los nuevos, referidos al paso metodológico del análisis, luego de lo cual este conjunto de información (conocimientos) se han de organizar de manera coherente, para elaborar la definición del problema.

Para el caso de la segunda etapa del proceso de resolución, los conocimientos previos y los nuevos se integran para diseñar la solución al problema. Aunque en este ámbito son preferentemente conocimientos de tipo procedimentales, la capacidad de integrarlos se da ahora en términos de la identificación, selección y organización de estos, para dar forma al modelo de solución expresado mediante un algoritmo.

En la tercera etapa intervienen principal conocimientos nuevos, como por ejemplo los conocimientos declarativos y procedimentales referidos al lenguaje de programación. Sin embargo, nuevamente se requiere de la capacidad para identificar, seleccionar, integrar y organizar los conocimientos que son pertinentes a la situación que se resuelve, ya que por ejemplo, no todos los algoritmos que se implementan como programas requieren del uso de estructuras de ciclos, o arreglos, etc. Con esto se logra dar forma al modelo de programa de solución.

Finalmente, en el ámbito de la etapa de prueba y depuración, se requiere de la capacidad para identificar, seleccionar, integrar, y organizar conocimientos declarativos y procedimentales, para establecer el modelo de prueba y depuración del programa, y para ejecutar la prueba y depuración.

Además, el propio proceso de resolución de estos problemas, exige que la actividad cognitiva que hemos descrito en términos del desarrollo de cada etapa, sea evaluada y realimentada de manera permanente, ya que ésta es la estrategia que utiliza el proceso de resolución para determinar, si el propio proceso y el uso de los conocimientos, se está ejecutando de manera apropiada, lo cual significa que, la actividad cognitiva debe ser regulada con miras a establecer si los propios conocimientos están siendo aplicados de manera adecuada.

Hablamos por tanto de la necesidad de que todo el proceso sea regulado por el ejecutor, no sólo para determinar que se está ejecutando de manera correcta el proceso de resolución, sino que en términos del aprendizaje, la regulación la entendemos como el mecanismo que le permite al estudiante, asegurarse que está actuando de manera correcta hacia el logro de los objetivos cognitivos. Este es un aspecto que nos parece fundamental, para el aprendizaje y el desarrollo de las capacidades y habilidades, para usar y aplicar los conocimientos que hemos señalado.

Así, cuando un estudiante aborda las tareas de resolución de un problema de programación, no sólo ha de desarrollar el acto intelectual de transferir conocimientos, identificando cuales conocimientos están implicados en la solución, para luego seleccionarlos, integrarlos y organizarlos, en un diseño e implementación de la solución, sino que además, deberá llevar a cabo tareas intelectuales dirigidas a monitorear y evaluar la efectividad de sus propios actos intelectuales, y conjuntamente evaluar la efectividad que está teniendo el propio proceso de resolución del problema que aborda.

En el esquema que mostramos en la Figura 14, en la página sigiente, sintetizamos los aspectos esenciales de la actividad cognitiva que se lleva a cabo durante el proceso de resolución de problemas de procesamiento de datos.

Desde la perspectiva del proceso de aprendizaje y el desarrollo de las capacidades y habilidades, la puesta en juego de estrategias de control y regulación, demanda la disposición del estudiante para llevar a cabo un aprendizaje de tipo activo, según el cual, más que un mero receptor de información, se involucra de manera efectiva y real en su propio proceso de aprendizaje, todo ello en el marco de la ejecución de las tareas de resolución de los problemas.

En términos más concretos, esto implica un estudiante que al llevar adelante su proceso de aprendizaje, actúa en todo momento de manera reflexiva, analítica y crítica, comprometiéndose de manera activa en las actividades aprendizaje, adoptando un rol de argumentador y cuestionador respecto de cómo aplica los conocimientos aprendidos, y evaluando y validando la validez que tiene dicha

aplicación en términos de la efectividad del proceso de resolución que está llevando a cabo.

Creemos que bajo esta perspectiva de utilización de los recursos cognitivos, se estará favoreciendo no sólo la adquisición e integración de los conocimientos de tipo declarativo y procedimental que son propios de la disciplina, sino que lo más importante, se está favoreciendo la adquisición y el desarrollo del conocimiento estratégico, es decir aquel conocimiento que el estudiante construye, cuando ha sido capaz de llevar a cabo de manera consciente, intencionada y controlada, el conjunto de actos cognitivos dirigidos a identificar y seleccionar cuáles conocimientos declarativos y procedimentales se encuentran implicados en un determinado problema, y además, determinar cómo deberá organizarlos para buscar, diseñar e implementar la solución, pero cuando también, ha desarrollado la capacidad y habilidad para controlar, evaluar y regular la efectividad que conlleva la utilización que hace de sus propios conocimientos y su actividad intelectual.

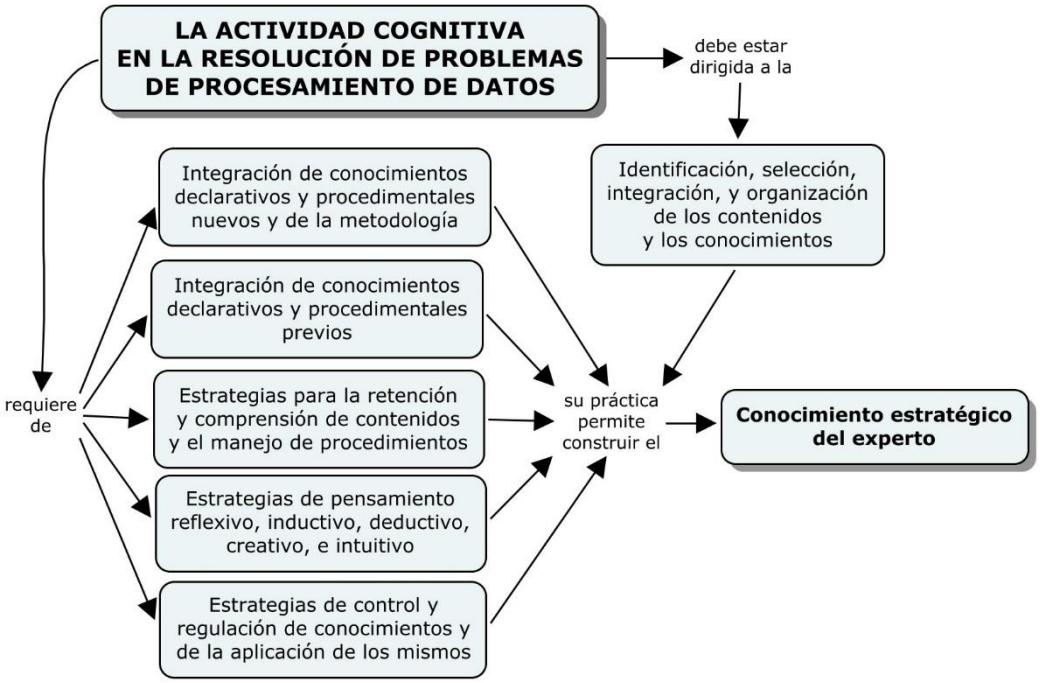

Figura 14: La actividad cognitiva en la resolución de problemas de procesamiento de datos (Caneo, 2010)

Como hemos argumentado en apartados anteriores, es precisamente la adquisición de este tercer tipo de conocimiento (el estratégico), y el desarrollo de la capacidad y habilidad para controlarlo, evaluarlo y regularlo, lo que constituye el rasgo de aprendizaje más difícil de alcanzar para la generalidad de los estudiantes.

En este punto es importante destacar que, a pesar de los importantes descubrimientos que se han hecho desde la psicología cognitiva en estos últimos años, respecto de las capacidades intelectuales generales que entran en juego cuando se lleva a cabo un procesos de resolución de un problema en cualquier

ámbito, aún no es posible determinar cómo operan estas capacidades de manera conjunta e integrada a la hora de ejecutar el proceso de resolución.

Así, si bien ha sido posible señalar cuáles son estas capacidades, con la identificación de ellas, no se ha conseguido proporciona una idea de la prioridad relativa de estas como contribuciones para lograr un desempeño adecuado durante la solución del problema.

Este es un aspecto de la resolución de problemas que actualmente se desconoce, y sobre la cual se trabaja fundamentalmente desde dos ámbitos: desde la inteligencia artificial en términos de poder simular mediante computadores la capacidad de pensamiento humano, y desde el ámbito del propio pensamiento humano.

Nuestra propuesta para abordar esto, es que se deben diseñar actividades y ambientes de aprendizaje, para que los estudiantes cuenten con buenas oportunidades para, aplicar y usar los conocimientos, en las tareas de resolución de los problemas, y junto con ello, puedan también desarrollar, y perfeccionar, las capacidades cognitivas y de resolución de problemas que requieren para tener éxito en la asignatura.

3.2.2. ¿Cómo trabajar la integración de conocimientos y las estrategias de pensamiento en la resolución de problemas de programación?

Atendiendo a la limitación que hemos señalado, y en la idea de clarificar los detalles de la actividad intelectual que debe ocurrir cuando se trata con el proceso de resolución de los problemas de procesamiento de datos, en los siguientes párrafos nos concentraremos fundamentalmente, en ilustrar la forma en que los conocimientos (declarativos, procedimentales y estratégicos) se deben conjugar, a la hora de resolver un problema de este tipo, identificando además, cuales son las capacidades intelectuales que entran en juego.

En esta idea, se analiza cómo dichos conocimientos van participando e integrándose al momento de desarrollar la actividad intelectual que implica el uso de los conocimientos declarativos y procedimentales previos y los propios de la metodología de resolución de los problemas, y el acto de diseñar e implementar el programa de solución.

Para esto efectos, hemos organizado el análisis en términos de la secuencia de los momentos que van ocurriendo en dicho acto intelectual.

3.2.2.1. La integración en la fase inicial

Partamos del supuesto que tratamos con la resolución de uno de los problemas típicos que abordamos en el desarrollo de la asignatura, cuyo enunciado es el siguiente:

> *"Construya un programa que permita determinar si con un conjunto de tres valores de longitudes cualesquiera, se puede formar un triángulo, de ser así, el programa deberá indicar si el triángulo que se puede formar es equilátero, isósceles o escaleno. En caso contrario, el programa deberá indicar que con los valores no se puede forman un triángulo".*

Primero que nada, para considerar que este enunciado constituye un problema para un estudiante, damos por supuesto que él dispone de los conocimientos relacionados con el área y tipo de problema, pero desconoce el algoritmo de solución para el enunciado específico del problema, y desconoce también la implementación del programa correspondiente.

Disponer de los conocimientos del área del problema significa, que maneja los conocimientos declarativos y procedimentales relacionados con el tipo de problema, esto es, maneja el concepto de triángulo y la clasificación de ellos, en base a las magnitudes de sus lados. Además, conoce las reglas que deben cumplir tres magnitudes, para formar un triángulo, y maneja los procedimientos para verificar estas reglas.

Como se recordará, la utilización y aplicación de la metodología de resolución para los problemas de programación, implica la ejecución de un proceso sistemático que considera cuatro etapas; análisis, diseño, implementación, y prueba y depuración. Desde la perspectiva procesal, con la ejecución de cada una de las etapas se construye un producto, el cual constituye el objeto de entrada para la ejecución de la siguiente etapa, hasta llegar al objeto final que es el programa de solución.

En estos términos, para nosotros la característica principal de la actividad cognitiva implicada en la resolución de los problemas, es que ésta debe orientarse a la construcción de diversos objetos, y no a la mera repetición en la aplicación de determinados conocimientos.

Así por ejemplo en la etapa de análisis la actividad cognitiva debe dirigirse a la construcción del modelo del espacio del problema y su especificación; en la etapa de diseño deberá estar dirigida a la construcción del modelo de solución y el correspondiente algoritmo; en la etapa de implementación deberá estar dirigida a la construcción del modelo de programa; y en la etapa de prueba y depuración, deberá estar dirigida al modelo de prueba del programa.

Como consecuencia de lo anterior y desde la perspectiva del aprendizaje, estos actos cognitivos han de permitir la construcción de nuevos conocimientos, que resultan de la práctica de resolución con diversos y distintos tipos de problemas, con lo cual se persigue el propósito de desarrollar y perfeccionar el conocimiento de experto del programador.

Otro aspecto que creemos necesario recordar, es que cuando se trabajar en la resolución de un problema de programación, se debe disponer de la capacidad para utilizar y aplicar conocimientos de dos áreas; los conocimientos referidos al tipo de problema, que para el caso de los tipos de problemas que se abordan en el primer curso de programación, constituyen conocimientos previos que el estudiante debería conocer y manejar, y los conocimientos referidos a la metodología que es propia de la resolución de estos problemas.

3.2.2.2. La integración en la etapa de análisis del problema

En esta etapa, las primeras acciones intelectuales que ha de ejecutar el estudiante, deben estar dirigidas a identificar, seleccionar y combinar las dos clases de conocimientos, los que son propios del espacio del problema, y los referidos a los propósitos y los procedimientos para desarrollar la etapa de análisis, todo ello con miras a elaborar, el modelo del problema que intenta resolver.

En un plano más concreto y conforme al enunciado del problema, estas acciones se han de dirigir a la identificación y selección de los conocimientos respecto de los triángulos como un polígono de tres lados, que conforme a lo que hemos planteado anteriormente, son conocimientos que deberían estar incorporados en la estructura cognitiva del estudiante. Estos conocimientos los debe integrar con los referidos a los fundamentos del proceso de análisis y con los referidos a la definición del procedimiento para construir el modelo del espacio del problema y la especificación del mismo.

Como se ha planteado, desde la perspectiva del proceso de resolución, con la ejecución de esta se persigue el propósito de definir la especificación del problema a resolver. Sin embargo desde la perspectiva de la actividad intelectual, el proceso cognitivo se orienta a la construcción del modelo del espacio del problema.

Para estos efectos, la ejecución de estas dos tareas implica que el resolutor debe poner en juego la capacidad para interpretar y comprender el enunciado, seleccionado, identificando y organizando la información que es relevante, y a partir de ella construir un modelo de representación del espacio del problema, lo que se debe materializar en la especificación del mismo.

Para llevar a cabo estas tareas deberá activar los conocimientos que dispone en relación el área del problema (problema de triángulos), y relacionar esos conocimientos con la interpretación del enunciado del problema, dando así origen a su modelo de problema.

Así, en este acto intelectual ha de identificar, seleccionar y relacionar los siguientes conocimientos declarativos:

- Los triángulos como polígonos compuestos por tres lados,
- Las condiciones para que un conjunto de tres lados forme un triángulo, y
- Las condiciones para que los tres lados formen o un triángulo equilátero, o uno isósceles, o uno escaleno.

Con esto podrá construir el modelo de espacio del problema. Luego de combinar estos conocimientos con los relativos a los fundamentos y los propósitos de la etapa de análisis, el resolutor debe elaborar la especificación del problema, la que para este caso debería resultar en la siguiente:

- Entrada: Los tres valores de magnitud de cada lado.
- Salidas: Mensajes posibles:
  - ♦ Uno más valores no corresponde, caso que ocurre cuando uno o más de los valores es menor o igual a cero.
  - ♦ El conjunto de valores no permiten formar un triángulo.
  - ♦ El conjunto de valores forma un triángulo equilátero.
  - ♦ El conjunto de valores forma un triángulo isósceles.
  - ♦ El conjunto de valores forma un triángulo escaleno.

La construcción del modelo y la elaboración de esta especificación, para nosotros, involucra la ejecución de varios actos intelectuales por parte del resolutor. Por una parte, ha de tener en consideración los propósitos de la primera etapa de la metodología y los detalles del procedimiento para llevarla a cabo, y por otra, deberá interpretar y comprender el enunciado del problema para identificar y seleccionar los elementos que lo caracterizan, y de esta forma ir construyendo el modelo que señalamos, lo que finalmente se debe traducir en la especificación que hemos descrito.

Mientras va ejecutando estos actos, se espera que vaya monitoreándolos con el propósito de revisar y verificar que los esté ejecutando correctamente. Esta verificación tiene dos aspectos, la verificación del proceso y la verificación del producto. El proceso tiene que ver con la correcta aplicación de los conocimientos y del paso metodológico, en tanto el producto, tiene que ver con el resultado de la aplicación de ellos.

Para ilustrar la importancia que tiene este proceso de verificación, podemos dar un ejemplo de uno de los errores típicos que cometen nuestros estudiantes en la

construcción del modelo del espacio del problema y consecuentemente en la especificación del mismo. Es habitual que los estudiantes tanto en la definición del modelo como en la especificación establezcan que se requiere como datos de entrada los valores de los ángulos interiores del supuesto triangulo, lo que sería erróneo ya que en el enunciado del problema se establece que se debe tratar con los lados. Otro de los errores habituales es considerar que los valores de los supuestos lados de un triángulo, pueden corresponder a valores negativos o cero. Estos errores deben ser detectados al efectuarse de manera correcta la verificación.

Para nosotros, este proceso de verificación implica el uso de estrategias de control y regulación dirigidas a constatar si la forma en que se están usando y relacionando los distintos tipos de conocimientos en el contexto del problema particular que se aborda, es correcta.

Las dificultades que tienen los estudiantes al desarrollar esta primera etapa, se relacionan con la identificación y selección de los conocimientos que están implicados en la solución.

Esta dificultad provocará que ellos, cometan errores importantes en la especificación del problema, es decir, en establecer los datos de entrada, los resultados que debe producir la solución del problema, y las restricciones que deben considerarse para el diseño de la misma, todo ello atendiendo a las características de generalización que deben poseer las soluciones a estos problemas.

Respecto de lo anterior, se observa que los estudiantes al tratar con los problemas, con frecuencia piensan en casos particulares del mismo.

3.2.2.3. La integración en la etapa de diseño de la solución

Para desarrollar la segunda etapa, el resolutor deberá utilizar el modelo de espacio del problema y la especificación que elaboró en la etapa anterior, para ahora llevar a cabo la tarea de identificar, organizar y describir los procesos que actuaran sobre los datos de entrada, para producir los resultados de salida establecidos en la especificación, procesos que además, deben responder a las restricciones establecidas en la misma especificación. Así, con este conjunto de elementos ha de trabajar en el diseño de la solución al problema.

Desde la perspectiva de la actividad cognitiva, en esta etapa debe construir el modelo de solución del problema.

En estos términos, su actividad intelectual ha de dirigirla hacia la identificación y selección de los conocimientos procedimentales relacionados con el espacio del problema.

En el paso siguiente, deberá organizar los procesos como una secuencia de acciones y operaciones, que permitan tratar con los datos de entrada, para producir los resultados establecidos en el modelo del espacio del problema y su especificación.

Para esto, ha de llevar a cabo un acto intelectual constructivo, que implica la integración de varios conocimientos (los previos y los nuevos), con el propósito de diseñar el modelo de solución al problema. Sin embargo, este acto intelectual no sólo debe ser constructivo, sino que debe ser también evaluativo, ya que mientras elabora el diseño de la solución, ha de evaluar tanto la pertinencia de los conocimientos que utiliza y su organización, como también, la validez del proceso de solución que está ejecutando.

Estos actos intelectuales son los responsables de la construcción del conocimiento estratégico, aquel tipo de conocimiento que el resolutor va elaborando mientras trabaja en el diseño de la solución. Es el conocimiento que resulta de las acciones intelectuales que ha llevado a cabo para identificar, seleccionar, combinar y organizar, aquellos conocimientos declarativos y procedimentales que son pertinentes al problema que está tratando de resolver.

Pero esta construcción de conocimiento estratégico es efectiva, si junto con identificar, seleccionar, combinar y organizar los conocimientos declarativos y procedimentales, se ejecutan también acciones intelectuales de regulación y control respecto de la validez de los conocimientos que se utilizan y del proceso que se sigue, para elaborar la solución al problema.

Como hemos señalado, este es el aspecto más complejo y crítico del proceso de aprendizaje y el desarrollo de las capacidades para resolver problemas de programación, ya que exige que los estudiantes se involucren de manera activa y participativa en el proceso de resolución de los problemas, actuando en todo momento como cuestionadores y argumentadores, respecto de la validez de los conocimientos que se implican y del proceso que se sigue para utilizarlos en el diseño de la solución.

Nuestra visión es que para el logro de estos propósitos, los profesores debemos crear ambientes y situaciones que sean propicios para el logro de estos aprendizajes, hablamos de ambientes y situaciones que posibiliten y favorezcan que los estudiantes, junto con tratar con el aprendizaje para la construcción de las soluciones, durante los procesos de resolución de los problemas, actúen como cuestionadores y argumentadores, y que de esta forma favorezcan sus aprendizajes, y traten de manera activa, con el desarrollo y la puesta en juego de las estrategias cognitivas que se requieren para la ejecución de los actos, y tareas intelectuales que hemos descrito.

Desde el punto de vista cognitivo, disponer de estas capacidades implica que el sujeto está en condiciones de llevar a cabo el acto cognitivo de construir el conocimiento estratégico, aquel conocimiento que resulta de la adecuada identificación, selección, combinación y organización de los conocimientos que se encuentran implicados en la situación problema, y a partir de esto, elaborar y evaluar propuestas de solución.

Por tanto, realizadas estas tareas, en el paso siguiente, se debiera obtener el modelo de solución del problema el cual se materializa en la organización y descripción de la consecuencia de acciones y operaciones que permitirán establecer una solución para el problema, la cual podría corresponder a la siguiente:

- Ingresar los tres valores que representan la magnitud de cada supuesto lado del triángulo.
- Verificar que los valores sean mayores que cero. Si esto se cumple, continuar con el paso siguiente, en caso contrario mostrar un mensaje que indique que uno o más valores no corresponden, y terminar.
- Verificar que para las tres combinaciones posibles se cumple que la suma de dos de los valores, es siempre mayor que el tercero. Si esto se cumple, continuar con el paso siguiente, en caso contrario, mostrar el mensaje “las magnitudes no permiten forman un triángulo”, y terminar.
- Verificar si las tres magnitudes son iguales. Si esto se cumple mostrar el mensaje “Pueden formar un triángulo equilátero” y terminar. En caso contrario, pasar al paso siguiente.
- Verificar si existen a lo menos dos magnitudes iguales. Si esto se cumple mostrar el mensaje “Pueden formar un triángulo isósceles” y terminar. En caso contrario, pasar al paso siguiente.
- Mostrar el mensaje “Pueden formar un triángulo escaleno” y terminar.

Esta segunda etapa termina con la expresión de la secuencia anterior mediante un lenguaje algorítmico, para lo cual el estudiante debe combinar el conocimiento declarativo y el procedimental referido a la forma de representación algorítmica, junto con el conocimiento estratégico que ha construido para diseñar la solución.

La representación algorítmica debe expresarse como diagrama de flujo y como seudocódigo. La representación como diagrama de flujo, por corresponder a una forma esquemática, facilita la visualización, el análisis, y el seguimiento de las secuencias de acciones, facilitando de paso, la verificación del diseño de la solución.

En tanto la representación en seudocódigo, por ser cercana a la estructura que utilizan los lenguajes de programación declarativos, facilita el paso siguiente de la metodología, que es la implementación del programa.

Las dificultades para el desarrollo de esta etapa se relacionan en primer término, con la identificación de los procesos y su organización, como una secuencia lógica, que al actuar sobre los datos de entrada, permitan obtener los resultados establecidos en la especificación del problema, atendiendo además a las restricciones del mismo.

También se aprecian dificultades relacionadas con el uso del lenguaje algorítmico para la representación y descripción del modelo de procesos, usando diagramas de flujo y seudocódigos.

Son dificultades asociadas con el correcto uso de las palabras claves, para el caso de los seudocódigos, así también, como dificultades para organizar estas representaciones como estructuras coherentemente lógicas, a la luz del tipo y características del modelo de procesos que se está representando.

En la siguiente figura 15 de la siguiente página, se muestra la representación del modelo de proceso como algoritmo en diagrama de flujo, para el problema que tratamos como ejemplo.

En tanto el algoritmo expresado en seudocódigo se debe corresponder con el siguiente:

```
inicio
leer A, B, C
si A>0 y B>0 y C>0 entonces
  si A+B>C y B+C>A y A+C>B entonces
  si A=B y B=C entonces
        escribir "pueden formar un triángulo equilátero"
   sino
        si A=B o B=C o A=C entonces
        escribir "pueden formar un triángulo isósceles"
     sino
       escribir "pueden formar un triángulo escaleno"
     fin_si
   fin_si
 sino
   escribir "las magnitudes no permiten formar un triángulo"
 fin_si
sino
 escribir "uno o más valores no corresponden"
fin_si
fin
```

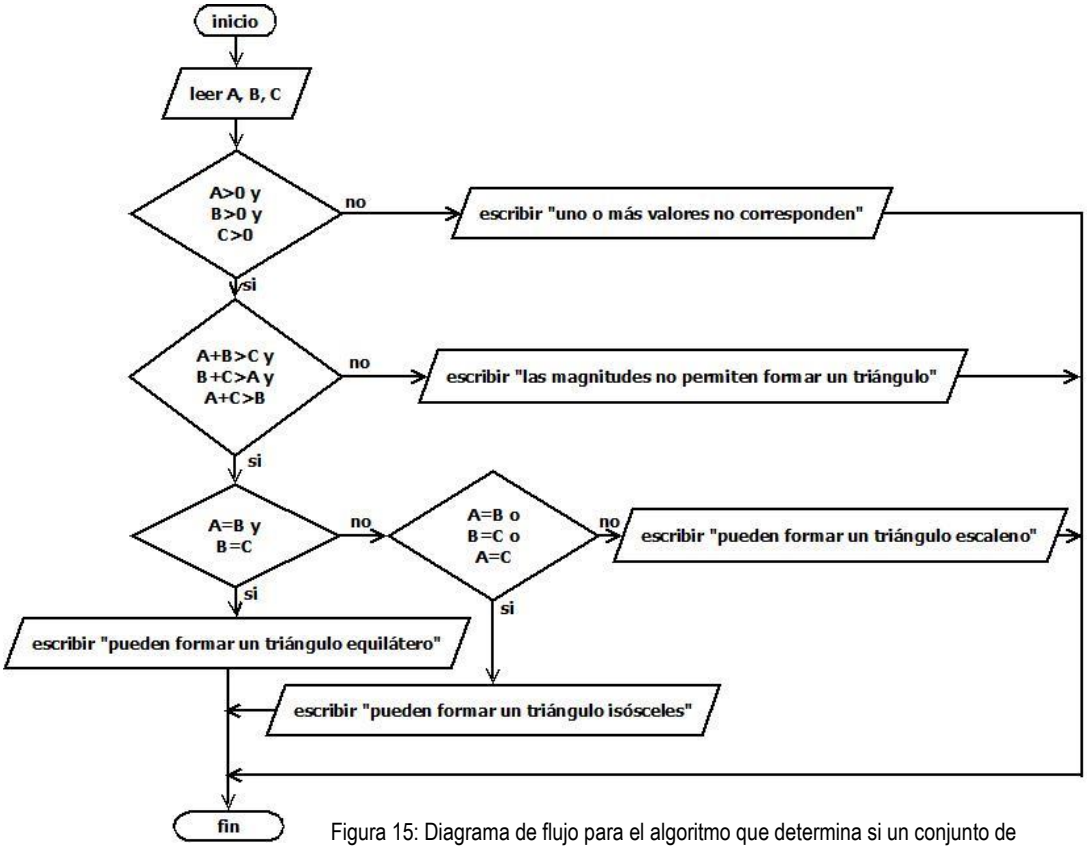

Figura 15: Diagrama de flujo para el algoritmo que determina si un conjunto de tres lados forma triangulo (Caneo 2010)

Nuevamente, el estudiante deberá poner en juego estrategias que le permitan verificar si las construcciones que ha realizado son correctas. Deberá constatar si está usando correctamente el conocimiento declarativo y procedimental tanto en lo referido al espacio del problema con que trata, como en lo referido a la etapa de la metodología que le corresponde.

A nivel de producto, debe verificar si el modelo de solución que esta expresado en el algoritmo, es correcto para el problema y para la especificación que elaboró en la primera etapa. Para realizar esta verificación, ha de usar la técnica del “ruteo” o “seguimiento” de la ejecución del algoritmo, proceso que se ve facilitado al realizarlo sobre el DF.

Reiteramos la idea de que si, para abordar estos aprendizaje, se diseñan situaciones y ambientes, dónde los estudiantes tengan buenas oportunidades para practicar la ejecución de estas tareas intelectuales de resolución de los problemas, se contará con mejores condiciones para ellos puedan alcanzar aprendizaje más efectivos.

3.2.2.4. La integración en la etapa de implementación del programa

El siguiente paso metodológico, consiste en utilizar un lenguaje de programación para implementar el algoritmo en seudocódigo anterior, como un programa computacional. Para ello el resolutor deberá combinar el conocimiento declarativo y

procedimental referido al lenguaje de programación con el modelo algorítmico en seudocódigo de solución.

Para ello deberá primero, identificar y seleccionar los conocimientos declarativos y procedimentales referidos al lenguaje de programación y luego integrarlos con los del modelo algorítmico de solución. El resultado de esta integración es la base para construir el modelo del programa de solución al problema.

En este contexto, ha de identificar las variables y las constantes que requiere la implementación, las sentencias del lenguaje, y las formas declarativas de las acciones y operaciones que deberá utilizar, así como la organización de las mismas dentro de las estructuras del programa, con el objeto de que este concuerde con el modelo de solución algorítmica. En términos más específicos, habrá de identificar, seleccionar y combinar conocimientos relacionados con:

- Los fundamentos y propósitos de la tarea de convertir un algoritmo en un programa,
- La forma de estructurar los programas en el lenguaje de programación escogido.
- Las sentencias e instrucciones del lenguaje de programación.
- El proceso de traducción de los pasos y operaciones del algoritmo en un programa,
- El conceptos de computador y programa,
- El concepto de lenguaje de programación,
- Los conceptos de variable y sus formas declarativas,
- Formas declarativas de las sentencias para la definición de acciones y/u operaciones,

Con la ejecución de esta actividad intelectual, el modelo del programa de solución debería generar el código en lenguaje C que se muestra al comienzo de la siguiente página:

Las dificultades que tienen los estudiantes para llevar a cabo esta etapa, están referidas fundamentalmente al uso del lenguaje del lenguaje de programación.

En términos más específicos, presentan dificultades en la declaración de las variables, en el manejo de las instrucciones y en la organización de instrucciones como estructuras del programa. Este último tipo de dificultades pueden hacerse más severas, cuando tratan con la implementación de seudocódigos que poseen varias estructuras y anidaciones de ellas, como es el caso de los ciclos.

```
#include<stdio.h>
main()
{float a,b,c;
 scanf("%f %f %f",&a,&b,&c);
 if((a>0)&&(b>0)&&(c>0))
  if((a+b>c)&&(b+c>a)&&(c+a>b))
   if((a==b)&&(b==c))
     printf("pueden formar un triángulo equilátero");
   else
     if((a==b)||(b==c)||(a==c))
       printf("pueden formar un triángulo isósceles");
     else
       printf("pueden formar un triángulo escaleno";
  else
   printf("las magnitudes no permiten formar un triángulo");
 else
  printf("uno o más valores no corresponden");}
```

3.2.2.5. La integración en la etapa de prueba y depuración del programa

Con la última etapa de la metodología se persigue el propósito de probar y depurar el programa de solución. Para su ejecución, el resolutor nuevamente ha de integrar conocimientos declarativos y procedimentales, tanto los referidos al paso metodológico, como los referidos a las técnicas de prueba de programas, para así, construir el modelo de prueba del programa.

Para estos efectos deberá tratar con los conocimientos referidos a:

- Los propósitos de la prueba y depuración de programas,
- El procedimiento de prueba y depuración de programas,
- El proceso de compilación de un programa,
- El proceso de ejecución de un programa,
- La elaboración de casos de prueba,
- El proceso de detección e identificación de errores de un programa, y
- Técnicas de prueba y depuración de un programa.

Para probar el programa, el resolutor ha de ejecutar el acto intelectual de combinar y organizar estos conocimientos para construir el modelo de prueba del programa. Uno de los aspectos esenciales de este modelo, es establecer los casos o situaciones de prueba que son adecuados para evaluar la solución.

Para estos efectos, se deben diseñar casos de prueba que permitan verificar si el programa, produce los resultados esperados, y si responde de manera adecuada a las restricciones definidas en la especificación hecha en la etapa de análisis del problema, como por ejemplo, qué resultados deberá producir el programa, para valores de magnitudes negativas o cero.

También se han de diseñar casos de prueba que permitan ver, cómo responde el programa para otros valores de magnitudes de un supuesto triángulo, con lo cual, se diseñan casos de prueba para verificar que la solución responda de manera adecuada, a las condiciones de generalidad de la misma, es decir para cualquier conjunto de tres valores de magnitudes, que podrían o no, formar un triángulo.

La elaboración de los casos de prueba, requiere en primer término la integración de todos los modelos que se han construido durante el proceso de resolución del problema.

Se han de integrar las especificaciones de las entradas, salidas y restricciones del problema, las cuales se establecieron en el modelo del espacio del problema (etapa de análisis), junto con el modelo de solución del problema (etapa de diseño de la solución), expresado en el algoritmo, además del modelo del programa de solución (etapa de implementación del programa).

Dadas las características del proceso de resolución de estos problemas, la actividad de prueba del programa, no sólo debe ser entendida como una prueba al producto de la solución, sino que es una prueba que apunta a verificar la validez del proceso completo que se ha seguido hasta llegar al programa de solución. Desde la perspectiva del proceso de aprendizaje, la podemos entender también como la constatación de que el uso y aplicación de los conocimientos, las estrategias y el acto intelectual de diseñar la solución, han sido realizados de manera correcta.

Lo anterior implica que cuando la prueba del programa da cuenta de la existencia de errores, el resolutor deberá en primer término, identificar el o los errores, para luego establecer, en que parte o partes del proceso de resolución se ha producido el error, y trabajar sobre su análisis y corrección.

Desde el punto de vista del aprendizaje de la programación, la prueba y depuración de programas es especialmente compleja para la generalidad de los estudiantes, ya que los errores pueden producirse por varias razones, dentro de las que podemos identificar una categoría de errores que se deben a la dificultad inherente de un cierto problema en particular, y que por tanto, podrían ser errores que también podría cometer un experto en programación, y una segunda categoría compuesta por aquellos errores debidos a una condición de inmadurez en el aprendizaje, los que podrían deberse a un inadecuado dominio y/o uso, ya sea de los conocimientos, o de las estrategias cognitivas, o de ambos.

Cuando los errores pertenecen a esta segunda categoría, tanto el proceso de prueba como la depuración de los programas se hacen aún más complejos.

En este contexto, el aspecto de más complejo para la mayoría de los estudiantes, es la creación de los casos de prueba, para validar la eficacia del programa de solución.

Esta tarea es compleja para los estudiantes, porqué les demanda la integración de todos los modelos y visiones que se han construido y utilizado, durante el proceso de resolución del problema.

Sin embargo, nuestra experiencia nos muestra que en la medida en que los estudiantes tienen oportunidades para socializar, argumentar y defender sus soluciones, se hace más fácil determinar si los errores son de la primera o segunda categoría, o de ambas, al tiempo que se clarifica y facilita el aprendizaje, y la ejecución y la de las tareas de diseño de casos de prueba, y la propia prueba y depuración de programas que hemos señalado.

3.2.2.6. Condiciones para el aprendizaje de la PC

Como se puede deducir del análisis que hemos hecho hasta este punto, las capacidades intelectuales que se requieren para resolver problemas de procesamiento de datos y para programar computadores corresponden a capacidades de orden esencialmente prácticas, es decir son capacidades cuyo aprendizaje se ha de manifestar en términos de un saber hacer.

Cuando un estudiante se enfrenta a la resolución de un problema de programación, esperamos que en primer término disponga y maneje un conjunto de conocimientos previos, y en segundo término, que conozca y comprenda el conjunto de conocimientos que conforman la base teórico-formal de la disciplina y los procedimientos que implican la puesta en práctica de la metodología de programación, para luego a partir de esto, desarrollar la capacidad de utilizar y aplicar estos conocimientos mientras ejecuta el proceso de resolución.

Ahora bien, para que nuestros estudiantes aprendan y desarrollen estas capacidades deben estar en posesión de ciertas estructuras cognitivas, en las cuales debieran estar organizados los conocimientos declarativos y los conocimientos procedimentales de base, los cuales para el caso específico del problema que citamos como ejemplo, están constituidos por los conocimientos declarativos respecto de los triángulos como polígonos formados por tres lados, y los elementos que determinar su forma de clasificación, en tanto en el ámbito de los conocimientos procedimentales de base, los estudiantes deben conocer el procedimiento para verificar si un conjunto cualquiera de tres magnitudes pueden formar un triángulo, y el procedimiento para determinar el tipo de triángulo que se formaría.

Además de estos conocimientos de base, nuestros estudiantes también deben incorporar en sus estructuras cognitivas, los conocimientos declarativos que son

propios de la disciplina, tales como los fundamentos teórico-formales de la metodología que se aplica para resolver estos problemas, y los fundamentos teórico-formales relacionados con la programación de computadores, el lenguaje de programación y el entorno de programación. Junto con esto, ellos deben apropiarse también de los conocimientos referidos a los procedimientos y las reglas para utilizar y aplicar estos conocimientos declarativos.

Sin embargo, como hemos comentado anteriormente, la sola posesión de estos conocimientos, no garantiza que el estudiante sea capaz de diseñar una solución al problema, ya que para ello ha de disponer de la capacidad para identificar, seleccionar, combinar y organizar estos conocimientos, como paso esencial para diseñar la solución. Conforme a este escenario, entra en juego el tercer elemento al cual hemos hecho mención, es decir, las características del proceso de enseñanza y aprendizaje según el cual podemos lograr que nuestros estudiantes consigan aprender y desarrollar esta capacidad.

El esquema que presentamos en la Figura 10, es válido también para ilustrar como los distintos tipos de conocimientos y las estrategias, van interviniendo en los distintos momentos del desarrollo del acto intelectual que estamos describiendo.

No cabe duda que dicho acto intelectual es de carácter complejo, en función de lo cual, consideramos que es muy importante que la forma de conducir el proceso de aprendizaje, se constituya en el mecanismo para crear las condiciones para el aprendizaje, el desarrollo y el perfeccionamiento de las capacidad y habilidades que demanda la ejecución del mismo.

Hablamos de un proceso de enseñanza y aprendizaje que se constituya en un espacio de práctica intencionada y sistemática, que posibilite el aprendizaje y el desarrollo de las capacidades, es decir un espacio que demande el uso y aplicación de los conocimientos implicados, así como también el desarrollo de las estrategias cognitivas que exige la tarea.

Para esto, este proceso debe ser un espacio de oportunidades donde los estudiantes sean puestos en situaciones que les permitan aprender y desarrollar las estrategias cognitivas que son de utilidad para estos propósitos, y que complementariamente, les demanden la utilización y puesta en práctica de las mismas, creando también las condiciones para que regulen y evalúen el aprendizaje, el uso y los resultados que están logrando con la aplicación de las estrategias.

Ahora bien, en la conducción del proceso de enseñanza y aprendizaje, no se debe perder la perspectiva de que si bien, el tratamiento de los contenidos transita desde los fundamentos teórico-formales (contenidos abstractos), hacia los contenidos de

carácter más concretos y verificables, nuestros estudiantes deben desarrollar la capacidad de integrar los conocimientos abstractos con los concretos y verificables.

Consideramos que este es un aspecto fundamental para el logro de los objetivos de la propia asignatura, pero también, para construir los conocimientos de base de la disciplina, condición que además es determinante para el éxito de los estudiantes en otras asignaturas, que poseen una relación disciplinar directa con este primer curso de la línea de programación.

Destacamos esto puesto que, conforme a nuestra experiencia, lograr un aprendizaje integral de estos conocimientos, no es una tarea trivial para los estudiantes, ya que con mucha frecuencia es posible verificar que cuando resuelven un problema, intentan construir la solución operando directamente con las sentencias e instrucciones del lenguaje de programación, desconociendo o dejando de lado por completo, la utilización y aplicación de los formalismos metodológico-disciplinares para el diseño, implementación y prueba de las soluciones.

Ahora bien, ¿cuál debería ser la modalidad de aprendizaje que nos permita abordar estos contenidos que transitan desde la abstracción teórico-formal, hacia los elementos de tipo procedimental de carácter concreto, y que son de aplicación práctica y verificable en el proceso mismo de resolución de problemas de procesamiento de datos, y que de manera complementaria favorezca la adquisición, el desarrollo, y la utilización de las capacidades cognitivas que son apropiadas para la adquisición de los conocimientos y las capacidades que demanda la actividad de programación?.

Conforme a lo que proponemos y hemos desarrollado en el marco de esta investigación, esta modalidad se debe corresponder con un modelo de enseñanza y aprendizaje multiestructurado, según el cual se integran de manera armónica las sesiones de clases expositivas, además de la puesta en práctica de un conjunto organizado de actividades de aprendizaje cooperativo.

El propósito de las sesiones de clases expositivas, es presentar a los estudiantes los aspectos teórico-formales de la disciplina y ejemplos de algunos problemas típicos de procesamiento de datos, con lo cual, se presenta a los estudiantes una primera aproximación a los aspectos de orden práctico, que conlleva la aplicación de los fundamentos teórico-formales y procedimentales, y se discute y reflexiona respecto a cómo se seleccionan y relacionan estos fundamentos, para construir el conocimiento estratégico, según el cual se busca, diseña y prueba la solución.

De esta forma entonces, se presenta y revisan los aspectos prácticos de cómo, los elementos formales y procedimentales de la teoría y la metodología de resolución

son usados y aplicados en el proceso de búsqueda, diseño, implementación y la prueba de las soluciones y el correspondiente programa computacional.

Tradicionalmente las sesiones de clases expositivas están orientadas fundamentalmente, a la intención del docente de presentar y comunicar información, por tanto, no constituyen una instancia que brinde oportunidades para que los estudiantes tengan una práctica directa en el proceso de resolución de los problemas, con lo cual tampoco tienen las oportunidades para construir, desarrollar y probar las estrategias cognitivas que posibiliten el aprendizaje de la práctica misma de resolución de problemas.

Desde este escenario surge el comentario habitual de los docentes de la asignatura, en el sentido de que, aunque a los estudiantes se les presenten las soluciones detalladas a muchos problemas, ellos no aprenderán a aplicar de manera autónoma los conocimientos y el proceso de resolución.

De manera similar, si ellos al realizar su estudio personal con el objeto de preparar por ejemplo, una evaluación, sólo se limitan a revisar las soluciones ya construidas, ya sea en las clases, en los textos de estudios, en registros de evaluaciones anteriores, etc. no conseguirán alcanzar el aprendizaje práctico del que estamos hablando.

En este sentido, existe consenso en el mundo académico respecto de que las competencias para la programación de computadores se aprenden y desarrollan con la práctica directa, e intencionada dirigida a la construcción y prueba de las soluciones a los problemas de procesamiento de datos, y poniendo énfasis en la aplicación y el desarrollo del proceso mismo de solución.

Por tanto, es en este contexto donde proponemos que el aprendizaje y el trabajo cooperativo como organización de actividades de aula, pase a constituirse en la instancia formal e intencional para crear y llevar a la práctica, tanto el aprendizaje de la disciplina, como el aprendizaje, el desarrollo y la puesta a prueba de las capacidades y habilidades cognitivas, que la tarea misma de resolución de los problemas demanda, todo esto basados en el hecho de que, el entorno de aprendizaje y las actividades de trabajo cooperativo, crean las condiciones para un aprendizaje socializado, en el cual el intercambio de experiencias de aprendizaje se posibilitan y favorecen, y lo más importante, actúan como una instancia que potencia y enriquece el proceso mismo de aprendizaje.

Es en este entorno donde creamos las condiciones para el aprendizaje práctico del que hemos hablado, el cual en este ambiente, esta mediado por el acto mismo de la discusión, la defensa y la reflexión colectiva respecto del uso y aplicación de los conocimientos disciplinares y las propuestas de soluciones, y de paso se constituye

en un espacio para aprender a utilizar y aplicar el proceso de diseño, implementación y prueba de soluciones a los problemas, acto en el cual unos aprende de otros, y donde además, aprenden, y tienen oportunidades para desarrollar y poner a prueba, las capacidades específicas requeridas para el logro de los aprendizaje de la disciplina.

Este ambiente nos provee también, las condiciones para la práctica que implica aprender y desarrollar la capacidad para identificar, seleccionar, combinar, y organizar los distintos tipos de conocimientos que se encuentran implicados en una determinada solución.

En consecuencia, es en este ambiente de práctica y reflexión permanente, compartida y de retroalimentaciones mutuas, donde se dan las mejores condiciones y oportunidades para que los estudiantes desarrollen y perfeccionen las estrategias cognitivas y de pensamiento que se requieren para la tarea de resolver estos problema, lo que irá de la mano con el desarrollo de los actos intelectuales dirigidos a la construcción de aquel tercer tipo de conocimiento del que hemos hablado, el conocimiento estratégico.

En la medida en que se discute, se defiende, se reflexiona, se argumenta, y se evalúan las propuestas de soluciones que cooperativamente se van proponiendo y construyendo, se está favoreciendo el desarrollo y uso de las estrategias cognitivas y de pensamiento, junto con la construcción del conocimiento estratégico, ya que al llevar a cabo estos procesos cooperativamente, todos se están haciendo participes activos de las prácticas que esto conlleva.

Por otra parte, como un entorno de aprendizaje de estas características se da fundamentalmente en un entorno social, esta práctica contribuye también al aprendizaje autónomo de los estudiantes, ya que cada estudiante es demandado a participar con sus ideas y aportaciones.

Bajo la idea del trabajo cooperativo, se insta a que cada estudiante se involucre de manera activa en el proceso de búsqueda e implementación de soluciones, actuando como argumentador, defensor, crítico y evaluador de las estrategias de solución personales y las que cooperativamente se van aplicando, así como también respecto de las soluciones mismas que se van construyendo.

Con esta modalidad no sólo estamos creando mejores condiciones para el aprendizaje de los contenidos, sino que además estamos favoreciendo las condiciones para aplicar y evaluar el uso que se hace de ellos, en la práctica misma del proceso de resolución de problemas, todo esto con el propósito de alcanzar el aprendizaje práctico que demanda el logro de las competencias para la programación de computadores.

Con estos argumentos, defendemos la inclusión de las prácticas de aprendizaje y trabajo cooperativo como el vehículo que nos permite favorecer el logro de las competencias de orden práctico que demanda la asignatura, y que al mismo tiempo nos permite mejorar el aprendizaje y desarrollo de determinadas estrategias cognitivas específicas, que se requieren para el mismo aprendizaje, cuestiones que además son relevantes no sólo para la asignatura misma, sino que son también de alta importancia, para todas las asignaturas que se encuentran en directa relación con el área disciplinar de la programación de computadores de los planes de formación profesional en computación.

Como síntesis, de lo que hemos expuesto en este apartado, podemos decir que para nosotros, las dificultades que tienen los estudiantes con el aprendizaje de las capacidades para resolver problemas de procesamiento de datos, están determinadas por los siguientes cuatro aspectos:

1.- La forma particular cómo evoluciona el tratamiento de los contenidos que deben ser aprendidos,
2.- Las características de las capacidades que se debe aprender y desarrollar,
3.- Los recursos cognitivos que deben ponerse en juego, y
4.- Las características del propio proceso de enseñanza y aprendizaje.

Como se ha analizado, para resolver los problemas, no es suficiente conocer y comprender los fundamentos teórico-formales y los procedimientos de la disciplina, sino que además se debe desarrollar la capacidad de usar e integrar estos tipos de conocimientos.

Las formas de usar e integrar los conocimientos cuando se trata con diversos tipos de problemas, da origen a lo que hemos llamado el conocimiento estratégico, y resulta de la aplicación y uso correcto, de los demás tipos de conocimientos, lo que hace de esta una tarea compleja, que para nosotros está condicionada, por la práctica directa y efectiva en las tareas de resolución.

Así la actividad cognitiva en la resolución de los problemas implica varios procesos (ver Figura 14 en página 173), y su desarrollo exige que el aprendiz actúe en todo momento de manera reflexiva, analítica y crítica, lo que conlleva que el estudiante este comprometido con su aprendizaje, adoptando el rol de argumentador y cuestionador respecto de cómo aplica los conocimientos aprendidos.

De esta forma se hará consciente de los recursos y procesos cognitivos que usa y aplica, y de la efectividad que ello tiene para el propio proceso de resolución de los problemas.

Así se estará trabajando también de una forma más efectiva, en la construcción del conocimiento estratégico, que como hemos señalado, es el tipo de conocimiento que

opera cuando se seleccionan, organizan e integran los distintos tipos de conocimientos, al trabajar en la resolución de un problema específico.

Ante estas complejidades, enfatizamos que el proceso de enseñanza y aprendizaje debe constituirse en un espacio de práctica intencionada y sistemática, que permita el aprendizaje y el desarrollo de las capacidades que hemos señalado.

Creemos que una organización de la asignatura conforme a un plan de clases multiestructurado en el que se integren las clases expositivas y las prácticas de resolución de problemas mediante actividades de aprendizaje cooperativo, contribuirán a desarrollar mejores aprendizaje en los estudiantes.

3.3. Principios para el aprendizaje de la PC

Tomando en consideración el tipo de contenidos que se deben aprender, la complejidad de las capacidad que se deben desarrollar, la caracterización de los aprendizajes, así como los elementos que se deben tener presente para la adquisición de las capacidades que hemos analizado en los apartados anteriores, presentamos ahora, los principios que a nuestro juicio, debieran orientar el proceso de aprendizaje en el curso de FP. En estos términos, consideramos que el proceso de aprendizaje debe estar dirigido a:

1.- Conocer y comprender de manera significativa los fundamentos teórico-formales de la disciplina.
   - Como hemos enfatizado en apartados anteriores, el aprendizaje de los fundamentos de la programación de computadores, debe llevar implícito el aprendizaje de los fundamentos teórico-formales de la disciplina, atendiendo a la consideración de que, estamos tratando con una asignatura que ha sido considerada como troncal en los planes de formación profesional en CcC, y que además, todos los contenidos del área de conocimientos de la cual se desprende esta primera asignatura, han sido considerados como esenciales e ineludibles para el desarrollo del cualquier plan de formación de este tipo, y que por ello, esta asignatura se encuentra relacionada de manera directa, con varias asignaturas de nivel intermedio y avanzado de estos planes de estudio.
   - Por otra parte, aun considerando que probablemente durante el desempeño laboral de estos profesionales, rara vez requieran programar, en el mundo académico se reconoce que una de las competencias importantes que deben poseer estos profesionales, es la capacidad para diseñar soluciones computacionales a problemas de procesamiento de datos.

2.- Adquirir y desarrollar las capacidades para identificar, seleccionar, combinar y organizar los diversos tipos de conocimientos que se implican en la resolución de un problema.
    - Conforme a lo que hemos planteado, esta es la habilidad más crítica y compleja de adquirir en un curso de este tipo. Esta es una habilidad esencial para asumir de manera efectiva la tarea de diseñar, implementar y probar soluciones a problemas de procesamiento de datos y para programar computadores. Estimamos que el desarrollo de estas capacidades, es altamente dependiente de la adquisición y desarrollo de estrategias cognitivas y de estrategias de regulación y de control del proceso de resolución, y de los producto que se general durante él.
    - El desarrollo de estas capacidades es fundamental en términos del aprendizaje que se requiere. Son las capacidades que entran en juego a la hora de llevar a cabo las tareas de combinar, los conocimientos declarativos y procedimentales implicados en un determinado problema, junto con los conocimientos declarativos y procedimentales vinculados con la metodología de resolución de estos problemas, y vinculados también a los conocimientos del ámbito del lenguaje de programación y del entorno de programación.
    - Este principio entonces, apunta al aprendizaje y el desarrollo de las capacidades de orden práctico que se requieren para utilizar el conjunto de conocimientos implicados en un determinado problema, y llevar a cabo las tareas de modelar la solución y para ejecutar la implementación del programa computacional de solución.

3.- Adquirir, desarrollar y perfeccionar, las estrategias cognitivas y de pensamiento que se requieren para el aprendizaje y el uso de los conocimientos teórico-formales y procedimentales, incluidas las estrategias cognitivas de regulación, y de control de la validez y efectividad de los aprendizajes.
    - Creemos que la adquisición y desarrollo de las estrategias cognitivas y de pensamiento, son esenciales para el aprendizaje y el desarrollo de las capacidades que hemos señalado, ya que ellas constituyen las herramientas para el aprendizaje y aplicación tanto de los contenidos teórico-formales como de los procedimentales concretos y verificables.
    - Partiendo de la premisa de que las estrategias cognitivas son para el aprendiz, las herramientas fundamentales para el aprendizaje, consideramos que este principio es de alta relevancia ya que dada la complejidad de los contenidos a aprender, se requiere no solo de un repertorio más amplio de estrategias, sino que también, se requiere de la capacidad para utilizarlas y aplicarlas de manera efectiva.
    - En este sentido por ejemplo, nos parecen de gran importancia el manejo de estrategias de pensamiento que permiten realizar de manera más

efectiva, las tareas de identificación, selección, combinación y organización de los distintos tipos de conocimientos que se encuentran implicados en la resolución de los problemas, ámbito en el cual las estrategias de pensamiento dirigidas a la construcción y el manejo de distintos modelos, tienen un carácter fundamental y esencial en las tareas de resolución de los problemas.

- Así también, juegan un papel crucial para el éxito de esta tarea, la disponibilidad y el manejo de estrategias de regulación y control, mediante las cuales el estudiante puede no solo evaluar la calidad y significancia de lo que aprende, sino que por sobre todo, evaluar y corregir los procesos personales de aprendizaje y de aplicación de los conocimientos que lleva a cabo, al momento de trabajar en la resolución de los problemas.

4.- Adquirir y desarrollar estrategias y habilidades para el trabajo en equipo.

- Para nosotros este principio tiene dos dimensiones. Una, fundamentada en las indicaciones contenidas en las recomendaciones curriculares (CC2001), en el sentido de que en el desarrollo de cualquier programa de formación profesional en computación, se deben hacer esfuerzos por dotar a los futuros profesionales, de habilidades interpersonales y de comunicación, que les habiliten para trabajar en equipos disciplinares e interdisciplinares de manera eficiente, esto en atención al hecho de que, durante su desempeño laborar, son profesionales que estarán enfrentados permanentemente, a la necesidad de interactuar y trabajar con profesionales de su misma área y con profesionales de otras áreas.
- En tanto la segunda dimensión de este principio, se relaciona con la forma de conducción del proceso de enseñanza, que nos interesa promover en el desarrollo de esta asignatura. Como se ha argumentado, uno de los pilares fundamentales en los que se sustenta el enfoque metodológico que hemos aplicado en el desarrollo de esta investigación, consiste en poner en práctica actividades de aprendizaje y trabajo cooperativo en la idea de favorecer el aprendizaje de los estudiantes.

Así entonces en este apartado hemos argumentado los cuatro principios que nos parecen esenciales en el aprendizaje de resolución de los problemas de procesamiento de datos: (1) Conocer y comprender de manera significativa los fundamentos teórico-formales de la disciplina, (2) Adquirir y desarrollar las capacidades para identificar, seleccionar, combinar, y organizar los diversos tipos de conocimientos que se implican en la resolución de los problemas, (3) Adquirir, desarrollar y perfeccionar las estrategias cognitivas y de pensamiento que se requieren para el aprendizaje y el uso de los conocimientos teórico-formales y procedimentales, incluidas las estrategias cognitivas de regulación, de control y de

evaluación de la validez de los aprendizajes, y (4) Adquirir y desarrollar estrategias y habilidades para el trabajo en equipo.

3.4. Estrategias metodológicas para la dinamización de los principios de aprendizaje

Actualmente, en el desarrollo de la asignatura, se utiliza una metodología tradicional, la que considera fundamentalmente, clases de tipo expositivas, en las que se presentan los fundamentos teórico-formales y los procedimientos para aplicarlos, y algunos ejemplos clásicos que son utilizados para explicar, desde la perspectiva del profesor, los detalles del desarrollo de los procedimientos que conllevan, la ejecución de las etapas del proceso completo de resolución de los problemas.

Durante el desarrollo de estas clases, el profesor va poniendo énfasis en aquellos aspectos que estima relevantes y/o que conforme a su experiencia docente, considera que pudieran representar mayores niveles de dificultad para los estudiantes.

Normalmente, durante estas sesiones los estudiantes actúan como meros receptores de la información que se les presenta, de tal forma que realizan pocas o ninguna intervención, ya sea para hacer preguntas aclaratorias respecto de los contenidos que se presentan, y/o respeto de los procedimientos que se aplican para desarrollar las distintas etapas de la metodología de resolución.

Los estudiantes en estas sesiones de clases expositivas, concentran sus esfuerzos en registrar la mayor cantidad de detalles, respecto de la ejecución de los procedimientos que implican el desarrollo de las etapas del proceso completo de resolución de los problemas.

Nuestra experiencia docente nos muestra que esta actuación de los estudiantes no favorece el aprendizaje, tanto de las bases teórico-formales y procedimentales de los fundamentos de programación, como tampoco, estimula y ayuda al desarrollo de la capacidad para transferir la ejecución de estos procedimientos, a situaciones de resolución de otros problemas de programación.

En algunos casos, estas sesiones son complementadas por prácticas en el laboratorio de computación, momento en el cual se trabaja en el uso del entorno de programación, y en los aspectos de implementación y prueba de los programas que han sido presentados en las sesiones de clases expositivas. Regularmente estas sesiones de laboratorio, son conducidas por los ayudantes de la asignatura.

Dado que estas sesiones de laboratorio se desarrollan en un contexto de práctica directa con el computador, en ellas los estudiantes se concentran esencialmente, en la manipulación del computador y del entorno de programación, abordando

principalmente las tareas de escribir los programas y utilizar el entorno de programación para compilar, ejecutar y depurar los programas.

En este contexto, cuando se realizan los procesos de prueba de los programas implementados y estos presentan errores, la generalidad de los estudiantes trata de corregirlos llevando a cabo un proceso no reflexivo de ensayo y error, en la intención de que en algún momento logren que el programa funcione correctamente.

Bajo estas condiciones, existen nulas oportunidades para que ellos puedan llevar a cabo un aprendizaje más reflexivo, y en el que participen de instancias intencionadas, ordenadas y sistemáticas, orientadas a profundizar en los aspectos de orden práctico, mediante los cuales se haga más evidente el uso y aplicación de las estrategias de pensamiento, que permiten hacer un uso adecuado de los conocimientos teórico-formales y los procedimentales al trabajar en el proceso de resolución de los problemas.

Dicho en otros términos, esta forma de conducir el proceso de enseñanza y aprendizaje, no favorece la implicación directa de los estudiantes en la práctica del propio proceso de resolución de los problemas que ellos deben aprender.

Por tanto consideramos que a la luz de la complejidad de los procesos de resolución de problemas que deben ser aprendidos, creemos que la implicación práctica y directa en estos procesos, es un aspecto esencial y fundamental, para mejorar el aprendizaje de nuestros estudiantes, tanto en lo referido al logro de los aprendizajes que demanda la asignatura misma, como para el aprendizaje de los conocimientos que les permitan construir la base disciplinar de la PC, que como hemos mencionado, es el andamiaje fundamental, para enfrentar otras asignaturas del plan de formación profesional de este tipo de carreras.

Conforme a esto, proponemos la puesta en práctica de un modelo de enseñanza y aprendizaje, que ponga el acento en la participación activa de los estudiantes, ello con el propósito de que adquieran, desarrollen, utilicen y apliquen estrategias cognitivas de orden superior, que son las que se requieren para la comprensión de los conocimientos de base, y para el desarrollo de la habilidad para usar los conocimientos en la ejecución de las tareas que implica el proceso de resolución de los problemas de procesamiento de datos.

Para llevar a cabo las tareas de resolución de estos problemas, además del aprendizaje de los conocimientos y su uso y aplicación en le ejecución de la tarea, se requiere que los estudiantes desarrollen estrategias de control y regulación, respecto de las formas de cómo, usan y aplican los conocimientos en ella, con el objeto de asegurar la efectividad del proceso de resolución que llevan a cabo.

Finalmente, se debe apuntar también al desarrollo de habilidades para el trabajo en equipo y de estrategias para el aprendizaje autónomo, como elementos propios del perfil profesional.

Hablamos de un modelo de enseñanza y aprendizaje, que se corresponde con el paradigma del enfoque estratégico de la instrucción, el cual se fundamenta en los principios de la psicología cognitiva de procesamiento de la información y en el constructivismo sociocultural.

Así, a la luz de la caracterización de los aprendizajes que demanda la asignatura, y los principios de aprendizaje que se han declarado, consideramos que la estrategia metodológica con que se aborde el desarrollo de la asignatura de FP, debe responder a dos dimensiones que nos parecen esenciales e ineludibles para un curso de este tipo. Una dice relación con el logro de las capacidades, habilidades y destrezas que la propia asignatura demanda, y otra relacionada con el desarrollo de determinadas estrategias cognitivas que los estudiantes necesitan para apropiarse de las capacidades, habilidades y destrezas que se pretenden.

La metodología debe favorecer los procesos de construcción del conocimiento, la compresión lógica y consciente de los contenidos, y la reflexión permanente respecto de cómo se aplican los conocimientos de la disciplina, y cómo se ejecuta el proceso mismo de resolución de problemas, todo ello a la luz de la aplicación rigurosa de los formalismos teóricos y procedimentales propios de la disciplina.

Así entonces, consideramos que el estilo metodológico, debe corresponder a uno que impulse a nuestros estudiantes, a poner en práctica un enfoque de aprendizaje de tipo profundo, lo que se consigue, creando un ambiente instruccional que ponga a los estudiantes en situaciones que les demanden un aprendizaje reflexivo, consciente y regulado.

Respecto a esto, existe claridad en reconocer que la utilización de un determinado enfoque de aprendizaje (en términos de la clasificación general de superficial o profundo), implica una interrelación entre las características personales y las reacciones inducidas por situaciones de aprendizaje en que el estudiante se ve involucrado.

Esto significa que aunque un estudiante pudiera tener una predisposición personal a utilizar un determinado enfoque de aprendizaje, esta condición puede ser modificada por la influencia de determinadas situaciones que estimulan, favorecen o inhiben la adopción de un determinado enfoque, con lo cual la elección de un enfoque es el resultado de una interacción rasgo-situación.

Así, un enfoque de aprendizaje designa tanto la forma en que un estudiante, de manera consistente, se enfrenta a la mayoría de las tareas de aprendizaje, como la forma en que se enfrenta a una tarea particular en un momento determinado (Biggs, 1991).

Desde la perspectiva de la apropiación del conocimiento, el enfoque de aprendizaje profundo se caracteriza por incorporar el análisis crítico de nuevas ideas, las cuales son integradas al conocimiento previo sobre el tema, favoreciendo con ello su comprensión y retención en el largo plazo, de tal modo que puedan ser utilizados de manera efectiva en tareas de solución de problemas (Biggs, 1991).

Para nosotros, en el contexto específico de los aprendizaje demandados por la asignatura, la puesta en práctica de este enfoque de aprendizaje, favorece la adquisición e integración de los conocimientos declarativos y procedimentales, con miras a la construcción del conocimiento estratégico, ya que este estilo de aprendizaje, promueve y desarrolla la utilización de altos niveles de habilidades cognitivas tales como “análisis” (comparar, contrastar) y “síntesis” (integrar el conocimiento en una nueva dimensión), además de caracterizarse por propender a desarrollar un aprendizaje de tipo autónomo, constructivo, cooperativo y diversificado.

Por tanto, creemos que la característica principal que debe poseer la metodología de enseñanza y aprendizaje de la asignatura, es que esta debe corresponder a una que promueva la adquisición, el desarrollo y el perfeccionamiento de estrategias y habilidades cognitivas, mediante la participación activa, reflexiva, regulada, ordenada, sistemática e intencionada de los estudiantes en el proceso de aprendizaje.

En este trabajo investigativo, apostamos por la idea de que en la medida que nuestros estudiantes tengan buenas oportunidades de, participar y apropiarse de la práctica directa en las tareas de resolución de los problemas, estarán teniendo mejores oportunidades, para apropiarse y desarrollar las capacidades, habilidades y destrezas de las que hablamos, por lo que la metodología debe concentrarse en establecer, una fuerte relación entre el individuo que aprende, los procesos cognitivos que este debe desarrollar, activar y perfeccionar, y el aprendizaje y desarrollo de la capacidad y habilidad para resolver los problemas de procesamiento de programación.

Consideramos que bajo las condiciones de una práctica directa, sistemática e intencionada, en las tareas de resolución de problemas, práctica en la cual los estudiantes son puestos intencionalmente en situaciones que les demanden la identificación, selección, combinación y organización de los conocimientos implicados en una determinada solución, y que junto con ello, los pongan en

situaciones que les exijan el control, y la regulación de los resultados que van alcanzando, al ejecutar las tareas de identificación, selección, combinación y organización de conocimientos, les estaremos creando mejores condiciones para el aprendizaje y el desarrollo de las capacidades que nos interesan.

Para nosotros, esta metodología debe considerar una mixtura adecuada y equilibrada entre, las sesiones de clases expositivas y las sesiones de práctica de resolución de problemas, las que deben darse en un ambiente de interacción y cooperación intencionada entre los estudiantes.

En la medida que los estudiantes actúan e interactúan de manera cooperativa en el aprendizaje, el desarrollo y el perfeccionamiento de las estrategias cognitivas y de la aplicación de las mismas, a las tareas de resolución de problemas, estarán favoreciendo el aprendizaje y el desarrollo personal y colectivo.

Sintetizando entonces, en este apartado hemos señalado que actualmente en el desarrollo de la asignatura, se utiliza una metodología de enseñanza y aprendizaje tradicional, fundamentalmente basada en clases expositivas, en las cuales se presentan a los estudiantes los elementos teórico-formales y los procedimientos para aplicarlos, y unos pocos ejemplos mediante los cuales se intenta explicitar, desde la perspectiva del profesor, los detalles del desarrollo de los procedimientos para realizar cada etapa del proceso de resolución de los problemas.

Durante la exposición de estos contenidos, los estudiantes son meros receptores de información realizando pocas o ninguna intervención.

Nuestra experiencia muestra que esta forma de desarrollar la asignatura no favorece el aprendizaje de los estudiantes, y especialmente, no crea las condiciones para el aprendizaje y el desarrollo de las capacidades de orden superior que requiere la tarea intelectual de resolución de los problemas.

Reiteramos entonces que para lograr los tipos de aprendizajes que hemos descrito, se requiere de la implicación práctica directa de los estudiantes, en las tareas de resolución de los problemas.

Proponemos la puesta en práctica de una metodología de enseñanza y aprendizaje, que ponga el acento en la participación activa de los estudiantes, con el objeto de que adquieran, desarrollen, utilicen y apliquen estrategias cognitivas de orden superior. La metodología debe también permitir la adquisición de estrategias de control, y regulación de lo que se aprende y de la forma de utilizar y aplicar lo que se aprende.

Esta metodología se corresponde con el paradigma del enfoque estratégico de la instrucción, el que se fundamenta en los principios de la psicología cognitiva de procesamiento de la información y el constructivismo sociocultural.

Con esto pretendemos favorecer los procesos de construcción del conocimiento, la comprensión lógica y consciente de los contenidos, y la reflexión permanente y sistemática, respecto de cómo se aplican los conocimientos de la disciplina y como se ejecuta el proceso de resolución de los problemas, todo ello a la luz de la aplicación rigurosa de los formalismos teóricos y procedimentales de la disciplina.

Para nosotros esta metodología debe considerar una mixtura adecuada y equilibrada, entre las sesiones de clases expositivas y las sesiones de práctica de resolución de problemas, las que deben darse en un ambiente de interacción y cooperación intencionada entre los estudiantes.

Mediante la actuación e interacción de los estudiantes de manera cooperativa, se crearan mejores condiciones para el aprendizaje, el desarrollo y el perfeccionamiento de las estrategias cognitivas y de pensamiento que se requieren.

SÍNTESIS GENERAL DEL CAPITULO 3

En este capítulo, nos hemos concentrado en analizar los principios que influyen el proceso de aprendizaje de la asignatura. Para estos efectos, hemos abordado los siguientes cuatro aspectos que nos parecen esenciales para este análisis:

1.- La asignatura de FP en el contexto del desarrollo del plan de formación profesional; ámbito en el cual hemos establecido que:
   - La asignatura es considerada por los expertos, como una asignatura troncal y nuclear para cualquier programa de formación profesional en CcC, cuestión que ha quedado establecida y justificada en varios informes de las comisiones técnicas que se preocupan de estos aspectos.
   - Aunque estas comisiones reconocen que en la actualidad pueden identificarse a lo menos 5 sub-áreas de formación profesional en CcC, indican que para cada una de estas áreas existen argumentos poderosos para considerar que los conocimientos de FP, son esenciales e ineludibles de incluir en cualquiera de estos planes de formación profesional.
   - Dentro de los argumentos para justificar estas afirmaciones, se señala que la formación de estos profesionales demanda que ellos deben alcanzar, una sólida base disciplinar respecto de los conceptos y los fundamentos teórico-formales de la programación de computadores, y que junto con ello, deben desarrollar la habilidad para aplicar estos conocimientos, al momento de diseñar e implementar las soluciones a los problemas, además de adquirir habilidades para el trabajo en equipos disciplinares e interdisciplinares.

- Para cubrir de manera adecuada el tratamiento de los contenidos y el desarrollo de las habilidades, normalmente se consideran a lo menos dos asignaturas de programación dentro de los planes de estudio, sin embargo mediante una análisis de las mallas curriculares es posible constatar además que, existen varias otras asignaturas que incluyen en mayor o menor grado, los temas de programación de computadores.
- Como se ha revisado, para el caso particular de la Universidad de Playa Ancha, la primera asignatura de programación corresponde a un curso de teórico-práctico, que pretende lograr que los estudiantes adquieran los conocimientos y las competencias fundamentales, que les permitan disponer de la capacidad para resolver problemas de procesamiento de datos y para programar computadores.
- En este contexto, el desarrollo de los contenidos se orienta a los propósitos de lograr que los estudiantes, adquieran la capacidad para, plantear soluciones a problemas susceptibles de ser resueltos mediante la programación de un computador, y que adquieran la capacidad para hacer uso de un lenguaje de programación, de sus recursos y de un entorno de programación, para implementar y probar soluciones a problemas de procesamiento de datos.

2.- Elementos a considerar para el aprendizaje de los conocimientos y las capacidades; en este ámbito hemos señalado que:

- Del análisis de la estructura de contenidos para el logro de los objetivos de esta primera asignatura, se identifican dos bloques temáticos, el primero referido a la metodología de resolución de los problemas de procesamiento de datos y el lenguaje algorítmico, y el segundo, trata con el concepto de computador como máquina que puede ser programada y con el lenguaje de programación.
- Con el tratamiento de estos bloques temáticos, se apunta al desarrollo de los conocimientos teórico-formales de la programación de computadores, pero también al desarrollo de las capacidades para utilizar y aplicar estos conocimientos a la resolución de los problemas de programación.
- Para nosotros el aprendizaje de estos conocimientos, así como también, para el desarrollo y perfeccionamiento de estas capacidades, los estudiantes necesitan disponer de un conjunto de estrategias de pensamiento, además de estrategias de regulación y control que les permitan evaluar y corregir el uso y aplicación, que están haciendo de los conocimientos, mientras realizan las tareas intelectuales de resolución de los problemas.
- El aprendizaje de estos conocimientos, así como el desarrollo de las capacidades que se han señalado, constituye la cuestión central y fundamental para el primer curso de programación de estos planes de

formación profesional, entendiendo que esto está por sobre la preocupación de sólo enseñar un lenguaje de programación.

- Consideramos que los elementos que se deben considerar para el aprendizaje de los conocimientos y las capacidades son: La forma particular cómo evoluciona el tratamiento de los contenidos que deben ser aprendidos; Las características de las capacidades que se deben aprender y desarrollar; Los recursos cognitivos que deben ponerse en juego; y Las características del propio proceso de enseñanza y aprendizaje.
- Un aspecto central del aprendizaje de estos conocimientos, así como también del aprendizaje y desarrollo de las capacidades para resolver problemas de procesamiento de datos, es el tipo de actividades cognitivas que los estudiantes deben desarrollar, las que incluyen actos cognitivos dirigidos a la retención y comprensión de los contenidos en ciertos momentos, y en otros, son actos cognitivos dirigidos a la puesta en práctica de formas de pensamiento reflexivo y analítico, que incluyen el pensamiento inductivo, deductivo, la creatividad y el pensamiento intuitivo.
- Estas son las estrategias cognitivas que se requieren para que al aborda un problema, el resolutor pueda hacer la transferencia de conocimientos, y para que pueda también, identificar, seleccionar, integrar, elaborar y organizar los conocimientos, en el contexto de la ejecución de las tareas de resolución de un problema particular.
- Sin embargo, las tareas de resolución de estos problemas requieren además del uso de estrategias cognitivas de control y regulación respecto de la propia actividad intelectual que se ejecuta.

3.- Principios básicos para el aprendizaje de las capacidades; ámbito en el cual hemos señalado que:

- El logro de los propósitos que se persiguen con el desarrollo de la asignatura, exige que el estudiante esté en disposición de llevar a cabo un aprendizaje activo, involucrándose de manera efectiva en su proceso de aprendizaje. Con esto se espera que los estudiantes, adopten una actitud reflexiva, analítica y crítica respecto de los conocimientos que aprenden y de la forma en que los utilizan y aplican al momento de trabajar en la resolución de los problemas.
- Desde otra perspectiva de la complejidad de estos aprendizajes, y conforme a lo que hemos descrito, al momento de resolver un cierto problema de procesamiento de datos, se deben ir desarrollando variadas tareas intelectuales, las cuales están dirigidas fundamentalmente a la elaboración y el uso de variados modelos. Para esto, junto con disponer de los conocimientos declarativos y procedimentales que son atingentes al problema, se ha de tener la capacidad para integrar estos conocimientos con el propósito de construir el conocimiento estratégico que permita

diseñar la solución al problema.

- Como se ha argumentado, para ejecutar de manera correcta el proceso constructivo de este tercer tipo de conocimiento, se requiere disponer de estrategias cognitivas de control y regulación, la cuales son necesarias para evaluar si el uso y aplicación que se está haciendo de los distintos tipos de conocimientos es correcto, y para evaluar el uso y aplicación del propio proceso de resolución que se está ejecutando, así como también para determinar y ejecutar las correcciones que son necesarias cuando las evaluaciones así lo indiquen.
- Conforme a nuestra experiencia, estos son los aspectos más críticos en el proceso de enseñanza y aprendizaje de los estudiantes en un primer curso de programación. Desde la perspectiva de la enseñanza, los profesores debemos ser capaces de crear condiciones y situaciones para que nuestros estudiantes aprendan los conocimientos, desarrollen la capacidad para usarlos y aplicarlos, pero que también, posibilite y favorezca en ellos el desarrollo y perfeccionamiento de las estrategias cognitivas de orden superior que hemos descrito.

4.- Estrategias metodológicas para la dinamización de los principios de aprendizaje; en este contexto hemos señalado que:

- Consideramos que el espacio de enseñanza y las condiciones para el aprendizaje, debe corresponderse con uno que promueva y favorezca la práctica intencionada y sistemática, que posibilite el aprendizaje y el desarrollo de las capacidades, es decir un espacio que demande la construcción, el uso y aplicación de los conocimientos implicados, así como también, que posibilite el desarrollo de las estrategias cognitivas que demanda la tarea de resolver los problemas.
- En esta perspectiva, para nosotros el proceso de enseñanza y aprendizaje debe ser un espacio de oportunidades, donde los estudiantes sean puestos en situaciones que les permitan aprender y desarrollar, las estrategias cognitivas que son de utilidad para estos propósitos, y que complementariamente, les demanden la utilización y puesta en práctica de las misma, creando también las condiciones para que regulen y evalúen el aprendizaje, el uso y los resultados que están consiguiendo con la aplicación de los conocimientos y las estrategias, procurando además, que el aprendizaje y la puesta en práctica de estos conocimientos, sea integral, en el sentido que los estudiantes deben entender que aunque el producto final del proceso de resolución de los problemas, es la propia solución expresada como un programa computacional, la ejecución de este proceso debe hacerse con rigurosidad disciplinar, esto es, haciendo uso correcto de las bases teórico-formales de la programación de computadores.
- Para ello, se propone un plan de clases multiestructurado en el que se

combinen de manera adecuada, tanto las clases expositivas, como las sesiones de práctica intencionada que hemos señalado. Para estas sesiones de práctica intencionada proponemos una organización basada en prácticas de aprendizaje y trabajo cooperativo, con el objeto de crear las condiciones que promuevan y favorezcan las situaciones en las cuales los estudiantes, mediante una participación directa en las tareas de resolución de los problemas, tengan también oportunidades para, practicar el uso y aplicación de los conocimientos, y también para desarrollar y perfeccionar las estrategias cognitivas que nos interesan.

- Sustentado en todo lo anterior, hemos establecido cuatro principios que nos parecen fundamentales de tener en cuenta para el diseño de un proceso de aprendizaje que apunte al desarrollo y perfeccionamiento de las capacidades que hemos señalado. Estos principios han de dirigirse a que los estudiantes:
  - A.- Conozcan y comprendan de manera significativa los fundamentos teórico-formales de la programación;
  - B.- Adquieran y desarrollen la capacidad para identificar, seleccionar, combinar y organizar los diversos tipos de conocimientos que se encuentran implicados en la solución de un problema;
  - C.- Adquirieran, desarrollen y perfeccionen las estrategias cognitivas requeridas; y

Adquirieran y desarrollen estrategias y habilidades para el trabajo en grupo.

CAPITULO 4
HACIA EL MODELO ALTERNATIVO DE ENSEÑANZA Y APRENDIZAJE

4. HACIA EL MODELO ALTERNATIVO DE ENSEÑANZA Y APRENDIZAJE

INTRODUCCIÓN

Tal y como hemos dejado constancia más arriba, aprender los FP de computadores en el contexto de las demandas de un primer curso de programación de una carrera de ciencias de la computación, implica por una parte, aprender y comprender las bases teórico-formales, los procedimientos y las técnicas en los cuales se sustenta la disciplina de la programación, pero también implica el aprendizaje, el desarrollo y el perfeccionamiento de las capacidades y habilidades para aplicar estos elementos en el diseño e implementación de soluciones a problemas de procesamiento de datos.

Conforme a lo que se ha argumentado y analizado, en el logro de estas capacidades y habilidades, se encuentran implicados el aprendizaje de tres tipos de conocimientos; los de tipos declarativos y conceptuales, los de tipo procedimentales, y aquel tipo especial de conocimiento, que se denomina conocimiento estratégico, que como se recordará, es el conocimiento que se construye como resultado de la propia práctica en la resolución de los problemas de procesamiento de datos.

En estos términos, el conocimiento estratégico, es un conocimiento que el resolutor va construyendo, como resultado de su propia práctica en la resolución exitosa de problemas, con lo cual, en una fuente de conocimiento adicional, de la cual podrá disponer para su uso y aplicación cuando trate con problemas similares, o cuando trate con problemas en los cuales una parte de la solución se corresponda con una solución ya construida y/o conocida. Por tanto, la construcción de este tipo de conocimiento tiene un carácter recursivo, en el sentido que se construye sobre sí

mismo, lo que lo enriquece, lo perfecciona, y lo hace ser una fuente de conocimientos que le permite al resolutor, tratar con problemas más complejos.

Son tres los aspectos fundamentales que deben aprender los estudiantes en un primer curso de programación de computadores:

1.- La base teórico-formal de la disciplina de la programación,
2.- Los procedimientos y las técnicas que usa y aplica la programación, y
3.- Las estrategias de resolución de los problemas de programación.

Como enfatizado, el último de estos aspectos, es el más complejo de aprender para la mayoría de los estudiantes, ya que implica el desarrollo y perfeccionamiento de las capacidades para usar y aplicar las dos primeras categorías de conocimientos.

Esta dificultad ha sido reconocida en los informes de resultados de varias investigaciones tales como las de: Raadt (2007); Lister, Adams, Fitzgerald, Fone, Hamer, Lindholm, et al. (2004); McCracken, Wilusz, Almstrum, Diaz, Guzdial, Hagan, et al. (2001); Porter, & Calder (2004); Sajaniemi & Kuittimen, (2005); Truong, Bancroft, & Roe, (2005); Robins, Rountree, & Rountree (2003); Sajaniemi, & Kuittinen, (2005); Whalley, Lister, Thompson, Clear, Robins, Kumar, et al. (2006)).

Como se ha planteado con anterioridad, el desarrollo de esta capacidad está asociado con la habilidad para poner en juego un conjunto de estrategias cognitivas y de pensamiento, al momento de desarrollar las tareas de búsqueda, diseño e implementación de soluciones a los problemas. Hablamos de estrategias dirigidas a identificar, seleccionar y organizar los conocimientos declarativos, procedimentales y los estratégicos cuando se trabaja en la resolución de un problema.

4.1. Las limitaciones del método tradicional de enseñanza

Conforme a la experiencia que hemos recogido en nuestros años de docencia en esta asignatura, así como también, sustentándonos en lo que plantean varios autores entre los cuales podemos citar a: Naidoo and Ranjeeth (2007); Ali (2005); Berglund, Daniels, and Pears, (2006); Burton and Bruhn (2003); Mitchell (2000); White and Sivitanides (2002); El-Sheikh (2006); Wolf (2003); Connolly (2007); Dann, Cooper & Pausch (2006); Gonzalez (2004); Lui, Kwan, Poon & Cheung (2004); McKenna, Nocedal, Freeman & Carr (2005); y Wellington, Briggs & Girard (2005), los métodos de enseñanza tradicionales que hacen uso exclusivo de las formas de instrucción directa y frontal para comunicar conocimientos, para lo cual se apoyan en las clases expositivas, sólo permite al docente mostrar ejemplos de cómo la teoría, los procedimientos y las técnicas han sido aplicadas para alcanzar el diseño y la implementación de la solución de un problema, pero no permiten evidenciar las estrategias cognitivas y los procesos mentales que entran en juego al momento de

diseñar e implementar las soluciones. Para nosotros, la principal razón de esto es que los procesos mentales son complejos y no estandarizados.

La no estandarización puede verse desde dos ámbitos. Por una parte, está el hecho de que la resolución de cada problema demanda una selección, integración y organización particular de conocimientos, y por otra, si se pide a dos o más resolutores que trabajen sobre la solución de un mismo problema, aunque ellos puedan llegar a una misma solución, con toda seguridad cada uno realizará una selección, integración y organización particular de conocimientos, esto debido fundamentalmente a que, como lo plantea la psicología cognitiva, los procesos mentales son propios de cada individuo.

Además, como sea señalado también, la ejecución del proceso de resolución de estos problemas, requiere la activación de procesos mentales dirigidos a la regulación y al control del propio proceso de resolución de los problemas, de su efectividad, es decir, requiere del uso de procesos cognitivos dirigidos a evaluar la validez del propio proceso de resolución que se sigue, y a evaluar la efectividad del uso y aplicación que se está haciendo de los distintos tipos de conocimientos, de las estrategias cognitivas y de las estrategias de pensamiento.

Desde nuestra perspectiva, estas acciones de regulación y control son fundamentales y esenciales para evaluar y corregir la propia tarea intelectual que lleva a cabo, ello con el objeto de asegurar que esta tarea está siendo correctamente encaminada al propósito último, que es el diseño y la implementación del programa de solución al problema.

Mediante las clases expositivas tradicionales, no es posible explicitar la identificación y el uso este tipo de estrategias y de los procesos de pensamiento, tanto en lo referido a la ejecución de las tareas de identificar, seleccionar y organizar los conocimientos, como en lo referido al uso de las estrategias de regulación y control que el propio proceso de resolución requiere, como tampoco, permiten explicitar los detalles de la actividad intelectual que implica la construcción y el trabajo con las distintas abstracciones y los modelos de representación con los que se trabaja durante el proceso de resolución. Explicitar estos elementos y procesos es difícil y hasta quizás imposible, debido a la propia complejidad de ellos.

Por lo demás, estos son procesos no estandarizados y de carácter dinámico. No están estandarizados porque cada resolutor puede realizarlos de manera diferente, y son dinámicos porque que la resolución de cada problema demandará variantes y acomodaciones del uso y aplicación de los conocimientos y las estrategias de pensamiento.

Estos aspectos no parecen muy cruciales y tratamos de transmitírselos a nuestros estudiantes, para lo cual enfatizamos e intentamos de que adquieran conciencia de que el aprendizaje de la programación de computadores, y el desarrollo de las capacidades y habilidades que ello conlleva, requiere necesariamente de la implicación directa de ellos, en lo personal, porque cada uno es responsable de su propio aprendizaje y consecuentemente de la aprobación de la asignatura, y en lo colectivo, ya que cada uno de ellos puede aprender de los otros.

Para esto recalcamos que el aprendizaje de la resolución de problemas de procesamiento de datos y la programación, se consigue trabajando directamente en las tareas de resolución, y que para garantizar el aprendizaje de esto, no basta con asistir a clases y durante el estudio personal, repetir los ejercicios presentados en las ellas, así como tampoco es de utilidad para el aprendizaje, sólo observar y recoger anotaciones cuando otros trabajan en la resolución de los problemas.

Por tanto consideramos que la enseñanza y el aprendizaje de los FP de computadores, requieren de un enfoque metodológico que más que orientarse a la transmisión de conocimientos y a mostrar ejercicios en los cuales se les usa y aplica, se oriente hacia formas que promuevan y favorezca la construcción del conocimiento por parte de los estudiantes, por la vía de crear condiciones y situaciones para la práctica directa e intencionada dirigida al uso y aplicación de los conocimientos, las estrategias cognitivas, de pensamiento, de control y regulación que como hemos señalado, se requieren en este tipo de tareas intelectuales.

En síntesis, lo que estamos planteando es que conforme a lo que señalan y afirman varios investigadores y confirmado por nuestra experiencia docente en esta asignatura, es que los métodos de enseñanza tradicional que usan fundamentalmente clases expositivas, sólo permiten al docente presentar los contenidos teóricos y procedimentales y mostrar algunos ejemplos de cómo la teoría, los procedimientos y las técnicas han sido aplicados al diseño e implementación de la solución de un problema particular, pero no permiten evidenciar las estrategias cognitivas y los procesos mentales que deben actuar durante el trabajo en la solución, esto debido a la complejidad de las estrategias y los procesos mentales mismos. Cada problema, demanda una selección, integración y organización particular de conocimientos, y además, como lo plantea la psicología cognitiva, frente a una tarea intelectual, los procesos mentales de dos o más individuos serán distintos, por lo que no es posible enseñarlos mediante una estandarización de los mismos.

Otro factor condicionante en el aprendizaje de la resolución de estos problemas, es el hecho de que al aplicar los distintos tipos de conocimientos, se requiere de acciones de regulación y control dirigidas a verificar si la propia aplicación de los conocimiento que se está llevando a cabo, es correcta, es decir, se requiere de

acciones de evaluación y corrección (si es necesario) de la propia tarea intelectual que se está realizando, acciones que claramente no son posibles de enseñar mediante las clases expositivas.

4.2. Consideraciones para un modelo de enseñanza y aprendizaje

Como hemos señalado, la asignatura de FP es una asignatura de tipo teórico-práctica que pretende que los estudiantes aprendan, comprendan y apliquen los fundamentos teórico-formales de la programación, como base disciplinar, para aplicar los procedimientos y las técnicas de diseño e implementación de las soluciones a los problemas que se abordan en el primer curso de programación.

Sin embargo, la experiencia muestra que aunque un estudiante conozca y comprenda los fundamentos teórico-formales, los procedimientos y las técnicas de diseño e implementación, esto no es suficiente para lograr el desarrollo de las capacidades y habilidades para aplicarlos al diseño e implementación de soluciones.

La integración entre los elementos teórico-formales, los procedimentales y las técnicas, ha sido reconocida por varios autores como una capacidad fundamental que deben desarrollar los estudiantes, en todas las asignaturas de la línea de programación, y es un aspecto que ha sido destacado de manera especial en las recomendaciones contenidas en el documento CC2001.

Estas dificultades se evidencian por la constatación experiencial, que da cuenta que la mayoría de los estudiantes son conocen y manejan con bastante precisión:

- El concepto de algoritmo y su utilidad,
- El concepto de computador y su organización,
- El concepto de lenguaje de programación y las distintas estructuras de control que posee un lenguaje,
- Las etapas de la metodología de resolución y los propósitos que se persiguen con el desarrollo de cada una de ellas,
- Las características de los paradigmas de programación y cómo un determinado lenguaje de programación se relaciona con uno o más de los paradigmas, y
- Los procedimientos de cálculo matemáticos implicados en la generalidad de los problemas con que se tratan en un primer curso de programación,

pero, sin embargo, tienen dificultades a la hora de utilizar y aplicar estos conocimientos, para diseñar e implementar soluciones para problemas de procesamiento de datos.

En términos prácticos, tienen dificultades para llevar a cabo el trabajo intelectual que se requiere para seleccionar, integrar y organizar estos conocimientos, de manera tal

que les permitan construir el conocimiento estratégico que es pertinente, para diseñar e implementar una determinada solución.

Si bien para resolver cada problema con los que se trata en un curso regular de FP, se utiliza la misma base teórico-formal, los mismos procedimientos y técnicas, ocurre que la selección, integración y organización de estos elementos es de carácter particular para cada problema que se aborda, ya que éstos tienen la categoría de problemas y no de ejercicios en los que se demandan la repetición de procedimientos.

Si bien la psicología cognitiva ha estudiado los aspectos procesuales referidos a la ejecución de la tarea intelectual de resolver problemas de carácter general, aún no es posible establecer con claridad los detalles de cómo operan los procesos en el intelecto del resolutor, la ejecución de dicha tarea y los pasos que la ejecución de esta tarea conlleva.

Lo que sí está claro para los psicólogos cognitivistas, es que el acto de resolver un problema demanda del resolutor una construcción de conocimientos, construcción que además, tiene un carácter particular para cada problema que se resuelve y que como proceso intelectual, éste también puede ser ejecutado de manera diferente por cada resolutor.

La psicología cognitiva plantea que la solución satisfactoria y eficaz de un problema, demanda una interacción entre las bases de conocimientos, la organización de la información de entrada, el uso de estrategias de procesamiento y la realización de actividades orientadas al logro de una meta.

Llevado esto a la resolución de problemas de procesamiento de datos, implicaría la interacción entre los conocimientos teórico-formales, los procedimentales y las técnicas, lo que debe dar origen a la identificación, selección, integración y organización de los distintos conocimientos, lo que resulta en la construcción del conocimiento estratégico. Sin embargo, esta construcción demanda también el uso de estrategias de regulación y control que permitan orientar la tarea intelectual hacia el logro de la meta.

En esta perspectiva, las teorías más modernas señalan que resolver problemas, es un proceso cognitivo, dirigido a la utilización de conocimientos y estrategias, para construir representaciones o modelos, que sean de utilidad y ayuda para transformar el enunciado de un problema y la información contenida en ella, en la solución al mismo, bajo el supuesto de que no existe un método obvio de solución, o que si existe tal método, este es desconocido por quién elabora la solución.

Se reconoce entonces, que resolver problemas demanda la ejecución de una actividad cognitiva que ocurre dentro de la mente o en el sistema cognitivo del resolutor. Además, resolver un problema conlleva la ejecución de un proceso dirigido a la manipulación de diversos conocimientos que se encuentran almacenados en el sistema de memoria del resolutor, lo que implica la realización de operaciones cognitivas a partir de representaciones que este maneja, además de otras representaciones o modelos que deberá ir elaborando como producto de la propia tarea intelectual.

También conlleva dirigir acciones con intencionalidad para obtener la meta deseada, acciones que además deben ser evaluadas, corregidas y reguladas a objeto de asegurar que ellas están siendo correctamente encaminadas al logro de la meta.

Por último, se considera que la tarea de resolver problemas es personal, los problemas son diferentes para cada resolutor, dependiendo de los conocimientos previos, de sus experiencias, y de las habilidades cognitivas con que cuente, lo que concuerda con la afirmación que señalamos anteriormente, en el sentido de que para aprender a resolver problemas de procesamiento de datos, cada resolutor debe trabajar en las tareas intelectuales que son propias de este proceso.

Por tanto, consideramos que los procesos de resolución de problemas, son procesos que más que poder ser enseñados, deben ser aprendidos, practicados y perfeccionados por cada resolutor. Bajo esta perspectiva, se requiere de la puesta en práctica de situaciones de aprendizaje diseñadas intencionalmente, para promover y favorecer el involucramiento directo de los estudiantes, en las tareas de resolución de los problemas.

Para el logro de estos propósitos, consideramos que los estudiantes deben ser puestos en situaciones intencionadas que, favorezcan y promuevan en ellos un aprendizaje de tipo activo, con lo cual esperamos que ellos se involucren de manera directa y efectiva, en la propia tarea intelectual de resolución de los problemas, tomando también parte activa y directa en sus procesos personales dirigidos al uso, aplicación y construcción de los conocimientos implicados.

La experiencia que hemos recogido durante estos años de docencia en esta asignatura, nos ha mostrado que estas son las condiciones ideales para el aprendizaje de las capacidades y habilidades para resolver los problemas de procesamiento de datos, en el sentido de que cuando conseguimos que nuestros estudiantes se involucren de manera activa y directa en las tareas de resolución de estos problemas, ellos van paulatinamente, adquiriendo y perfeccionando las capacidades y habilidades para ser resolutores de problemas más efectivos.

Hemos observado que en la medida en que ellos construyen, comunican, expresan, defienden y discuten sus construcciones personales de conocimientos y sus estrategias de resolución de los problemas, van perfeccionando sus propias capacidades y habilidades para diseñar e implementar las soluciones.

Creemos que la razón de esto es que bajo estas condiciones, ellos actuaran como constructores de su propio conocimiento, haciendo uso y mejorando sus propios procesos cognitivos y de pensamiento, y llevando a cabo además acciones de regulación y control sobre sus actos intelectuales.

Así entonces, la enseñanza y el aprendizaje de la resolución de problemas de procesamiento de datos demanda, el diseño y puesta en práctica de situaciones que de manera intencionada y organizada, permitan que los estudiantes se involucren y trabajen directamente en las tareas de resolución de problemas.

En la medida en que ellos sean puestos en situaciones concretas de resolución de problemas debidamente contextualizados (es decir, que trabajen en problemas lo más reales posibles, aun cuando sean versiones gradualmente complejizadas de un mismo problema), los estudiantes tendrán mejores condiciones para tratar con la construcción de los conocimientos, al tiempo que tendrán buenas oportunidades para aprender, desarrollar y perfeccionar las estrategias cognitivas, de pensamiento y de regulación y control, que se requieren para alcanzar las capacidades y habilidades para resolver los problemas de programación.

Por tanto, las consideraciones tendremos en cuenta para proponer un modelo de enseñanza y aprendizaje, que sea más efectivo para el aprendizaje de la programación de computadores en el contexto de una primera asignatura las podemos sintetizar en que:

- Resolver problemas de programación, demanda la ejecución de una actividad intelectual dirigida a la utilización de conocimientos y estrategias, para construir representaciones o modelos, con los cuales trabajar a partir del enunciado para obtener la solución del problema.
- Un aspecto crucial y crítico para el éxito de esta actividad intelectual, es la identificación, selección, integración y organización de los distintos conocimientos que podrían verse involucrados en la solución de un cierto problema.
- Para aseguran que lo anterior está conduciendo a la solución del problema, se deben poner en juego estrategias de regulación y control sobre el proceso que se ejecuta y de los conocimientos que se usan.
- La ejecución de esta actividad intelectual es de carácter personal, ya que cada resolutor utilizará sus propios conocimientos y estrategias para la construcción de las representaciones o modelos, sin embargo la experiencia

> muestra que el aprendizaje y el perfeccionamiento en la ejecución de esta actividad intelectual, puede verse enriquecido con la interacción social entre pares que están abocados a la misma tarea, ya que en ella, potencialmente se dan condiciones para la discusión, la reflexión, la confrontación, argumentación y defensa de ideas y estrategias, condiciones bajo las cuales, unos aprendices pueden beneficiarse de otros, y pueden también enriquecer sus propios conocimientos y estrategias.

Por lo tanto en el desarrollo de la asignatura de FP, deben considerarse instancias para el tratamiento de los conocimientos teórico-formales, de los fundamentos, de los procedimientos y las técnicas, pero también han de considerarse instancias que favorezcan la adquisición, el desarrollo y el perfeccionamiento de las capacidades y estrategias que se han señalado.

En esta perspectiva se propone que el plan de clases de la asignatura considere el desarrollo de clases expositivas para el tratamiento de los conocimientos, y junto con ellas, el desarrollo de actividades basadas en el aprendizaje cooperativo como instancias para favorecer el aprendizaje, el desarrollo y el perfeccionamiento de las capacidades y estrategias, para ejecutar las tareas intelectuales que son propias del uso y aplicación de los conocimientos.

Mediante la puesta en práctica de este tipo de actividades de aprendizaje, se pretende también que nuestros estudiantes desarrollen y perfeccionen el uso y aplicación de las estrategias de regulación y control que se requieren, para el correcto desarrollo de la tarea intelectual de resolución de los problemas de procesamiento de datos.

Confiamos en que la combinación de clases expositivas y las actividades de tipo cooperativas que se proponen, nos permitiría tanto transmitir los conocimientos teórico-formales y los referidos a los procedimientos y técnicas en que se sustenta la disciplina de la programación, y además promover y favorecer el aprendizaje activo de los estudiantes.

Con las actividades de tipo cooperativo se trabajan los aspectos de uso y aplicación de los conocimientos para la resolución de los problemas, pero por sobre todo, se trabajan los aspectos de construcción del conocimiento para la resolución de los problemas y también el desarrollo y el perfeccionamiento de las capacidades y estrategias para estos propósitos.

Mediante el diseño, el uso y la aplicación de actividades de trabajo cooperativo se apunta a favorecer condiciones para la construcción del conocimiento, y para el desarrollo y perfeccionamiento de las capacidades y estrategias que se han señalado.

Confiamos que mediante la confrontación de ideas entre los estudiantes, se generaran oportunidades para la construcción, organización y expresión de argumentos, mientras otros estudiantes observan, registran y buscan información, y piensan que hacer y cómo usar la información que están recibiendo de sus pares, y tendrán mejores oportunidades de alcanzar los aprendizajes que se requieren.

En síntesis, se ha planteado que la asignatura de FP es una asignatura teórico-práctica, con la que se persigue el propósito de que los estudiantes aprendan, comprendan y apliquen los fundamentos teórico-formales de la programación. Además de esto, se espera que los estudiantes aprendan, comprendan y apliquen los procedimientos y las técnicas para diseñar e implementar soluciones a problemas de procesamiento de datos.

Sin embargo, como se ha reiterado en varios apartados, un aspecto relevante que se debe lograr con esta asignatura, es el aprendizaje, desarrollo y perfeccionamiento del conocimiento estratégico, el cual, es absolutamente necesario para que los estudiantes dispongan de la capacidad y las estrategias para seleccionar, integrar y organizar los distintos tipos de conocimientos que se encuentran implicados en este tipo de soluciones.

Para esto, se considera que los estudiantes deben ser puestos en situaciones intencionadas de resolución de problemas, situaciones que promuevan y favorezcan en ellos un aprendizaje activo y reflexivo, donde presten conscientemente atención, a los procesos de pensamiento implicados en la tarea de resolución de los problemas.

Hablamos de un ambiente de aprendizaje en el que los estudiantes tanto individual como colectivamente, construyan, comuniquen, expresen, defiendan y discutan, respecto de la aplicación de los conocimientos y las estrategias de pensamiento durante la resolución de los problemas.

Nuestra experiencia nos ha mostrado que estas condiciones favorecen el logro de estos aprendizajes y el desarrollo de las capacidades. Por lo tanto el desarrollo de la asignatura no sólo debe pretender el tratamiento de los contenidos teórico-formales, los procedimientos y las técnicas, sino que también el desarrollo y el perfeccionamiento de las capacidades para la resolución de los problemas, para lo cual se propone que en el plan de clases se consideren las clases expositivas, y un conjunto importante de actividades de trabajo cooperativo, las que han de ser diseñadas fundamentalmente para, trabajar los aspectos de aplicación de los conocimientos y para desarrollar y perfeccionar las capacidades que hemos descrito.

4.3. Método de Aprendizaje Cooperativo (AC) a utilizar como alternativa.

El AC es una denominación genérica utilizada para referirse a una amplia gama de enfoques (Coll y Colomina, (1990)), y cuyo denominador común es la división del grupo de estudiantes que conforman un curso, en subgrupos o equipos, a los cuales se les pide que desarrollen una actividad o una tarea previamente programada y estructurada.

El AC se caracteriza por generar condiciones para que en los grupos exista un elevado grado de igualdad en las relaciones entre sus integrantes, lo cual nos motiva a su uso ya que nuestra intención es generar condiciones para que, todos los estudiantes que conforman un grupo, participen de manera activa en las reflexiones y discusiones en torno al diseño y la implementación de las soluciones a los problemas de programación, al tiempo que favorezcan sus construcciones personales de conocimientos y estrategias, al socializar las formas particulares en que usan y aplican los conocimientos y las estrategias de resolución, por la vía de comunicarlas, argumentarlas y defenderlas.

Para Echeita (1995), Johnson y Johnson (1989) y Slavin (1990), existen tres requisitos básicos para que se pueda hablar de AC. El primero hace referencia a la existencia de una tarea grupal, es decir, debe existir un objetivo que los distintos integrantes del grupo que trabajan conjuntamente deben alcanzar de manera conjunta. Por tanto, la situación debe implicar no sólo hacer una determinada actividad juntos, sino afrontar y resolver una tarea o problema común y como consecuencia de ello, aprender juntos. En este sentido, una de las intenciones es que los integrantes de cada grupo trabajen de manera comprometida en las tareas de resolución de los problemas.

El segundo requisito es que, para la resolución de la tarea o problema se demanda la contribución de todos y cada uno de los integrantes del grupo. En nuestro caso para conseguir esto, deberemos motivar permanentemente a los integrantes de cada grupo, para que contribuyan de manera activa en las tareas de resolución de los problemas, resaltando que el involucramiento efectivo en estas tareas, contribuirá a sus aprendizajes individuales.

Y el tercer requisito se refiere a los recursos del grupo, los que deben ser suficientes para mantener y hacer progresar la actividad colectiva e individual, tanto desde el punto de vista de la regulación de las relaciones interpersonales, como en lo relativo al desarrollo y ejecución de la tarea. Este es un aspecto que debemos cuidar y monitorear, para dar garantías de alcanzar el mejor aprovechamiento posible de las actividades cooperativas, ya que tratamos con estudiantes que no están acostumbrados a trabajar cooperativamente, por lo que se corre el riesgo que aunque tenga una organización administrativa como grupo, no trabajen cooperativamente,

con lo que sin duda, estará poniendo en riesgo nuestras intenciones de que unos aprendan de los otros.

La intención es sacar el máximo provecho posible a lo que afirman Karau y Williams (1993) en el sentido de que uno de los principales elementos que da impulso para que los estudiantes se motiven por participar en las actividades de AC se basa en la teoría del esfuerzo colectivo según la cual, los sujetos sólo se esfuerzan en una tarea grupal si creen que esforzándose conseguirán un mejor rendimiento y si creen que el rendimiento será recompensado de alguna forma.

Además en una perspectiva más práctica, confiamos en que como nuestros estudiantes de la asignatura de FP, tienen antecedentes y/o saben de las dificultades inherentes que tiene el aprendizaje de la programación, y los altos índices de fracaso que se dan en la asignatura, ellos estarán más dispuestos a participar en actividades que favorezcan sus aprendizajes.

El AC se sustenta en una visión constructivista del aprendizaje, enfoque que conforme a los argumentos que se han dado en apartados anteriores, es propicio para los tipos de conocimientos, capacidades y estrategias que deben aprender y perfeccionar los estudiantes en esta asignatura. Para nosotros, la visión constructivista del aprendizaje de los FP se hace manifiesta en términos de que se orienta a:

- El procesamiento de la información y del conocimiento, y sus significados, lo que permite que estos sean usados y aplicados de manera correcta,
- Al trabajo en la resolución de problemas y no de ejercicios repetitivos; la identificación, selección, integración y organización de conocimientos que requiere la resolución de cada problema,
- Al trabajo con distintas abstracciones y modelos, como una cuestión esencial para transformar el enunciando del problema y la solución del mismo, y
- Al aprendizaje y el perfeccionamiento del conocimiento estratégico, que está determinado por la capacidad para usar y aplicar los conocimientos a la resolución de los problemas.
- Al aprendizaje y el perfeccionamiento de las estrategias de resolución de los problemas, y las estrategias de control y regulación sobre el propio procesos de resolución y del uso y aplicación de los conocimientos, estrategias que como se ha argumentado, son fundamentales para abordar con éxito las tareas de resolución de esta clase de problemas.

Para el constructivismo el estudiante construye el conocimiento, aprendiendo de forma significativa y activa. El constructivismo representado por Piaget, Vygostky, Novak y Ausubel, confirma que el progreso en el desarrollo del aprendizaje, esta mediado por la interacción social.

Por tanto, nuestra intención es usar AC como alternativa pedagógica que, sustentada en el constructivismo, nos permita mejorar el aprendizaje de la resolución de problemas de programación.

Con la visión constructivista de los procesos de enseñanza y aprendizaje en el contexto específico de las actividades de la asignatura de FP, intentamos que los estudiantes construyan su propio conocimiento y desarrollen y perfeccionen las capacidades y estrategias para resolver problemas, y donde el docente actúe en todo momento como mediador, guía, apoyo y monitor del aprendizaje.

Estos son los elementos que nos han motivado a usar el AC, como enfoque metodológico que nos permita que los estudiantes puedan asociar lo conocido con lo nuevo y, modificar y reestructurar sus concepciones previas, y de manera especial, usar formas que favorezcan la práctica en el desarrollo de los procesos cognitivos, de pensamiento, y de desarrollo y perfeccionamientos de las estrategias que se encuentran implícitas en las tareas de resolución de los problemas de procesamiento de datos.

En el meta-análisis efectuado por Johnson, Maruyama, Johnson, Nelson y Skon (1981), se verificó que la cooperación es superior a la competición y a la individualidad, en cuanto al rendimiento y la productividad de todos los participantes. Estos hallazgos son independientemente de la naturaleza del contenido (lenguaje, lectura, matemáticas, ciencias naturales, estudios sociales, psicología, actividades artísticas, educación física...), de las edades de los integrantes de grupos y de los niveles, incluido el universitario.

Según estos mismos autores, esto se hace más evidente en tareas que implican adquisición de conceptos, solución de problemas complejos, retención y memoria, ejecución motora y tareas de suposición, juicios y predicciones, ámbitos en los cuales las técnicas de AC mejoran la calidad de las estrategias de aprendizaje, desarrollan estrategias de procesamiento de información, favorecen el pensamiento crítico y constructivo, a la vez que favorecen la capacidad de comunicación y expresión.

En el aprendizaje de la resolución de problemas de programación de computadores, se requiere de gran parte de estos elementos ya que se necesita:

- Adquisición de conceptos, la mayoría de ellos muy abstractos, como el concepto de algoritmo, de computador, de lenguajes de programación, de programa computacional en ejecución, etc.,
- La adquisición de la capacidad para usar y aplicar diversos tipos de conocimientos en tareas de resolución de los problemas que son complejos porque no tienen solución conocida para los estudiantes,
- La realización de juicios y predicciones respecto, por ejemplo, de cómo se comportará computacionalmente un algoritmo o un programa en ejecución,

- Procesamiento de la información tanto en el ámbito de la interpretación del enunciado de un problema, como en el ámbito de la determinación de cuáles conocimientos son aplicables para diseñar la solución al mismo,
- La adquisición y perfeccionamiento de estrategias de pensamiento, y de regulación y control, para evaluar la efectividad del propio pensamiento y el uso que se hace de los conocimientos.

En opinión de estos autores, además de las consecuencias positivas a nivel cognitivo, el AC produce resultados de gran interés pedagógico tales como, motivación intrínseca, actitudes positivas hacia los contenidos, mejora en la autoestima y en el apoyo social, favoreciendo la cohesión grupal y la participación activa de los integrantes del grupo. Estos son aspectos sobre los cuales también nos interesa sacar el máximo de provecho.

Dada la complejidad intrínseca de esta primera asignatura, consideramos que es muy importante que los estudiantes desarrollen una fuerte motivación por aprender y que desarrollen actitudes positivas hacia el aprendizaje de los contenidos que en ella se tratan, ya que como sea señalado y argumentado en apartados anteriores, los conocimientos respecto de programación y más específicamente, respecto del diseño de soluciones a problemas de procesamiento de datos, constituye un núcleo de conocimientos que es fundamental en los planes de formación en Ingeniería en informática, y por tanto los conocimientos iniciales que en esta asignatura se deben aprender, son fundamentales para tratar con varias otras asignaturas de estos planes de formación profesional.

Por otra parte, es ampliamente sabido que un adecuado nivel de autoestima en un aprendiz, contribuye de manera favorable a sus actitudes como estudiante.

Para Brown y Atkins (1998), los objetivos de las técnicas de AC en los niveles de educación superior, son principalmente tres:

1.- El desarrollo de estrategias de comunicación: lo que comprende estrategias de comprensión, explicación, pregunta y respuestas. La discusión y el debate, sirven no sólo para comunicarse con los demás, sino también para perfeccionar la utilización del lenguaje que es propio de los contenidos que se tratan en una situación particular de aprendizaje.

   Nos parece que el desarrollo de este tipo de estrategias favorecerá tanto la interacción de los estudiantes durante las actividades de AC, como el desarrollo de una de las habilidades importantes que se señala en los perfiles profesionales de estos planes de formación, ya que se considera que un Ingeniero informático para su desempeño laboral, requerirá de buenas capacidades comunicativas, para interactuar en y con equipos multidisciplinares.

2.- Desarrollo de competencias intelectuales y profesionales: lo que comprende estrategias que permiten al estudiante analizar, razonar lógicamente, valorar y juzgar, pensar críticamente, sintetizar, diseñar, resolver problemas. Las situaciones de AC fomentan este tipo de pensamiento superior, al favorecer todo tipo de interacciones entre los estudiantes.

   Este objetivo está en directa relación con el propósito central de la primera asignatura programación, en el sentido de que con esta asignatura, se espera que los estudiantes adquieran los conocimientos fundamentales de la disciplina, pero que también sean capaces de usarlos y aplicarlos para resolver problemas de programación, lo que lleva implícito, el aprendizaje y el perfeccionamiento de las estrategias de pensamiento, de regulación y control, que son necesarias para llevar a cabo las tareas de resolución de estos problemas.

3.- Crecimiento personal: lo que incluye el desarrollo de la autoestima, procesos metacognitivos, conocerse a sí mismo y a los demás. La experiencia en grupo proporciona conocimientos internos personales, derivados de los procesos de interacción e investigación dentro del grupo que se traducen en una mayor maduración personal, una mayor autoconciencia y compromiso. Dentro del grupo también se satisfacen las necesidades personales, las dudas y las ansiedades lo que favorece ambientes más gratos para el aprendizaje.

   Para el interés en este trabajo, el logro de este objetivo es significativo, ya que esperamos principalmente que los integrantes de los grupos cooperativos, adquieran y fortalezcan su compromiso, tanto por el propio aprendizaje, como por el aprendizaje colectivo, disminuyendo de paso las ansiedades, para lograr que las actividades de trabajo cooperativo represente para ellos un ambiente de aprendizaje grato y provechoso.

La técnica de AC que proponemos utilizar corresponde a una adaptación de la técnica conocida como Learning together (aprendizaje juntos) (Johnson y Johnson, (1987)), la cual se fundamenta en el aprendizaje y la interacción de estudiantes en pequeños grupos heterogéneos. Esta técnica ha demostrado su utilidad en el aprendizaje sobre resolución de problemas, aprendizaje de conceptos y en el desarrollo de la creatividad.

La técnica se apoya en cinco elementos: interdependencia positiva, interacción cara a cara, rendimiento de cuentas individual, destrezas sociales y procesamiento grupal (Johnson y Johnson, (1987)).

Para sus autores, la interdependencia positiva entre los miembros del grupo se consigue mediante la forma de estructurar el material de trabajo que se entregará para la realización de la actividad cooperativa. Los autores comparten la idea de

Bossert (1989) de que se puede conseguir interdependencia positiva estructurando la tarea y no la recompensa.

Para Johnson y Johnson (1987), la interdependencia positiva se puede conseguir dándole al grupo una sola copia de los materiales de trabajo, de tal manera que para llevar a cabo la tarea encomendada necesitan trabajar juntos tanto en el estudio, como en el análisis y la reflexión del contenido del material y de las tareas de aprendizaje demandadas por la actividad cooperativa.

En nuestro caso, proponemos que al inicio de la actividad se entregue a cada grupo cooperativo una sola copia de los enunciados de los problemas para los cuales deben diseñar e implementar la solución, sin embargo, una vez cumplido el tiempo de entrega de las soluciones, se distribuirán copias todos los integrantes del grupo, con el objeto de que cada estudiante disponga del material necesario para su estudio personal.

Así también, consideramos que la interdependencia positiva se puede potenciar proponiendo a los integrantes de los grupos, que se asignen responsabilidades individuales, las que al ser cumplidas cabalmente, favorecerán el desempeño y el éxito del trabajo grupal. Estas responsabilidades están referidas por ejemplo, a la provisión de los materiales complementarios necesarios para el trabajo del grupo, y a tomar notas y apuntes de manera clara y ordenada, en la intención de que estos sean compartidos y complementados por el resto de los integrantes del grupo cooperativo.

Para los creadores de la técnica, la interacción cara a cara se consigue creando ciertas condiciones de distribución de los integrantes durante el desarrollo de la actividad cooperativa, como por ejemplo, distribuyendo el mobiliario de la sala de clases o del espacio de trabajo, de manera tal que se facilite la interacción entre los integrantes del grupo.

Ellos proponen que los miembros del grupo debieran sentarse en círculo y estar lo suficientemente cerca unos de otros para comunicarse eficazmente, y conforme a una organización del espacio que consiga que los otros grupos no se constituyan en una dificultad para la interacción entre los integrantes de un grupo en particular.

En nuestro caso, todas las actividades de AC se realizaran en el laboratorio de computadores, el cual cuenta con una organización del espacio y el mobiliario que facilita la interacción cara a cara. Las mesas y en ellas se ubican a lo más dos computadores, de los que se dispone de la cantidad suficiente, para asignar uno a cada estudiante.

Sin embargo hemos analizado que esta última característica, podría constituirse en un elemento que ponga en riesgo la interacción cara a cara, ya que si cada estudiante dispone de un computador durante el desarrollo de las actividades, esto creará condiciones para que caigan en la tentación de separarse del grupo para desarrollar el trabajo de manera individual, con lo cual se pierde por completo el propósito de la interacción cara a cara como parte importante de la actividad cooperativa.

Con el objeto de prevenir y evitar esta situación, desarrollaremos una supervisión permanente, instando a los estudiantes a trabajar en la actividad de manera grupal, y tratando de convencerles de al interior del grupo, solo uno de los integrantes asuma la responsabilidad del manejo del computador, sugiriendo que quien asuma ésta responsabilidad, sea el estudiante que tenga un mayor nivel de dominio del manejo operativo del mismo. Lamentablemente no podremos realizar acciones de este tipo, en las actividades cooperativas que ellos realizarán en horario extra clases.

En relación con el principio de responsabilidad y rendimiento basado en la rendición de cuentas individual, Johnson y Johnson, (1987), enfatizan que son necesarios los mejores esfuerzos de aprendizaje de cada miembro para que el grupo tenga éxito, para lo cual, el desempeño de cada miembro debe ser visible y cuantificable para los demás integrantes.

Conforme a esto, en un grupo de AC los estudiantes deben entender, que son responsables tanto del aprendizaje personal como del de sus compañeros, y por tanto deben asegurarse de que todos los miembros del equipo, alcancen un dominio adecuado de lo que están intentando aprender de forma cooperativa.

El mecanismo que Johnson y Johnson (1987) defienden para conseguir esa responsabilidad individual, es la realización de pruebas individuales por las cuales los alumnos reciben una puntuación individual, y a la que recomiendan añadir, una puntuación grupal derivada de los productos que hayan logrado desarrollar como grupo.

Varios investigadores del AC concuerdan en que el tema de las recompensan grupales (la puntuación grupal que se otorga por los productos cooperativos), constituye un aspecto crítico (Riego (1990), Slavin (1985), Johnson y Johnson (1987), Holubec y Roy (1984)). Fundamentalmente las dificultades para el manejo de estas recompensas surgen debido a que, un mal manejo de ellas puede hacer cambiar el enfoque cooperativo de las actividades, tornándose en un enfoque competitivo, lo que pone en riesgo la generación de instancias de intercambio, reflexión y discusión que queremos promover y favorecer.

Respecto de esto mismo, diversos autores han señalado que el manejo de la recompensa es más simple con alumnos de corta edad y más complejo cuando son

mayores o son estudiantes que cursan niveles superiores de enseñanza. La causa de ello podría encontrarse en el hecho de que los alumnos mayores, están más acostumbrados a las recompensas y al modelo competitivo, y por tanto, consideran que se les debe recompensar por todas las actividades y trabajos que realizan durante el proceso de aprendizaje.

Las recompensas son un elemento importante de considerar en los enfoques conductistas del AC, ya que se considera que ellas son el principal estímulo para que los integrantes de un grupo, no sólo participen en él, sino que además desarrollen acciones de cooperación con el resto de los integrantes.

Existe gran desacuerdo en los criterios que se deben utilizar para aplicar los refuerzos, y existen técnicas de AC que no utilizan la estructura de recompensas.

En nuestra intervención hemos optado por un enfoque de AC sustentado en la psicología social. Deutsch (1949), define una situación social cooperativa, como aquella en la que las metas de los individuos separados van tan unidas que existe una correlación positiva entre las consecuciones o logros de sus objetivos. Un individuo alcanza su objetivo, si y sólo si también los otros miembros alcanzan el suyo (goal structure).

Así, en el diseño de esta intervención no hemos considerado el uso de recompensas. Nuestras razones para esto son las siguientes:

- Cómo señalan distintos autores, establecer los criterios para otorgar las compensaciones es complejo, en especial cuando se trata con estudiantes mayores como es nuestro caso, en que trataremos con estudiantes de enseñanza universitaria, los cuales ya poseen una experiencia, de a lo menos 13 años como alumnos de un sistema educativo que tiene un enfoque fundamentalmente competitivo.
- El enfoque cooperativo que proponemos pone el énfasis en la promoción del aprendizaje cooperativo, que significa que lo que nos interesa por sobre todo, es crear condiciones para que nuestros estudiantes aprendan cooperativamente, esto es, que las instancias de cooperación se constituyan en momentos para aprender con los otros y de los otros, por tanto usamos la cooperación para socializar el acto de aprender a resolver problemas de procesamiento de datos.

  En estos términos, nuestra intención es que las situaciones de AC nos sirvan para favorecer el aprendizaje de los distintos tipos de conocimientos, así como también el desarrollo y el perfeccionamiento de las estrategias de pensamiento que la propia tarea de resolución de problemas demanda.
- En este experimento se quiere medir la influencia que tienen las actividades de AC sobre las mejoras en el rendimiento de los estudiantes, y

complementariamente, como influyen este tipo de actividades en la satisfacción de los estudiantes.

- La intención es que el principio básico de la dinámica de las actividades sea de aprendizaje cooperativo y no de trabajo cooperativo, es decir, se pretende los estudiantes entiendan que las actividades se llevan a cabo para que cada integrante del grupo aprenda de manera más efectiva, y no que el cooperativismo se exige para resolver un conjunto de problemas o tareas. Así, las actividades cooperativas se diseñan para que contribuyan de manera más efectiva al aprendizaje individual, lo que se espera se refleje en los logros alcanzados por cada estudiante en las evaluaciones individuales que rinda durante el semestre académico.
- En a lo menos 3 semestres anteriores a esta intervención, hemos usado mecanismos de recompensas por los productos grupales, que consistían en otorgar una calificación a las tareas grupales, y a las cuales se les asignaba un cierto peso porcentual para el cálculo de la nota final de la asignatura. Sin embargo, se ha verificado que un alto porcentaje de los estudiantes (sobre el 75%), aunque obtenían buenas calificaciones grupales, éstas no se mantenían en las calificaciones obtenidas de manera individual, con lo cual para muchos estudiantes, existía una bajísima correlación entre la evaluación de recompensa obtenida grupalmente y sus rendimientos en las prueba individuales.

En síntesis, lo que hemos argumentado en este apartado es que utilizaremos el AC como estrategia para favorecer y mejorar el aprendizaje de nuestros estudiantes.

El AC, nos permite poner en práctica una visión constructivista y social del aprendizaje, con lo que pretendemos favorecer el aprendizaje y el desarrollo de las capacidades y estrategias de pensamiento, regulación y control, para resolver problemas de programación.

Los fundamentos del AC nos servirán para crear ambientes en los cuales nuestros estudiantes dispongan de mejores oportunidades para aprender, desarrollar y perfeccionar el uso y aplicación de los conocimientos de FP, para desarrollar y perfeccionar el conocimiento estratégico que se requiere, y para desarrollar y perfeccionar las capacidades para resolver los problemas.

La dimensión social, la utilizaremos para crear condiciones para que los estudiantes mediante una participación activa dentro de sus grupos, reflexionen, discutan, argumenten y defiendan sus visiones particulares respecto del uso y aplicación de los conocimientos, durante las tareas de resolución de los problemas, y que mediante esta socialización, mejoren también sus construcciones de conocimiento estratégico

y las capacidades y estrategias de pensamiento, de regulación y control, que se requieren en la propia tarea de resolver los problemas de programación.

Usaremos una adaptación del modelo de AC conocido como Learning together (aprendizaje juntos) (Johnson y Johnson, (1987)), pero no otorgaremos recompensas por el trabajo grupal, ya que lo que la intención es que mediante el AC se favorezca y potencie el aprendizaje individual de cada estudiante.

4.4. Orientaciones para el diseño del plan de desarrollo de la asignatura

Para alcanzar el logro de los conocimientos, las capacidades y las estrategias de pensamiento, de regulación y control que se requieren para resolver los problemas de programación, proponemos que el plan de desarrollo de la asignatura, se organice en torno a sesiones de actividades dirigidas a los cuatro aspectos del desarrollo de los contenidos (definidos en el apartado 3.3):

1.- Sesiones de clases expositivas; dirigidas a la presentación de los elementos teórico-formales, los procedimientos y las técnicas de la programación,
2.- Sesiones de clases expositivas para la presentación y el desarrollo del proceso de resolución para problemas tipos,
3.- Sesiones de actividades de aprendizaje cooperativo; para que los estudiantes se involucren de manera directa, intencionada y organizada en el proceso de resolución de los problemas, y
4.- Sesiones dirigidas a evaluar los aprendizajes individuales alcanzados por los estudiantes.

En las sesiones de clases expositivas dirigidas a la presentación de los elementos teórico-formales, los procedimientos y las técnicas de la programación (1º aspecto), se debe poner énfasis en la clarificación de los elementos que dan el carácter disciplinar a la programación de computadores, resaltando cómo estos elementos fundamentales de la programación de computadores, son los que dan el rigor científico y la sistematicidad al proceso completo de resolución de los problemas de programación.

Cómo se recordará, el rigor científico y la sistematicidad del proceso de resolución de estos problemas, son aspectos que han sido destacados de manera especial en el informe de las recomendaciones curriculares, enfatizándose que estos constituyen un conjunto de conocimientos cuyo aprendizaje es relevante y trascendental, en especial cuando se trata con el primer curso de esta área disciplinar. Como sea planteado en apartados anteriores, aprender el rigor disciplinar de la programación de computadores, es fundamental para abordar con buena base metodológica, las otras asignaturas de esta área de conocimientos que existen en el plan de formación profesional.

En las sesiones dirigidas a la presentación y el desarrollo del proceso de resolución de problemas (2º aspecto), la intencionalidad debe estar puesta en ejemplificar como los fundamentos teórico-formales, los procedimientos, las técnicas y las estrategias, se aplican en situaciones concretas de resolución de problemas. Estas son instancias de ejemplificación esenciales para que los estudiantes tengan un primer acercamiento a la formas de llevar a cabo la integración de los conocimientos a la hora de tratar con el desarrollo de las soluciones.

En esta misma perspectiva, es importante mostrar a los estudiantes, como se trabaja con las distintas abstracciones, y a partir de ellas, cual es el proceso de construcción y utilización de los distintos modelos con que trata el proceso de resolución de los problemas, resaltando la forma de utilizar y aplicar los procedimientos, las técnicas y las estrategias que intervienen en la construcción y utilización de los modelos. En definitiva, el énfasis debe estar puesto en el proceso de resolución y en cómo los elementos que intervienen en este proceso se van utilizando e integrando en el mismo.

En las sesiones en las que los estudiantes tratan directamente con el proceso de resolución de los problemas (3º aspecto), el énfasis debe estar puesto en el desarrollo y el perfeccionamiento de las capacidades y las estrategias de pensamiento, de control y regulación que intervienen en el proceso de resolución de los problemas.

Para estos efectos, estas sesiones deben organizarse como instancias que permitan que los estudiantes, trabajen de manera intencionada en los procesos de resolución de problemas, y que en este contexto, dispongan de las mejores oportunidades para hacerse participes activos de los procesos de pensamiento dirigidos a la reflexión, discusión, argumentación, defensa, etc. que son parte del proceso de diseño y construcción de soluciones a los problemas.

Bajo esta perspectiva, los estudiantes deben ser puestos en situaciones de desarrollo y perfeccionamiento de las estrategias cognitivas, de pensamiento y de control y regulación, que interviene en el desarrollo de la tarea intelectual de resolver estos problemas.

Estas sesiones deben constituirse en momentos que favorezcan y faciliten la adquisición, el desarrollo y el perfeccionamiento de las capacidades y habilidades para resolver problemas de procesamiento de datos y su programación, cuestiones que en definitiva, constituyen el propósito que se debe alcanzar en este tipo de cursos.

Ahora bien, para lograr estos objetivos con las sesiones de práctica directa de los estudiantes en los procesos de resolución de los problemas, ellos deben ser puestos

en situaciones que promuevan y posibiliten el AC, entendido esto como instancias que permitirán crear condiciones adecuadas para que los estudiantes adquieran, desarrollen y perfeccionen las capacidades y habilidades de las que hemos hablado.

Las actividades de AC constituyen la componente metodológica más fuerte en el desarrollo del plan de la asignatura, por lo cual, este ha de considerar tanto actividades de AC dentro de las sesiones de clases, junto con actividades de este tipo que realizarán en momentos fuera las sesiones de clases, con lo cual de alguna manera, se les estarán organizando sus actividades de estudio más allá de las sesiones de clases.

Otro de los aspectos relevantes, tiene que ver con las instancias de evaluación individual del aprendizaje de los estudiantes (4º aspectos). Estas evaluaciones deben diseñarse con el objeto de medir el logro de las capacidades y habilidades para resolver los problemas de procesamiento de datos, y para distintos niveles de complejidad de los mismos. Para evaluar de forma objetiva el logro de las capacidades y habilidades, se definirá un conjunto de competencias y sus indicadores, los cuales serán medidos mediante las pruebas individuales que se aplicaran durante el semestre académico.

En síntesis estamos proponiendo que el plan de clases debe considerar sesiones de clases expositivas para la presentación de los contenidos teórico-formales (1º aspecto), procedimentales, las técnicas, y para la presentación y explicitación de ejemplos en los cuales se usan y aplican los formalismos teóricos, los procedimientos y las técnicas, y para explicitar cómo este conjunto de elementos se integran en la propia tarea de resolución de los problemas (2º aspecto).

Una parte sustantiva de este plan de clases, estará dedicado a las actividades de AC, las que se diseñarán con el propósito de que los estudiantes desarrollen y perfeccionen la capacidad y las estrategias para resolver problemas de procesamiento de datos (3º aspecto).

Por último, el plan de clases debe considerar las sesiones para evaluar los logros en el aprendizaje individual alcanzado por los estudiantes (4º aspecto).

SÍNTESIS GENERAL DEL CAPÍTULO 4

En este apartado hemos analizado las limitaciones del método de enseñanza que se utiliza y que se ha utilizado para desarrollar la asignatura de FP. Las limitaciones se refieren esencialmente a que es un método que no se favorece el aprendizaje, el desarrollo y el perfeccionamiento de la capacidad para resolver los tipos de problemas con que trata la asignatura.

Como se han planteado, las características complejas de los conocimientos y capacidades que demanda la asignatura exige la puesta en práctica de metodologías que consigan que los estudiantes adopten un rol activo en sus procesos personales de aprendizaje, como condición para alcanzar aprendizajes de mejor calidad.

Se trata con un aprendizaje que es complejo no solo por lo tipos de conocimientos que se deben aprender, sino que por sobre todo, porque se requiere desarrollar la capacidad para utilizar estos conocimientos al aplicarlos a situaciones concretas de resolución de problemas de procesamiento de datos y la programación de computadores, contexto para el cual, además de los conocimientos, se requiere de determinadas estrategias de pensamiento, de control y regulación, que son de carácter complejo.

Así, para crear las condiciones que permitan abordar las complejidades de estos aprendizajes, es que se propone utilizar una estrategia de AC.

Se han establecido las orientaciones para el diseño del plan de desarrollo de la asignatura, el cual además de las clases expositivas para la presentación de los contenidos y para la ejemplificación de los usos de ellos, han de considerar un conjunto importante de actividades de AC, cuya intencionalidad es que los estudiantes adquieran, desarrollen y perfeccionen las capacidades, y estrategias que se han descrito.

CAPITULO 5

METODOLOGÍA DE LA INVESTIGACIÓN

5. METODOLOGÍA DE LA INVESTIGACIÓN

INTRODUCCIÓN

En este apartado abordamos los aspectos referidos al experimento que se ha llevado a cabo en esta investigación.

En estos términos exponemos los objetivos generales y específicos de la investigación, y planteamos las hipótesis de la misma.

Junto con ello describimos los participantes, el procedimiento que hemos diseñado y aplicado, ámbito en el cual describimos en detalle el plan de clases que se ha puesto en práctica con el desarrollo de la asignatura.

Seguidamente exponemos los detalles de las variables y sus medidas, y presentamos los instrumentos que hemos utilizado para la recopilación de la información.

5.1. Objetivos e hipótesis de la investigación

La presente investigación tiene como objetivo central desarrollar una intervención que nos permita mejorar los aprendizajes de los estudiantes de la asignatura de FP.

Como hemos descrito en los apartados anteriores, esta asignatura, aunque en algunos casos con distinto nombre, se ubica en el primer o segundo semestre académico del plan curricular de prácticamente todas las carreras de Ingeniería en Informática.

Además es una asignatura que tiene carácter troncal para estos planes de estudio, en el sentido de que existen en ellos varias asignaturas, para las cuales los conocimientos, capacidades y habilidades que se deben desarrollar en esta asignatura, son esenciales. Estos juicios no solo están avalados por nuestras experiencias y práctica docente, sino que también están suficientemente respaldados en varios documentos que contienen las recomendaciones curriculares.

Estos documentos son el fruto del trabajo y la reflexión en torno a los procesos formativos en las carreras de Ingeniería en Informática y sus similares, y que han sido elaborados por comisiones de expertos que se agrupan en organizaciones prestigiosas, y que han sido comunicados a la comunidad académica que trabaja y toma decisiones en el ámbito de estos planes de formación profesional.

Así también, conforme a lo que hemos expuesto, las dificultades que tienen los estudiantes con esta asignatura, se relacionan con varios aspectos entre los cuales destacan: la complejidad propia de los contenidos a aprender; las dificultades para alcanzar el conocimiento práctico para resolver los problemas de procesamiento de datos; las dificultades para aprender, desarrollar y perfeccionar determinadas capacidades y habilidades, y de ciertas estrategias cognitivas y de pensamiento que son requeridas para llevar a cabo el proceso de aplicación de los conocimientos para resolver los problemas de procesamiento de datos y la programación de computadores.

Atendiendo a estos factores y en la intención de mejorar los aprendizajes de los estudiantes, hemos desarrollado la presente investigación que en lo fundamental, apunta al diseño y puesta en práctica de una intervención al proceso de enseñanza y aprendizaje de la asignatura.

Para estos efectos se ha diseñado un plan de clases, el cual en términos diferenciadores respecto de otras formas de desarrollar la asignatura, incorpora estrategias de aprendizaje cooperativo, con el propósito de crear mejores condiciones para tratar con las dificultades que hemos señalado y consecuentemente, lograr mejores resultados en el aprendizaje de los estudiantes.

En este contexto entonces, el propósito fundamental de esta investigación es determinar el efecto que tiene el uso de estrategias y actividades de aprendizaje cooperativo (EyAAC) sobre el aprendizaje de los estudiantes de un curso de FP, en el nivel de enseñanza universitaria de pre-grado, en la carrera de Ingeniería en Informática de la Universidad de Playa Ancha. Conforme a esto, hemos establecido los siguientes objetivos.

5.1.1. Objetivos generales

- A partir del análisis de los elementos que intervienen y afectan el proceso de enseñanza y aprendizaje de los Fundamentos de Programación de computadores, elaborar y aplicar una propuesta metodológica, sustentada en el aprendizaje y el trabajo cooperativo, para mejorar el rendimiento de los estudiantes en la asignatura.
- Validar la metodología propuesta

5.1.2. Objetivos específicos

a. Realizar el análisis de los elementos que más pudieran condicionar el proceso de enseñanza y aprendizaje de los Fundamentos de Programación de Computadores
b. Proponer una metodología sustentada en el aprendizaje y el trabajo cooperativo, que favorezca el proceso de enseñanza y aprendizaje de los estudiantes.
c. Ensayar y validar la aplicabilidad de la nueva metodología en el desarrollo de la asignatura.

5.1.3. Hipótesis de trabajo

Teniendo en cuenta los objetivos de la investigación se formulan las siguientes hipótesis de trabajo:

H1.- Existen diferencias estadísticamente significativas entre el rendimiento académico de los estudiantes del grupo experimental y del grupo control después de la intervención.
Es decir, se deben evidenciar mejoras en los estudiantes del grupo experimental, que corresponde al grupo intervenido mediante el uso de actividades de aprendizaje y trabajo cooperativo respecto del grupo control sin cambio de metodología.

H2. La experiencia de aprendizaje y trabajo cooperativo es valorada positivamente por los estudiantes del grupo experimental.
Considerado a partir de que se les facilita el aprendizaje, favorece su motivación y mejora el logro de las expectativas de logro académico.

5.2. Descripción de los participantes: Población y muestra

Los participantes de esta investigación han estado conformados por cinco grupos de estudiantes de la carrera de Ingeniería en Informática de dos universidades de la Quinta Región en Chile, y que cursan la asignatura de FP.

Con estos grupos se han desarrollado dos intervenciones las que se realizaron en dos semestres distintos separados en la temporalidad por un semestre. En la primera intervención se ha tenido un grupo curso de la Universidad de Playa Ancha como grupo experimental y otro de la misma universidad como grupo control, además de un segundo grupo control de la Universidad de Valparaíso.

En la segunda intervención, el grupo experimental y el control han sido de la Universidad de Playa Ancha.

Sin embargo, para los efectos del tratamiento y el análisis de los datos se tratará con los dos grupos experimentales como uno sólo, y también como uno sólo con los tres grupos control.

Así entonces, en estos términos, el grupo experimental (GE) ha estado conformado por 61 estudiantes, en tanto el grupo control (GC) ha estado conformado por 83 estudiantes.

El grupo GE está conformado por un 47,54% de estudiantes de sexo femenino y un 52,46% de estudiantes de sexo masculino, con edades entre los 17 y los 21 años con una media de 18,48 años.

Para este grupo de estudiantes se tiene que sus puntajes ponderados de la Prueba de Selección Universitaria (PSU) de ingreso a la carrera varían entre un mínimo de 515,70 y un máximo de 575,80 puntos, con una media de 544,1934 una desviación típica de 14,33669 y una varianza de 205,541.

Para este mismo (GE), 63.93% corresponden a estudiantes que cursan por primera vez la asignatura y un 32.79% corresponden a estudiantes que la cursan por segunda vez, y un 3.28% corresponden a estudiantes que cursan por tercera vez la asignatura.

Por su parte, el grupo GC está compuesto por un 40,96% de estudiantes mujeres, y un 59,04% de estudiantes hombres. Sus edades fluctúan entre los 17 y los 24 años con una media de 18,63 años.

Los estudiantes de este grupo, ingresaron a la carrera con puntajes ponderados de la PSU que varían entre un mínimo de 515,70 y un máximo de 626,10 puntos, con una media de 542,9807 una desviación típica de 18,62645 y una varianza de 346,945.

Así también, para el grupo control se tiene que un 50,60% corresponden a estudiantes que cursan por primera vez la asignatura, un 34,94% corresponden a estudiantes que la cursan por segunda vez, y un 14,46% corresponden a estudiantes que la cursan por tercera vez.

Conforme a estos parámetros, se aprecia un alto grado de similitud entre los grupos experimental y control.

5.3. Caracterización de la investigación

La presente investigación aunque corresponde a un estudio centrado en un caso, relativo al desarrollo durante un semestre completo, de la asignatura de Fundamentos de Programación, para estudiantes de Ingeniería en Informática en la UPLA, se puede considerar como cuasi experimental al poder atribuirles las características propias de este tipo de invetsigación.

Trata con un grupo experimental, el cual ha sido intervenido mediante el uso de una metodología de enseñanza y aprendizaje, en la que se han integrado un número importante de actividades de aprendizaje y trabajo cooperativo. Este grupo ha estado compuesto por estudiantes de la Universidad de Playa Ancha, Valparaíso, Chile.

Con el objeto de contrastar los resultados de la intervención, se ha utilizado un grupo control, compuesto por estudiantes de la misma asignatura y carrera, para quienes, la asignatura ha sido desarrollada usando la metodología tradicional, esto es, clases expositivas y de ejercicios.

El grupo de estudiantes del grupo control, ha estado compuesto por estudiantes de la Universidad de Playa Ancha y de la Universidad de Valparaíso, donde todos cursan la misma asignatura de la misma titulación.

La investigación trata con aspectos cuantitativos y cualitativos, ya que por una parte, trata con la mejora en los niveles de aprobación de los estudiantes y con la mejora en el logro de las competencias para resolver problemas de procesamiento de datos. En tanto en el segundo aspecto, trata con la calidad del trabajo cooperativo y sus efectos sobre el aprendizaje, y trata también con el nivel de satisfacción de los estudiantes respecto de la metodología utilizada en el desarrollo de la asignatura.

5.4. Diseño de la investigación

El diseño del experimento se fundamenta en el diseño del plan de clases que se pondrá en práctica, instancia en la cual se integran las clases expositivas de presentación de contenidos, las actividades cooperativas dentro de las sesiones de clases, como fuera de ellas, y las actividades de evaluación propias del desarrollo de una asignatura.

A continuación se presentan los aspectos generales del diseño del plan de clases que hemos puesto en práctica en la intervención del grupo experimental.

Primera Semana:

Para un primer momento (tabla 5), el propósito fundamental de esta sesión es exponer y explicitar a los estudiantes los propósitos generales de la asignatura.

Tipo de actividad:	Presentación y análisis de información.
Duración:	25 minutos
Aspectos a tratar:	Presentación y análisis de los objetivos y los contenidos de la asignatura. Análisis y discusión de los propósitos de la asignatura. Análisis y discusión de los tipos de aprendizaje que deben alcanzar los estudiantes. Calendarización del tratamiento de los contenidos y las evaluaciones.

Tabla 5: Primera Semana, 1º Período, primer momento

En esta perspectiva se presentan y explicitan los objetivos y contenidos curriculares que están definidos para la asignatura, la forma en que éstos están organizados en las unidades temáticas, y los aprendizajes que se deben lograr. Para este propósito, se analizan las capacidades, habilidades y destrezas que se han de lograr.

En un segundo momento (tabla 6), se presenta y explicita la calendarización del desarrollo de la asignatura, detallando los momentos en que se trabajará sobre el tratamiento de los contenidos, los momentos en que trabajará en las actividades de aprendizaje cooperativo, y las instancias de evaluación del logro de los aprendizajes, y finalmente se presentan los requisitos de calificación para la aprobación de la asignatura.

Tipo de actividad:	Presentación y análisis de información; Conformación de grupos cooperativos
Duración:	25 minutos
Aspectos a tratar:	Presentación de la modalidad de trabajo. Presentación y análisis de los aspectos generales del aprendizaje cooperativo desde la perspectiva de los estudiantes. Conformación de los grupos de aprendizaje cooperativo. Lectura y firma del contrato didáctico.

Tabla 6: Primera Semana, 1º Período, segundo momento

Se expone y explicita la modalidad de trabajo que se utilizará para desarrollar la asignatura, poniendo énfasis en la importancia que tiene el compromiso tanto individual como colectivo, y cómo este compromiso contribuirá a las mejoras en el aprendizaje individual.

A continuación, se presenta y explicita los elementos prácticos del trabajo cooperativo, como práctica fundamental para dar paso al aprendizaje cooperativo, resaltando los beneficios que esto tiene tanto para el aprendizaje individual, como para el desarrollo de ciertas competencias que consideramos esenciales para los profesionales de las CcC. En este contexto se analizan los fundamentos del

aprendizaje cooperativo y los elementos de compromiso individual y grupal que hacen que esta forma de trabajo sea efectiva.

Luego de esto, se establecen las reglas para la conformación de los grupos cooperativos, seguido de ello, los estudiantes leen y firman el contrato didáctico que regirá el compromiso de los integrantes del grupo cooperativo. Junto con ello, los estudiantes eligen el jefe de grupo que tendrá la responsabilidad de liderar y animar el trabajo de su grupo.

Aunque el requisito es que la conformación de los integrantes de cada grupo se debe mantener durante todo el semestre académico, cada grupo tiene la libertad de cambiar a su jefe de grupo en atención a las evaluaciones de desempeño que se realicen.

Aunque las recomendaciones generales respecto a la conformación de los grupos cooperativos, señalan que estos deben formarse con integrantes que sean heterogéneos respecto de sus capacidades intelectuales, al no disponer de información concreta en relación con esto, se permitirá que cada grupo se defina por criterios de afinidad entre sus integrantes, teniendo en cuenta que cada grupo estará compuesto por a lo más tres estudiantes, y que la conformación de los mismos no habrá de cambiarse durante todo el semestre académico.

Tipo de actividad:	Desarrollo de la evaluación diagnósticas
Duración:	40 minutos
Aspectos a tratar:	Evaluación diagnóstica; primera parte.

Tabla 7: Primera Semana, 1º Período, tercer momento

El objetivo de esta instancia (tabla 7), es iniciar la aplicación del test de evaluación diagnostica, para ello se explican los propósitos de ella y la forma en que llevará cabo.

Mientras se desarrolla de esta primera parte, se monitorea el trabajo de los estudiantes.

En esta primera parte se desarrollará hasta el ítem 2.5.3 del instrumento de evaluación.

2º Período

Tipo de actividad:	Desarrollo de la evaluación diagnósticas
Duración:	90 minutos
Aspectos a tratar:	Evaluación diagnóstica; segunda parte.

Tabla 8: Primera Semana, 2º Período

En esta segunda parte los estudiantes trabajan en el desarrollo de los ítems desde 2.6 al 4.7, y se continúa con el monitoreo.

Tipo de actividad:	Análisis de los resultados de la evaluación diagnósticas
Duración:	45 minutos
Aspectos a tratar:	Análisis general de los resultados de la evaluación diagnóstica

Tabla 9: Primera Semana, 3º Período, primer momento

Si bien, como lo hemos señalado, curricularmente la asignatura no tiene pre-requisito, consideramos importante analizar los resultados del test de diagnóstico, el cual vemos desde dos perspectivas. La primera referida a la obtención de información respecto del dominio que poseen los estudiantes sobre ciertos conceptos y procedimientos, lo cual permitirá disponer de información relevante para orientar el tratamiento de los nuevos contenidos, en especial, cuando este tratamiento requiera de la disponibilidad e integración que los conocimientos considerados en el diagnóstico.

La segunda está referida al hecho de que si bien estos conocimientos deberían haber sido aprendidos en los niveles de enseñanza regular anterior a la universitaria, éstos podrían estar un poco olvidados, con lo cual a partir de los resultados arrojados por el diagnóstico, se podrán hacer recomendaciones a los estudiantes para que actualicen el manejo de aquellos conocimientos que se muestran deficitarios.

Tipo de actividad:	Clase expositiva
Duración:	45 minutos
Contenidos a tratar:	Concepto de algoritmo. Algoritmos y programas.

Tabla 10: Primera Semana, 3º Período, segundo momento

Al presentar el concepto de algoritmo, es importante explicitar tanto el concepto mismo como la utilidad metodológica que está implícita en el formalismo conceptual de algoritmo, clarificando cómo la programación de computadores hace uso de este formalismo, para abordar el diseño de las soluciones a problemas de procesamiento de datos y su correspondiente programa de solución.

Así, el tratamiento de estos contenidos ha de orientarse al logro de los siguientes aprendizajes:

- Comprender los formalismos del concepto de algoritmo,
- Reconocer los algoritmos como una herramienta eficaz para diseñar y expresar la solución a un problema.
- Relacionar los conceptos de algoritmo y programación de computadores.
- Comprender el concepto de algoritmo y su utilidad para el diseño de soluciones a problemas de procesamiento de datos y en la implementación del programa de solución

Así también, este mismo tratamiento ha de orientarse al desarrollo de la siguiente competencia específica:

Comprender y explicar el concepto de algoritmo y su importancia para el diseño e implementación de soluciones computacionales a problemas de procesamiento de datos.

Para ejemplificar la utilidad del concepto de algoritmo, se sugiere dar ejemplos basados en las secuenciación lógica de acciones y operaciones para resolver situaciones sencillas de cálculo matemático.

Segunda Semana:

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenido a tratar:	Metodología de la programación

Tabla 11: Segunda Semana, 1º Período

En el 1º período de esta sesión (tabla 11), se abordan los aspectos formales de la metodología de la programación, por tanto debe orientarse a la explicitación de los fundamentos de la metodología, de las etapas de la misma, contextualizándolas en el propio proceso de resolución de los problemas. En esta perspectiva el profesor expone y ejemplifica los fundamentos teóricos y los procedimientos de aplicación y uso de:

- La metodología de programación,
- La etapa de análisis.
- La etapa de diseño.
- La etapa de codificación.
- La etapa de prueba y depuración

Junto con la presentación y clarificación de los fundamentos teóricos de estos contenidos, se debe explicitar la justificación metodológica de cada etapa y de los productos que cada una de ella ha de generar para el proceso de resolución. Además, se ha de ejemplificar el uso y aplicación de la metodología y de las etapas de la misma, poniendo énfasis en la importancia de la integración de ellas a la hora de aplicarlas al proceso de resolución de los problemas de procesamiento de datos.

Para la ejemplificación se han de presentar casos o situaciones sencillas de generalizaciones de problemas matemáticos de cálculo, mediante los cuales se ilustre tanto la justificación de cada etapa, como los productos que en ellas se generan, y la forma en que estos se integran y sirven para la resolución final del problema.

Por tanto el tratamiento de estos contenidos debe apuntar al logro de objetivos de aprendizaje referidos a la comprensión de:

- Los fundamentos teóricos de la metodología de la programación, y entenderla como el recurso metodológico disciplinar que se aplica a la resolución de los problemas de procesamiento de datos.
- Los fundamentos teóricos de la etapa de análisis, y el producto que ha de generar su aplicación,
- Los fundamentos teóricos de la etapa de diseño, y el producto que ha de generar su aplicación,
- Los fundamentos teóricos de la etapa de codificación, y el producto que ha de generar su aplicación,
- Los fundamentos teóricos de la etapa de prueba y depuración, y el producto que ha de generar su aplicación,
- Del carácter integrador que debe el proceso de desarrollo y aplicación de las etapas de la metodología y sus productos intermedios.

Con lo cual se trata que los estudiantes desarrollen la capacidad para comprender los propósitos, los procedimientos y productos de cada etapa de la metodología de la programación y de su conjunto, como método para la búsqueda e implementación de soluciones de los problemas de programación de computadores, desde una perspectiva disciplinar.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Proceso de análisis Diseño de algoritmos

Tabla 12: Segunda Semana, 2º Período

En el 2º período (tabla 12), se abordan desde la dimensión práctica y aplicativa el desarrollo de las etapas de análisis de los problemas y la etapa de diseño de soluciones.

Para estos efectos se tratara con problemas de cálculo matemáticos simples, partiendo con la aplicación sobre problemas que requieran el análisis y el diseño de algoritmos secuenciales, para luego dar paso a problemas que requieran del análisis y el diseño de algoritmos de estructuras de tipos selectivas simples y dobles.

En el tratamiento de los aspectos prácticos se debe explicitar como los conocimientos matemáticos y los metodológicos se integran para aplicarlos a la resolución de los problemas en el desarrollo de estas dos etapas.

En esta perspectiva se debe explicitar la integración de conocimientos matemáticos para dar forma a la definición del espacio del problema y cómo el análisis de éste, da origen a la especificación del mismo, para luego evidenciar cómo a partir de la especificación, se trabaja en el diseño de la solución.

Así también se debe hacer explícita la forma en que a partir del diseño de la secuencia de pasos para resolver el problema se pasa a la expresión del algoritmo como diagrama de flujo y como seudocódigo.

Teniendo en cuenta que la representación de algoritmos es un recurso y una herramienta fundamental dentro de la programación de computadores, al tratar con la representación algorítmica de las soluciones, se debe poner especial énfasis en los aspectos en la clarificación de los aspectos formales de las representaciones y el uso del lenguaje algorítmico de cada forma de representación.

Con esto queremos señalar, que se debe explicitar tanto las formas de representación de los algoritmos, como el significado de estas formas y de sus estructuras.

También es importante explicitar cómo se interpreta y se trata con el lenguaje algorítmico tanto en los diagramas de flujo como en los seudocódigos que se implementen, para los cual se sugiere trabajar con las tablas de ruteo de las representaciones algorítmicas que se elaboren para las soluciones.

En un ámbito más general, se debe explicitar cómo operan los procedimientos metodológicos para cada problema particular, en la intención de que se entienda que el método de solución como procedimiento, es aplicable al diseño de soluciones para cualquier problema, sin embargo, la propia solución de un cierto problema particular no es completamente aplicable a cualquier otro.

La ejemplificación debe hacerse usando problemas distintos y de complejidad creciente.

En esta perspectiva se tratan los siguientes contenidos:

- La etapa de análisis y su aplicación a la resolución de problemas.
- La etapa de diseño de soluciones y su aplicación a la resolución de problemas
- Elementos de la forma de representación algorítmica en diagrama de flujo.
- Elementos de la forma de representación algorítmica en seudocódigo.
- Uso de las formas de representación algorítmicas secuenciales y selectivas.
- Uso del lenguaje algorítmico.
- Equivalencias entre las formas de representación algorítmicas.
- Ventajas y desventajas de las distintas formas de representación

El tratamiento de estos contenidos se ha de orientar al logro de los siguientes objetivos de aprendizaje:

- Aplicar los formalismos de la etapa de análisis en la tarea resolución de problemas.

- Aplicar los formalismos de la etapa de diseño en la tarea de resolución de problemas.
- Reconocer las características de las distintas formas de representación algorítmica.
- Reconocer la simbología que es propia de cada forma de representación algorítmica.
- Comprender e interpretar representaciones algorítmicas.
- Diseñar y representar algoritmos de solución, en diagramas de flujo y seudocódigo.
- Usar el lenguaje algorítmico para interpretar y analizar representaciones algorítmicas.
- Relacionar distintas formas de representación algorítmicas
- Describir las ventajas y desventajas de las distintas formas de representación algorítmica.

Las competencias específicas a desarrollar en los estudiantes son:

A.- Capacidad para aplicar los formalismos de la etapa de análisis.
B.- Capacidad para realizar el proceso de análisis de problemas.
C.- Capacidad para aplicar los formalismos de la etapa de diseño de soluciones.
D.- Habilidad para diseñar soluciones a problemas que requieren de la implementación de algoritmos secuenciales y selectivos.
E.- Comprender los formalismos de las representaciones algorítmicas y su lenguaje.
F.- Habilidad para interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.

Teniendo en cuenta que esta es la primera actividad de aprendizaje cooperativo (tabla 13 en la página siguiente), se debe poner especial atención a la explicitación de los objetivos de la actividad, así como de las tareas de desarrollar por los estudiantes, las condiciones para ello, y los productos que cada grupo de cooperativo deberá entregar al finalizar el tiempo de la actividad.

Así también, durante el desarrollo del trabajo de los estudiantes, se debe prestar atención a las acciones de seguimiento y monitoreo que se consideren necesarias para garantizar tanto el buen ambiente del trabajo, como el buen desempeño de los integrantes de cada grupo.

La actividad se orienta al trabajo en las tareas de resolución de problemas para las etapas de análisis y diseño de soluciones, para ello se entregará a los grupos cooperativos el enunciado de tres problemas para los cuales deben elaborar la especificación y el diseño de los algoritmos de solución.

Tipo de actividad:	Trabajo cooperativo

Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Algoritmos y programas. Análisis del problema. Diseño de algoritmos de solución. Representación algorítmica de la solución.	
Tipo de tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas secuenciales y selectivas simples.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	10 minutos
	Trabajo de los grupos cooperativos;	40 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	15 minutos
Acciones de los estudiantes:	Lectura interpretativa del enunciado del problema,	3 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos
	Identificar los conocimientos implicados en el problema,	6 minutos
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos
	Elaborar la especificación del problema y el algoritmo de solución,	8 minutos
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos
Competencias a desarrollar:	Generales para el trabajo grupal; Capacidad para: Trabajar de manera activa en su grupo cooperativo. Presentar, argumentar, y discutir ideas y propuestas de solución a los problemas. Específicas disciplinares; Capacidad para: Aplicar los formalismos de la etapa de análisis. Realizar el proceso de análisis de problemas. Aplicar los formalismos de la etapa de diseño de soluciones. Diseñar soluciones a problemas que requieren de la implementación de algoritmos secuenciales y selectivos simples. Comprender los formalismos de las representaciones algorítmicas y su lenguaje. Interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.	

Tabla 13: Segunda Semana, 3º Período

A partir del análisis colectivo del desarrollo el profesor realizará una categorización de las dificultades que expresen los estudiantes, lo que se tendrá en cuenta para hacer recomendaciones para superarlas.

Complementariamente, se entregará a los estudiantes una guía con el enunciado de nueve problemas, que deberán resolver mediante trabajo cooperativo (tabla 14 en la página siguiente), a realizar fuera del tiempo de la clase.

Tercera Semana:

Se profundiza la explicitación de los contenidos y el uso de la metodología de la programación para las etapas de análisis y diseño algorítmico de las soluciones (tabla 15 en la página siguiente), tratando con la ejemplificación para problemas de mayor complejidad, para lo cual, se trabajará sobre la resolución de problemas que corresponden a generalizaciones de problemas matemáticos, que demanden la definición de especificaciones con más detalles, además de la inclusión en ella de restricciones y limitaciones que han de ser consideradas para el diseño de la solución.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Algoritmos y programas. Análisis del problema. Diseño de algoritmos de solución. Representación algorítmica de estructura secuencial y selectiva simple		
Descripción de la tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas secuenciales y selectivas simples.	Cantidad de problemas planteados:	9
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	3 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	6 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Elaborar la especificación del problema y el algoritmo de solución,	8 minutos	
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Generales para el trabajo grupal; Capacidad para: Trabajar de manera activa en su grupo cooperativo. Presentar, argumentar, y discutir ideas y propuestas de solución a los problemas. Específicas disciplinares; Capacidad para: Aplicar los formalismos de la etapa de análisis. Realizar el proceso de análisis de problemas. Aplicar los formalismos de la etapa de diseño de soluciones. Diseñar soluciones a problemas que requieren de la implementación de algoritmos secuenciales y selectivos simples. Comprender los formalismos de las representaciones algorítmicas y su lenguaje. Interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.		

Tabla 14: Segunda Semana, 3º Período, actividad complementaria

Así también, el diseño de las soluciones algorítmicas para estos problemas ha de involucrar el uso de varias condiciones lógicas por lo que las representaciones algorítmicas deberán tener un mayor número de estructuras de control selectivas tanto simples, como dobles y anidadas.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Representación algorítmica de estructura simple, doble y anidada. Proceso de análisis Diseño de algoritmos

Tabla 15: Tercera Semana, 1º período

Se debe continuar resaltando que lo importante es aprender a aplicar y usar los procedimientos metodológicos, además, en virtud de la mayor complejidad de los algoritmos de solución, se debe continuar con las acciones dirigidas a explicitar el uso del lenguaje algorítmicos, y con el uso de las técnicas de ruteo de los algoritmos, para trabajar los aspectos de comprensión y análisis de los algoritmos, y del lenguaje relativo a cada forma de representación.

En la siguiente sesión (2º período en tabla 16) se realiza la segunda actividad cooperativa, y en ella se deben enfatizar los aspectos de organización de los grupos con miras a conseguir la optimización de los desempeños del trabajo de los integrantes.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Algoritmos y programas. Análisis del problema. Diseño de algoritmos de solución. Representación algorítmica de la solución.	
Tipo de tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas selectivas simples, dobles y anidaciones de estas estructuras.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	10 minutos
	Trabajo de los grupos cooperativos;	40 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	15 minutos
Acciones de los estudiantes:	Lectura interpretativa del enunciado del problema,	3 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos
	Identificar los conocimientos implicados en el problema,	6 minutos
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos
	Elaborar la especificación del problema y el algoritmo de solución,	8 minutos
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos
Competencias a desarrollar:	Generales para el trabajo grupal; Capacidad para: Trabajar de manera activa en su grupo cooperativo. Presentar, argumentar, y discutir ideas y propuestas de solución a los problemas. Específicas disciplinares; Capacidad para: Aplicar los formalismos de la etapa de análisis. Realizar el proceso de análisis de problemas. Aplicar los formalismos de la etapa de diseño de soluciones. Diseñar soluciones a problemas que requieren de la implementación de algoritmos selectivos simples, dobles y anidaciones de estas estructuras. Comprender los formalismos de las representaciones algorítmicas y su lenguaje. Interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.	

Tabla 16: Tercera Semana, 2º período

Para esto, junto con explicitar los objetivos de la actividad, las tareas a desarrollar, y los productos que debe generar cada grupo cooperativo, se hará mención a aquellos aspectos relevantes del análisis de dificultades y las sugerencias para resolverlas, que resultaron de la primera actividad cooperativa.

En esta perspectiva, se trabaja sobre la profundización de las competencias específicas anteriores.

La actividad anterior se complementa con la que se muestra en la siguiente tabla, la que los estudiantes deberán realizar fuera del horario de clases.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Algoritmos y programas. Análisis del problema. Diseño de algoritmos de solución. Representación algorítmica de la solución.		
Descripción de la tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas selectivas simples y dobles.	Cantidad de problemas planteados:	9
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	3 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	6 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Elaborar la especificación del problema y el algoritmo de solución,	8 minutos	
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Generales para el trabajo grupal; Capacidad para: Trabajar de manera activa en su grupo cooperativo. Presentar, argumentar, y discutir ideas y propuestas de solución a los problemas. Específicas disciplinares; Capacidad para: Aplicar los formalismos de la etapa de análisis. Realizar el proceso de análisis de problemas. Aplicar los formalismos de la etapa de diseño de soluciones. Diseñar soluciones a problemas que requieren de la implementación de algoritmos selectivos simples y dobles. Comprender los formalismos de las representaciones algorítmicas y su lenguaje. Interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.		

Tabla 17: Tercera Semana, 2º período, actividad complementaria

Con esta actividad (tabla 18 en la página siguiente), intentamos profundizar y perfeccionar las capacidades para tratar con el análisis y el diseño de algoritmos de solución para un problema que requiera del uso de estructuras dobles y anidaciones de estas estructuras, por tanto, trata los mismos aspectos de las dos anteriores, para lo cual se propondrá a los estudiantes un problema que demanden la construcción de especificaciones más complejas, que demanden el uso de más restricciones y

limitaciones para su solución, y que sus algoritmos impliquen el uso de varias estructuras selectivas simples y dobles y anidaciones de las mismas.

Además se intenta que los estudiantes profundicen y perfeccionen las competencias referidas a las capacidades para trabajar cooperativamente.

La actividad se complementa con la que se muestra en la tabla 19 (en la página 250), la que se entrega para ser desarrollada por los grupos fuera del horario de clases.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Algoritmos y programas. Análisis del problema. Diseño de algoritmos de solución. Representación algorítmica de la solución.	
Tipo de tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas secuenciales y selectivas simples.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	10 minutos
	Trabajo de los grupos cooperativos;	40 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	15 minutos
Acciones de los estudiantes:	Lectura interpretativa del enunciado del problema,	3 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos
	Identificar los conocimientos implicados en el problema,	6 minutos
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos
	Elaborar la especificación del problema y el algoritmo de solución,	8 minutos
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos
Competencias a desarrollar:	Generales para el trabajo grupal; Capacidad para: Trabajar de manera activa en su grupo cooperativo. Presentar, argumentar, y discutir ideas y propuestas de solución a los problemas. Específicas disciplinares; Capacidad para: Aplicar los formalismos de la etapa de análisis. Realizar el proceso de análisis de problemas. Aplicar los formalismos de la etapa de diseño de soluciones. Diseñar soluciones a problemas que requieren de la implementación de algoritmos secuenciales y selectivos simples. Comprender los formalismos de las representaciones algorítmicas y su lenguaje. Interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.	

Tabla 18: Tercera Semana, 3° período

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Algoritmos y programas. Análisis del problema. Diseño de algoritmos de solución. Representación algorítmica de la solución.		
Descripción de la tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas selectivas simples, dobles y anidaciones de estas estructuras.	Cantidad de problemas planteados:	9
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	3 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	6 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Elaborar la especificación del problema y el algoritmo de solución,	8 minutos	
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Generales para el trabajo grupal; Capacidad para: Trabajar de manera activa en su grupo cooperativo. Presentar, argumentar, y discutir ideas y propuestas de solución a los problemas. Específicas disciplinares; Capacidad para: Aplicar los formalismos de la etapa de análisis. Realizar el proceso de análisis de problemas. Aplicar los formalismos de la etapa de diseño de soluciones. Diseñar soluciones a problemas que requieren de la implementación de algoritmos selectivos simples, dobles y anidaciones de estas estructuras. Comprender los formalismos de las representaciones algorítmicas y su lenguaje. Interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.		

Tabla 19: Tercera Semana, 3º período, actividad complementaria

Cuarta Semana:

Tipo de actividad:	Primera Evaluación Acumulativa
Duración:	90 minutos
Contenidos a evaluar:	Metodología de programación. Algoritmos y programas. Representación algorítmica

Tabla 20: Cuarta Semana, 1º período

En esta primera evaluación (tabla 20), se medirán en un nivel inicial, el logro de las siguientes competencias.

- Aplicar los procedimientos para desarrollar las etapas de análisis y diseño de la metodología,
- Elaborar especificaciones de problemas de procesamiento de datos,
- Diseñar algoritmos de solución,
- Utilizar el lenguaje algorítmico y las formas de representación algorítmicas.

Tipo de actividad:	Análisis de resultados de la primera evaluación.
Duración:	45 minutos

Tabla 21: Cuarta Semana, 2º período, primer momento

Con el propósito de sacar el mejor provecho posible de esta instancia de revisión y análisis de resultados (tabla 21), previamente se realizará una categorización de las deficiencias en el manejo de los conocimientos y competencias que presentaron los estudiantes.

En esta categorización se tratarán de encontrar además relaciones entre las deficiencias de distintas categorías, con el propósito de establecer influencias de unas deficiencias sobre otras.

Así también se hace conveniente establecer cuales deficiencias fueron las mayormente observadas, con el objeto de establecer cuáles de ellas, representan errores que se cometieron con más frecuencia.

Usando esta información, en el desarrollo de la sesión de revisión y análisis de los resultados, el profesor presentará el desarrollo de la resolución de cada problema de la prueba, e irá haciendo mención a las dificultades detectadas, al tiempo que provee de retroalimentación con miras a superar las deficiencias apreciadas, dando espacio además, para la intervención de los estudiantes ya sea para realizar consultas, precisiones o comentarios.

Tipo de actividad:	Clase expositiva
Duración:	45 minutos
Contenido a tratar:	El lenguaje de programación

Tabla 22: Cuarta Semana, 2º período, segundo momento

En esta segunda parte del período (tabla 22), se inicia el tratamiento de los contenidos referidos a la cuarta etapa de la metodología, para lo cual se abordará el concepto de lenguaje de programación, una breve descripción de los tipos de lenguajes de programación, y la caracterización del lenguaje C.

Luego se presentará el concepto de tipo de datos, y los conceptos de variable y constante. En este contexto es importante clarificar con ejemplos estos conceptos, estableciendo la asociación de ellos y las representaciones algorítmicas presentadas en clases como ejemplos.

A continuación se explicitarían los operadores aritméticos y lógicos, y de asignación que utiliza el lenguaje C

Objetivos de aprendizaje que se han de perseguir con el tratamiento de estos contenidos son:

- Comprender las características y recursos de un lenguaje de programación,
- Reconocer tipos de datos,
- Declarar tipos de datos simples,
- Utilizar operadores aritméticos,

- Utilizar operadores de asignación, y
- Utilizar operadores lógicos.

Con lo cual las competencias a desarrollar en los estudiantes son:

- Declarar y utilizar correctamente los tipos de datos simples, y
- Utilizar correctamente la jerarquía de operadores.

Para el logro de estas competencias se mostrarán ejemplos de uso de declaraciones de variables y operadores, relacionándolos con los ejemplos de diseños algoritmos en sus correspondientes representaciones, que fueron tratadas en clases.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de la programación, Implementación de programas

Tabla 23: Cuarta Semana, 3º período

Habiendo tratado los aspectos formales e iniciales del lenguaje de programación, corresponde ahora explicitar la práctica referida a la implementación de programas.

Para iniciar este proceso, se trabajará sobre la implementación de algoritmos en seudocódigo sencillos que fueron presentados y trabajados en clases. Así entonces se trabajará primero codificando algoritmos secuenciales, que requieran del uso de unas pocas variables, constantes y sentencias de entrada y salida.

Complementariamente, se han de explicitar las funcionalidades básicas del entorno de programación, tales como la funcionalidad de escritura, grabación y recuperación de archivos con código de programas, y las funcionalidades referidas a la compilación y ejecución de un programa, ámbito en el cual es importante además, clarificar el significado de los mensajes de error más comunes como los referidos a errores de sintaxis, y de declaración y uso de variables y constantes, y como corregirlos.

En la presentación, se debe hacer explícita la estructura básica que debe tener un programa en C, la declaración y manejo correcto de las variables, las constantes y los operadores, y la forma de utilizar las sentencias de entrada y salida de datos para cada tipo de dato simple. Así también se debe clarificar los aspectos sintácticos y semánticos del lenguaje.

Conforme a esto se tratan los contenidos referidos a la estructura básica de un programa en C, las declaraciones y el uso de variables y constantes, el uso de los operadores de asignación y los operadores aritméticos, y los aspectos de sintaxis y semántica en la escritura de las líneas de un programa y las sentencias.

Los objetivos de aprendizaje en términos de capacidades a lograr por los estudiantes corresponden a las siguientes:

- Aplicar correctamente la estructura de un programa en C,
- Declarar y usar correctamente tipos de datos simples,
- Utilizar correctamente operadores de asignación y aritméticos,
- Utilizar sentencias de entrada y salida,
- Utilizar estructuras secuenciales,
- Implementar programas a partir de un algoritmo que tiene estructura secuencial, y
- Utilizar las funcionalidades básicas del entorno de programación.

Y las competencias a desarrollar en los estudiantes son:

- Declarar y utilizar correctamente los tipos de datos simples
- Utilizar correctamente la jerarquía de operadores
- Utilizar correctamente la estructura de programación secuencial.
- Construir e implementar programas a partir de su representación algorítmica.
- Usar el entorno de programación.

Quinta Semana:

En el primer período se desarrolla una actividad de trabajo cooperativo con el objeto de que los estudiantes trabajen los aspectos aplicativos de los conocimientos y el desarrollo de las competencias.

Dicha actividad se muestra en la tabla 24 en la siguiente página.

Esta actividad se complementa con la que se muestra en la tabla 25 (en la página 255), la que los los estudiantes realizarna fuera del horario de clases.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Implementación de programas.	
Tipo de tarea:	Implementar programas a partir de su representación algorítmica, y definir y usar casos de prueba a partir de la especificación del problema.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar la representación algorítmica a partir de su seudocódigo, e identificar las variables y tipos que deben considerarse para la implementación.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del seudocódigo y de la identificación de variables y sus tipos.	5 minutos
	Individualmente, codificar el algoritmo.	10 minutos
	Presentar, argumentar y defender la codificación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor codificación del programa	8 minutos
	Escribir y probar el programa en el computador.	12 minutos
Competencias a desarrollar:	Generales para el trabajo grupal; Capacidad para: Trabajar de manera activa en su grupo cooperativo. Presentar, argumentar, y discutir ideas y propuestas de solución a los problemas. Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar algoritmos de solución para problemas que requieran el uso de estructuras secuenciales, selectivas simples y dobles Declarar y utilizar correctamente los tipos de datos simples, Utilizar correctamente la jerarquía de operadores, Utilizar correctamente las estructuras de programación, Construir e implementar programas a partir de su representación algorítmica.	

Tabla 24: Quinta Semana, 1º período

En el segundo período de esta quinta semana (tabla 26, en la página siguiente), se trabajará sobre los aspectos prácticos de la implementación de algoritmos como programa, usando los ejemplos vistos en clases, pero incorporando la codificación de algoritmos de tipo selectivos simples y dobles.

En esta perspectiva es importante explicitar la codificación y el uso de los operadores lógicos y las expresiones lógicas.

Además, se tratarán los aspectos relativos a la prueba y depuración de programas, para lo cual se deberá explicitar el concepto de casos de pruebas de un programa, estableciendo la relación entre las especificaciones del problema y la definición y uso de casos de prueba.

En esta perspectiva, se debe enfatizar los aspectos referidos a la identificación y corrección de errores. Para estos efectos se deben proponer variantes de problemas ya resueltos, para los cuales se pide a los estudiantes que desarrollen las cuatro etapas de la metodología.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Implementación de programas.		
Descripción de la tarea:	Realizar la implementación de programas a partir del algoritmo en seudocódigo.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Interpretación individual de la representación algorítmica a partir de su seudocódigo, e identificación de las variables y tipos que deben considerarse para la implementación,	3 minutos	
	Discusión y argumentación grupal para establecer consensos respecto interpretación del seudocódigo y de la identificación de variables y sus tipos.	7 minutos	
	Codificación Individual del algoritmo.	10 minutos	
	Presentar, argumentar y defender la codificación del programa,	10minutos	
	Discutir y argumentar elementos para establecer la mejor codificación del programa.	5 minutos	
	Escribir y probar el programa en el computador.	10 minutos	
Competencias a desarrollar:	Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar algoritmos de solución para problemas que requieran el uso de estructuras secuenciales, selectivas simples y dobles Declarar y utilizar correctamente los tipos de datos simples, Utilizar correctamente la jerarquía de operadores, Utilizar correctamente las estructuras de programación Construir e implementar programas a partir de su representación algorítmica.		

Tabla 25: Quinta Semana, 1º período, actividad complementaria

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Implementación y prueba de programas

Tabla 26: Quinta Semana, 2º período

Por tanto los objetivos de aprendizaje en términos de capacidades a lograr por los estudiantes son:

- Utilizar operadores lógicos,
- Utilizar estructuras selectivas simples y dobles.
- Implementar programas a partir de un algoritmo que tiene estructura selectiva simple y dobles, y
- Aplicar las etapas de la metodología a la resolución de problemas.

Y las competencias a desarrollar en los estudiantes son:

- Declarar y utilizar correctamente los tipos de datos simples,
- Utilizar correctamente la jerarquía de operadores,
- Codificar algoritmos que poseen estructuras selectivas simples, dobles,
- Construir e implementar programas a partir de su representación algorítmica, y
- Definir y usar casos de prueba.

Con el objeto de aplicar los aprendizajes y desarrollar las capacidades, se desarrolla la actividad cooperativa que se muestra en la siguiente.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Implementación y prueba de programas.	
Tipo de tarea:	Implementar programas a partir de su representación algorítmica, y definir y usar casos de prueba a partir de la especificación del problema.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar la representación algorítmica a partir de su seudocódigo, e identificar las variables y tipos que deben considerarse para la implementación.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del seudocódigo y de la identificación de variables y sus tipos.	5 minutos
	Individualmente, codificar el algoritmo.	10 minutos
	Presentar, argumentar y defender la codificación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor codificación del programa.	8 minutos
	Escribir y probar el programa en el computador.	12 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar algoritmos de solución para problemas que requieran el uso de estructuras secuenciales, selectivas simples y dobles Declarar y utilizar correctamente los tipos de datos simples, Utilizar correctamente la jerarquía de operadores, Utilizar correctamente las estructuras de programación, Construir e implementar programas a partir de su representación algorítmica.	

Tabla 27: Quinta Semana, 3º período

La actividad anterior se complementa con la que se muestra en la tabla 28 (página siguiente), entregada para desarrollar por los grupos fuera del horario de clases.

Sexta Semana:

A la complejidad de los algoritmos selectivos tanto simples como dobles, en el primer período de esta semana, se agrega la anidación de este tipo de estructuras (tabla 29 en la página siguiente).

Para estos efectos, en una primera instancia de tratará con la implementación de los algoritmos que contienen estructuras selectivas anidadas que fueron desarrolladas como ejemplo en clases, para luego en una segunda instancia trabajar en la resolución de problemas nuevos que requieran del uso de estas estructuras algorítmicas y de programación.

Complementariamente se debe insistir en el correcto manejo de los aspectos sintácticos y semánticos para estas estructuras más complejas, como también en la

definición de los casos de prueba que permitan garantizar el correcto funcionamiento de las soluciones ante las mayores complejidades.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Algoritmos y programas. Análisis del problema. Diseño de algoritmos de solución. Representación algorítmica de la solución.		
Descripción de la tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas secuenciales y selectivas simples y dobles.	Cantidad de problemas planteados:	10
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	3 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	5 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Elaborar la especificación del problema y el algoritmo de solución,	10 minutos	
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar los formalismos de la etapa de análisis. Realizar el proceso de análisis de problemas. Aplicar los formalismos de la etapa de diseño de soluciones. Diseñar soluciones a problemas que requieren de la implementación de algoritmos selectivos simples y dobles. Comprender los formalismos de las representaciones algorítmicas y su lenguaje. Interpretar y usar los lenguajes algorítmicos propios de las formas de representación de soluciones.		

Tabla 28: Quinta Semana, actividad complementaria

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Implementación y prueba de programas

Tabla 29: Sexta Semana, 1ª período

Para el caso del trabajo con problemas nuevos, se debe prestar especial atención a los errores de las soluciones, ya que en virtud de la complejidad de los mismos, los errores podrían producirse en el desarrollo de cualquiera de la etapas de la metodología, situación que se debe aprovechar para enfatizar que los errores no sólo se producen en la codificación y prueba de los programas, sino que, en una o más de las etapas.

En esta actividad cooperativa se trabajan las mismas competencias de la actividad cooperativa del tercer período de la quinta semana, pero agregando la competencia referida a la capacidad para tratar con la programación de anidaciones de estructuras selectivas simples y dobles, y con el diseño y uso de casos de prueba para este tipos de estructuras.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Implementación y prueba de programas.	
Tipo de tarea:	Implementar programas a partir de su representación algorítmica, y definir y usar casos de prueba a partir de la especificación del problema.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar la representación algorítmica a partir de su seudocódigo, e identificar las variables y tipos que deben considerarse para la implementación.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del seudocódigo y de la identificación de variables y sus tipos.	5 minutos
	Individualmente, codificar el algoritmo.	10 minutos
	Presentar, argumentar y defender la codificación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor codificación del programa.	8 minutos
	Escribir y probar el programa en el computador.	12 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar algoritmos de solución para problemas que requieran el uso de estructuras secuenciales, selectivas simples, dobles y anidaciones de estas estructuras. Declarar y utilizar correctamente los tipos de datos simples, Utilizar correctamente la jerarquía de operadores, Utilizar correctamente las estructuras de programación, Construir e implementar programas a partir de su representación algorítmica.	

Tabla 30: Sexta Semana, 2ª período

La actividad de la tabla 30 se complementa con la de la tabla 31 (en la siguiente página), entregada para desarrollar por los grupos fuera del horario de clases.

En el tercer período de esta semana, se realiza una segunda sesión de actividad de cooperativa (tabla 32 en la página), en la idea de profundizar el desarrollo de las capacidades trabajadas en la anterior. Para estos efectos los grupos trabajaran en la resolución de problemas de mayor complejidad.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Implementación y prueba de programas.		
Descripción de la tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas selectivas simples, dobles y anidaciones de estas estructuras.	Cantidad de problemas planteados:	10
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	3 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	4 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Elaborar la especificación del problema y el algoritmo de solución,	10 minutos	
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar algoritmos de solución para problemas que requieran el uso de estructuras secuenciales, selectivas simples, dobles y anidaciones de estas estructuras. Declarar y utilizar correctamente los tipos de datos simples, Utilizar correctamente la jerarquía de operadores, Utilizar correctamente las estructuras de programación, Construir e implementar programas a partir de su representación algorítmica.		

Tabla 31: Sexta Semana, 2º período, actividad complementaria

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Implementación y prueba de programas.	
Tipo de tarea:	Implementar programas a partir de su representación algorítmica, y definir y usar casos de prueba a partir de la especificación del problema.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar la representación algorítmica a partir de su seudocódigo, e identificar las variables y tipos que deben considerarse para la implementación.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del seudocódigo y de la identificación de variables y sus tipos.	5 minutos
	Individualmente, codificar el algoritmo.	10 minutos
	Presentar, argumentar y defender la codificación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor codificación del programa.	8 minutos
	Escribir y probar el programa en el computador.	12 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar algoritmos de solución para problemas que requieran el uso de estructuras secuenciales, selectivas simples, dobles y anidaciones de estas estructuras. Declarar y utilizar correctamente los tipos de datos simples, Utilizar correctamente la jerarquía de operadores, Utilizar correctamente las estructuras de programación, Construir e implementar programas a partir de su representación algorítmica.	

Tabla 32: Sexta Semana, 3º período

Esta actividad de complementa con la que se muestra en la tabla siguiente, a realizar por los estudiantes fuera del período de clase.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Implementación y prueba de programas.		
Descripción de la tarea:	Elaborar la especificación y el algoritmo de solución para problemas que requieren del uso de estructuras algorítmicas selectivas simples, dobles y anidaciones de estas estructuras.	Cantidad de problemas planteados:	10
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	3 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	4 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Elaborar la especificación del problema y el algoritmo de solución,	10 minutos	
	Presentar, argumentar y defender la especificación y el algoritmo elaborado.	6 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar algoritmos de solución para problemas que requieran el uso de estructuras secuenciales, selectivas simples, dobles y anidaciones de estas estructuras. Declarar y utilizar correctamente los tipos de datos simples, Utilizar correctamente la jerarquía de operadores, Utilizar correctamente las estructuras de programación, Construir e implementar programas a partir de su representación algorítmica.		

Tabla 33: Sexta Semana, 3º período, actividad complementaria

Séptima Semana:

Al tratar con las estructuras de tipo repetitivas, conocida también como ciclos o bucles, se debe tener presente que el manejo de ellas, constituye una dificultad importante para la generalidad de los estudiantes, y la experiencia muestra que existen dos dificultades importantes, la primera referida al diseño de las soluciones algorítmicas para problemas que requieren el uso de repeticiones, y la segunda referida a la comprensión de cómo funciona un código de programa que tiene repeticiones. Por tanto estos son dos aspectos a tener en consideración al tratar con estos contenidos.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Estructura algorítmicas repetitivas.

Tabla 34: Séptima Semana, 1º período

Para iniciar el tratamiento de este tipo de estructuras, primero se trabajará en el desarrollo de la capacidad de reconocer que parte o partes de un procesamiento requieren ser repetitivos, para los cual se ejemplificará usando problemas

matemáticos simples, como la construcción de una tabla de multiplicar, la determinación de los divisores de un número, y otros problemas similares.

En este período se trabajará sobre el diseño algorítmico para problemas que consideren procesos repetitivo controlados por contador, para los cuales una vez diseñado el algoritmo, se trabajará con las técnicas de ruteo del algoritmo clarificando cómo éste se ejecuta para un cierto conjunto de valores. Así también se trabajará en las formas de implementación como programa, de este tipo de estructuras.

Los contenidos a tratar son:

- El concepto de procesamiento repetitivo.
- Representación algorítmica de estructuras repetitivas.
- Técnicas de ruteo de estructuras algorítmicas repetitivas.
- Implementación en C de estructuras repetitivas controladas por contador.
- Aspectos del diseño de casos de pruebas para estructuras repetitivas.

Los objetivos de aprendizaje en términos de capacidades a lograr por los estudiantes son:

- Describir las características de un proceso repetitivo.
- Desarrollar los procesos de análisis y diseño a problemas que requieren de un proceso repetitivo controlado por un contador.
- Elaborar algoritmos en diagramas de flujo y seudocódigo para resolver problemas que requieran de una estructura repetitiva controlada por contador.
- Implementar en lenguaje C, programas que contengan estructuras repetitivas controladas por contador.
- Diseño de casos de prueba para estructuras repetitivas.

Por tanto las competencias a trabajar con los estudiantes son:

- Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo.
- Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador.
- Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador.
- Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador.
- Implementar programas que contengan estructuras repetitivas controladas por contador.
- Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador.

En esta actividad cooperativa (tabla 35), interesa fundamentalmente el desarrollo de las capacidades para tratar con el diseño de soluciones a problemas relacionados

con el manejo de estructuras repetitivas controladas por contador, ámbito en el cual se pretenden el diseño de las soluciones algorítmicas. Junto con ello se trabajará en la implementación como programa de este tipo de estructuras, y en el diseño de casos de prueba para verificar la correctitud de las soluciones.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	El concepto de procesamiento repetitivo. Representación algorítmica de estructuras repetitivas. Técnicas de ruteo de estructuras algorítmicas repetitivas. Implementación en C de estructuras repetitivas controladas por contador. Aspectos del diseño de casos de pruebas para estructuras repetitivas.	
Tipo de tarea:	Diseñar algoritmos para problemas que requieran el uso de estructuras repetitivas controladas por contador, e implementar y probar el programa correspondiente.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema.	5 minutos
	Individualmente, diseñar el algoritmo, implementar el programa, definir los casos de prueba y probar el programa.	25 minutos
	Presentar, argumentar y defender la solución.	10 minutos
	Discutir y argumentar elementos para establecer la mejor solución.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador. Implementar programas que contengan estructuras repetitivas controladas por contador. Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador.	

Tabla 35: Séptima Semana, 2º período

Esta actividad cooperativa desarrollada en clases, se complementa con la se muestra en la tabla 36 en la página siguiente, la que deberán realizar los grupos cooperativos en horario extra clase.

En el tercer período de esta semana (tabla 37, en la página siguiente), se trabajará sobre el diseño algorítmico para problemas que consideren más de un proceso repetitivo controlado por contador, incluyendo también anidaciones de repeticiones. Además, se continuará con la explicitación del ruteo de los algoritmos para mostrar cómo se ejecutan, en la idea de que los estudiantes profundicen la comprensión respecto de la forma en que operan las repeticiones cuando se organizan en estructuras más complejas.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase
Duración estimada:	360 minutos
Aspectos a trabajar:	El concepto de procesamiento repetitivo.

	Representación algorítmica de estructuras repetitivas. Técnicas de ruteo de estructuras algorítmicas repetitivas. Implementación en C de estructuras repetitivas controladas por contador. Aspectos del diseño de casos de pruebas para estructuras repetitivas.		
Descripción de la tarea:	Aplicar las etapas del proceso de resolución de problemas para problemas que requieren del uso de estructuras algorítmicas repetitivas controladas por contador.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	6 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Diseñar el algoritmo de solución, implementar el programa y los casos de prueba.	20 minutos	
	Presentar, argumentar y defender la implementación de la solución.	10 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	7 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador. Implementar programas que contengan estructuras repetitivas controladas por contador. Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador.		

Tabla 36: Séptima Semana, 2º período, actividad complementaria

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Estructura algorítmicas repetitivas anidadas.

Tabla 37: Séptima Semana, 3º período

Por tanto el objetivo de aprendizaje que se persigue está referido al desarrollo de la capacidad de comprender y diseñar algoritmos y programas que poseen varios procesos repetitivos controlados por contador, y que consideren también anidaciones de estas estructuras repetitivas.

Octava Semana:

En el primer período de esta semana se desarrolla la actividad cooperativa que se indica en la tabla 38 de la página siguiente. En ella, los estudiantes trabajaran en la resolución de problemas que impliquen el uso de anidaciones de estructuras repetitivas controladas por contador. Para estos efectos se propondrán tres situaciones problemas que deben ser resueltas.

En estos términos, se agrega la competencia referida el diseño de algoritmos, la implementación de programas, y la definición y uso de casos de prueba para anidaciones de estructuras repetitivas controladas por contador.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Estructura algorítmicas repetitivas anidadas. Implementación y prueba de programas.	
Tipo de tarea:	Diseñar algoritmos para problemas que requieran el uso de estructuras repetitivas anidadas controladas por contador, e implementar y probar el programa correspondiente.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema.	5 minutos
	Individualmente, diseñar el algoritmo, implementar el programa y definir y usar los casos de prueba.	25 minutos
	Presentar, argumentar y defender la solución desarrollada.	10 minutos
	Discutir y argumentar elementos para establecer la mejor solución.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador. Implementar programas que contengan estructuras repetitivas controladas por contador. Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador.	

Tabla 38: Octava Semana, 1º período

La actividad anterior se complementa con la que se muestra en la tabla 39 de la página siguiente, la que deberan realizar los grupos en horario extra fuera de las sesiones de clases.

En el segundo período de esta semana (tabla 40 en la página siguiente), tratamos con la resolución de problemas que demanden el uso de estructuras repetitivas controladas por tarea y por centinela, incluyendo problemas cuyas soluciones contengan anidaciones de las mismas.

Conforme a esta nueva complejidad de las estructuras repetitivas, se debe trabajar de manera intensiva en la explicitación de cómo identificar si un cierto problema requiere de repeticiones controladas por tarea o centinela.

Así también, se continúa con las explicitaciones de los ruteos de los algoritmos y los programas, ello con el propósito de que los estudiantes comprendan cómo funcionan estas nuevas estructuras, y las combinaciones de las mismas.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase
Duración estimada:	360 minutos
Aspectos a trabajar:	Metodología de programación. Estructura algorítmicas repetitivas anidadas controladas por contador.

	Implementación y prueba de programas.		
Descripción de la tarea:	Aplicar las etapas del proceso de resolución para problemas que requieren del uso de estructuras algorítmicas repetitivas anidadas controladas por contador.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	6 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Diseñar el algoritmo de solución, e implementar el programa y los casos de prueba.	20 minutos	
	Presentar, argumentar y defender el diseño de la solución, la implementación del programa y los casos de prueba.	10 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	7 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador. Implementar programas que contengan estructuras repetitivas controladas por contador. Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador.		

Tabla 39: Octava Semana, 1º período, actividad complementaria

El contenido a tratar entonces, corresponde a repeticiones controladas por tarea y por centinela, y el objetivo de aprendizaje apunta al desarrollo de la capacidad para identificar el tipo de estructuras repetitivas más apropiadas para resolver un cierto problema, y para desarrollar la capacidad de comprender como operan algorítmicamente estas nuevas estructuras.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Estructuras repetitivas controladas por centinela y por tarea. Implementación y prueba de programas.

Tabla 40: Octava Semana, 2º período

Nuevamente, es importante la explicitación de la definición de los casos de prueba adecuados para verificar la correctitud de las implementaciones, especialmente aquellos que permiten probar las condiciones de borde de cada problema.

El objetivo de aprendizaje que se debe perseguir, es que los estudiantes desarrollen y perfeccionen la capacidad para diseñar algoritmos e implementar programas que utilicen este tipo de estructuras repetitivas, y junto con ello sean capaces de determinar el tipo de implementación que es más apropiada para cada algoritmo repetitivo.

Además de lo anterior, se pretenden que los estudiantes aprendan como diseñar y usar casos de prueba para este tipo de estructuras.

Tipo de actividad:	Trabajo cooperativo

Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Estructuras repetitivas controladas por centinela y por tarea. Implementación y prueba de programas.	
Tipo de tarea:	Diseñar algoritmos, implementar y probar programa para problemas que requieran el uso de estructuras repetitivas controladas por centinela y por tarea.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema.	5 minutos
	Individualmente, diseñar el algoritmo, implementar el programa y probarlo.	20 minutos
	Presentar, argumentar y defender la solución desarrollada.	10 minutos
	Discutir y argumentar elementos para establecer la mejor solución.	10 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador. Implementar programas que contengan estructuras repetitivas controladas por contador. Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador.	

Tabla 41: Octava Semana, 3º período

El propósito de esta actividad cooperativa (tabla 41), es desarrollar en los estudiantes la capacidad para tratar con la resolución de problemas que demanden el uso de estructuras repetitivas controladas por centinela y por tarea.

En esta perspectiva se pretende también que desarrollen la capacidad para realizar las implementaciones en el lenguaje de estructuras de este tipo, y que sean capaces de diseñar y usar casos de prueba apropiados para verificar la correctitud de sus diseños algorítmicos e implementaciones.

Esta actividad cooperativa hecha en clases se complementa con la que se muestra en la tabla 42 (en la página siguiente), que deberá ser realizada en tiempo extra fuera de la clase.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Estructuras repetitivas controladas por centinela y por tarea. Implementación y prueba de programas.		
Descripción de la tarea:	Aplicar las etapas del proceso de resolución para problemas que requieren del uso de estructuras algorítmicas repetitivas controladas por centinela y por tarea.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	6 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Diseñar el algoritmo de solución, e implementar el programa y los casos de prueba.	20 minutos	
	Presentar, argumentar y defender el diseño de la solución, la implementación del programa y los casos de prueba.	10 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	7 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por tarea y centinela. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por tarea y centinela. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por tarea y centinela. Implementar programas que contengan estructuras repetitivas controladas por tarea y centinela. Diseñar y usar casos de prueba para estructuras repetitivas controladas por tarea y centinela.		

Tabla 42: Octava Semana, 3º período, actividad complementaria

Novena Semana:

Teniendo en consideración que la comprensión y el manejo de las estructuras de tipo repetitivas, es reconocida como una de las grandes dificultades que tienen los estudiantes en un curso de programación, en esta sesión se desarrolla una recapitulación integrada del tratamiento de este tipo de estructuras (tabla 43).

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Estructuras repetitivas controladas por contador, centinela y por tarea. Implementación y prueba de programas

Tabla 43: Novena Semana, 1º período

Para estos efectos, se trabajara sobre la explicitación del proceso de resolución de problemas que demanden la integración de las tres formas de repetición (contador, centinela y tarea) y de anidaciones de las mismas.

En esta explicitación se debe poner especial énfasis en la consideración de los elementos particulares de cada problema, mediante los cuales se llega a establecer el tipo de repetición que es aplicable a cada uno.

Así también, es de importancia clarificar la elección del tipo de implementación que es adecuada a cada caso, analizando para ello, las ventajas y desventajas para cada caso en particular, y de paso reiterar la explicitación del diseño y uso de los casos de prueba para evaluar la validez de cada solución.

La siguiente actividad cooperativa (tabla 44), se corresponde con la recapitulación hecha en el período anterior, y por tanto tiene como propósito que los grupos cooperativos traten con los aspectos prácticos referidos al uso y aplicación de las tres estructuras repetitivas.

Como tarea a desarrollar por los grupos cooperativos, se plantearan cinco problemas, cuyas soluciones demandarán el uso de alguno de los tres tipos de repeticiones y combinaciones de ellos.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Estructuras repetitivas controladas por contador, centinela y por tarea. Implementación y prueba de programas	
Tipo de tarea:	Diseñar algoritmos, implementar y probar programa para problemas que requieran el uso de estructuras repetitivas controladas por contador, centinela y tarea.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema.	5 minutos
	Individualmente, diseñar el algoritmo, implementar el programa y probarlo.	20 minutos
	Presentar, argumentar y defender la solución desarrollada.	10 minutos
	Discutir y argumentar elementos para establecer la mejor solución.	10 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador, centinela y tarea. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador, centinela y tarea. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador, centinela y tarea. Implementar programas que contengan estructuras repetitivas controladas por contador, centinela y tarea. Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador, centinela y tarea.	

Tabla 44: Novena Semana, 2º período

Esta actividad se complementa con la se realizará fuera del tiempo de clases, y que se muestra en la tabla 45, en la página siguiente.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Estructuras repetitivas controladas por contador, centinela y por tarea. Implementación y prueba de programas		
Descripción de la tarea:	Aplicar las etapas del proceso de resolución para problemas que requieren del uso de estructuras algorítmicas repetitivas controladas por contador, centinela y por tarea.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema,	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema,	7 minutos	
	Identificar los conocimientos implicados en el problema,	6 minutos	
	Negociar y establecer consenso respecto del significado de los conocimientos implicados,	5 minutos	
	Diseñar el algoritmo de solución, e implementar el programa y los casos de prueba.	20 minutos	
	Presentar, argumentar y defender el diseño de la solución, la implementación del programa y los casos de prueba.	10 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	7 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar el proceso de análisis a problemas que requieran del uso de un proceso repetitivo. Aplicar el proceso de diseño para problemas que consideren el uso de un proceso repetitivo controlado por contador, centinela y tarea. Construir representaciones algorítmicas para problemas que contengan un proceso repetitivo controlado por contador, centinela y tarea. Realizar la traza de algoritmos que contengan un proceso repetitivo controlado por contador, centinela y tarea. Implementar programas que contengan estructuras repetitivas controladas por contador, centinela y tarea. Diseñar y usar casos de prueba para estructuras repetitivas controladas por contador, centinela y tarea.		

Tabla 45: Novena Semana, 2º período, actividad complementaria

En la siguiente actividad (tabla 46), se realiza la segunda evaluación acumulativa. En ella se deben evaluar las siguientes competencias:

- Aplicar las cuatro etapas de la metodología a problemas que requieran el uso de estructuras algorítmicas y de programación de tipo repetitivas.
- Determinar el tipo de estructura repetitiva que es apropiada para resolver un determinado problema.
- Utilizar correctamente las estructuras algorítmicas y de programación repetitivas.
- Definir y usar casos de pruebas para evaluar la correctitud de programas que usan estructuras repetitivas.

Tipo de actividad:	Segunda Evaluación Acumulativa
Duración:	90 minutos
Contenidos a evaluar:	Metodología de programación. Estructuras repetitivas. Implementación y prueba de programas

Tabla 46: Novena Semana, 3º período

Décima Semana:

Tipo de actividad:	Análisis de resultados de la segunda evaluación.
Duración:	45 minutos

Tabla 47: Décima Semana, 1º período, primer momento

Con el propósito de sacar el mejor provecho posible de esta instancia de revisión y análisis de resultados (tabla 47), previamente se realizará una categorización de las deficiencias en el manejo de los conocimientos y competencias que presentaron los estudiantes.

En esta categorización se tratarán de encontrar además relaciones entre las deficiencias de distintas categorías, con el propósito de establecer si unas deficiencias inciden sobre otras.

Así también se hace conveniente establecer cuales deficiencias fueron las mayormente observadas, con el objeto de establecer cuáles de ellas, representan errores que se cometieron con más frecuencia.

Usando esta información, en el desarrollo de la sesión de revisión y análisis de los resultados, el profesor presentará el desarrollo de la resolución de cada problema de la prueba, e irá haciendo mención a las dificultades detectadas, y entregando retroalimentación con miras a superar las deficiencias apreciadas, dando espacio además, para la intervención de los estudiantes ya sea para realizar consultas, precisiones o comentarios.

Tipo de actividad:	Clase expositiva
Duración:	45 minutos
Contenidos a tratar:	Metodología de programación. Concepto de función. Programación modular.

Tabla 48: Décima Semana, 1º período, segundo momento

En la segunda parte de este período (tabla 48), se inicia el tratamiento del concepto de función. Para estos efectos, se partirá presentado las funciones matemáticas básicas que tiene el lenguaje, se clarificará el concepto de librería de funciones, para luego presentar y desarrollar aplicaciones simples, usando las funciones matemáticas y las de manejo de string.

También es pertinente clarificar que el lenguaje C, es por naturaleza un lenguaje funcional, que posee muy pocas instrucciones propias del lenguaje, y que gran parte de lo que se escribe en un programa no son instrucciones sino que funciones, como por ejemplo las funciones para de entrada y salida.

Con esto se deben pretender los siguientes objetivos de aprendizaje:

- Comprender el concepto de función,
- Reconocer las características de una función en programación,

- Comprender el concepto de librería de funciones,
- Comprender la utilidad de las funciones como estrategia de programación.
- Integrar funciones matemáticas y de manejo de string dentro de programas pequeños.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Concepto de función. Programación modular.

Tabla 49: Décima Semana, 2º período

En el segundo período de esta semana (tabla 49), se continuará profundizando el concepto de función, explicitando la utilidad de las funciones como estrategia de programación, y el concepto de programación modular.

Luego de ello se deben conceptualizar los tipos de función, en términos de si reciben parámetros o no, las formas de pasar parámetros a las funciones (por valor, por referencia), y de si devuelven valores o no.

Seguido de lo anterior se debe presentar y explicitar la estructura general de una función en el lenguaje C y el concepto de variables locales y globales, para a continuación explicitar la forma de invocar las funciones desde el programa principal, y las formas de tratar los valores que devuelve la función en el programa principal. Para esto se usarán pequeños ejemplos de implementación que contengan tanto el programa principal como funciones.

Para la implementación de las funciones se usarán ejemplos de algoritmos vistos anteriormente, como los algoritmos selectivos simples y dobles, y los algoritmos de repeticiones.

Los aprendizajes que se pretenden lograr con los estudiantes son:

- Comprender la utilidad de las funciones como estrategia de programación,
- Caracterizar los distintos tipos de función en relación con los tipos de parámetros que recibe y devuelve,
- Identificar la estructura de una función,
- Escribir, implementar e integrar funciones en un programa
- La competencia a desarrollar esta referida a la capacidad para diseñar e implementar pequeños programas en los que se integre el uso de funciones.

En el tercer período de esta semana, tabla 50 de la siguiente página, se realiza una actividad cooperativa, con la que se persigue el propósito de que los estudiantes tengan un primer acercamiento al tratamiento de las funciones, para lo cual trabajaran en los aspectos prácticos del uso y manejo de las librerías de funciones básicas que tiene el lenguaje.

Para ello se les presentarán cinco problemas que se relacionan con el uso de la librería de funciones matemáticas (math.h), y de manejo de cadenas de caracteres (string.h).

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Concepto de función. Programación modular	
Tipo de tarea:	Implementar y probar programas que utilicen funciones matemáticas y de manejo de cadenas de caracteres.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y establecer las funciones que deben usarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y las funciones a utilizar.	5 minutos
	Individualmente, implementar y probar el programa.	25 minutos
	Presentar, argumentar y defender la implementación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Reconocer la operatividad de las funciones matemáticas y de manejo de cadenas de caracteres más comunes. Reconocer las formas de traspaso de parámetros entre el programa principal y una función. Reconocer las formas de comunicación entre una función y el programa principal. Comprender los conceptos de variables globales y locales, y usarlos correctamente. Usar funciones de la librería matemática y de manejo de cadenas de caracteres, integrándolas en la implementación de programas.	

Tabla 50: Décima Semana, 3º período

La actividad cooperativa complementaria a esta que debe ser realizada en horario extra clase es la siguiente se muestra en la tabla 51 en la página siguiente.

Décimo primera Semana:

En el primer período de esta semana (tabla 52 en la página siguientes), tratamos con los aspectos referidos a la implementación de funciones. Para ello, se trabajará sobre la implementación como función de varios de los programas de ejemplo desarrollados tanto en las clases anteriores como en las actividades cooperativas, como es el caso de los problemas referidos al manejo de estructuras selectivas y repetitivas. Junto con la implementación de las funciones se explicitarán los aspectos referidos a la implementación del programa principal que haga uso de las funciones.

Lo fundamental es clarificar como se implementan estas funciones y el programa principal que las usa, reiterando además los elementos referidos al manejo del traspaso de parámetros y de las variables globales y locales.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase

Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Concepto de función. Programación modular		
Descripción de la tarea:	Implementar y probar programas que utilicen funciones matemáticas y de manejo de cadenas básicas.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema e identificación de las funciones que se requieren para implementar el programa.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la identificación de las funciones.	7 minutos	
	Implementar y probar el programa de solución	20 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Reconocer la operatividad de las funciones matemáticas y de manejo de cadenas de caracteres más comunes. Reconocer las formas de traspaso de parámetros entre el programa principal y una función. Reconocer las formas de comunicación entre una función y el programa principal. Comprender los conceptos de variables globales y locales, y usarlos correctamente. Usar funciones de la librería matemática y de manejo de cadenas de caracteres, integrándolas en la implementación de programas.		

Tabla 51: Décima Semana, 3º período, actividad complementaria

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Metodología de programación. Concepto de función. Programación modular. Implementación de funciones. Librería de funciones

Tabla 52: Décimo primera Semana, 1º período

En este primer período, se explicitará también la forma de crear una pequeña librería de funciones y el uso de ella en un programa principal.

En estos términos, los contenidos que se tratan son:

- Tipos de variables en una función y su ámbito,
- Aspectos del traspaso de parámetros por valor y por referencia,
- Aspectos de la implementación de funciones,
- Aspectos de la llamada de una función,
- El concepto de librería de funciones,
- Aspectos de la creación y uso de bibliotecas de funciones.

A continuación, en el segundo período de esta semana, se realiza la actividad cooperativa que se muestra en la tabla 53 de la página siguiente

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Concepto de función. Programación modular. Implementación de funciones. Librería de funciones	
Tipo de tarea:	Implementar funciones, el programa que las use y probarlo.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y caracterizar las funciones que deben implementarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y la caracterización de las funciones.	5 minutos
	Individualmente, implementar las funciones, el programa y probarlo.	25 minutos
	Presentar, argumentar y defender la implementación de las funciones y el programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar soluciones utilizando la programación modular basada en funciones. Realizar implementaciones en lenguaje C de programas modular que usan funciones de las librerías estándar, además de otras funciones creadas para el efecto.	

Tabla 53: Décimo primera Semana, 2º período

Esta actividad se complementa con la siguiente (tabla 54), a realizar en horario extra clase.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Concepto de función. Programación modular. Implementación de funciones. Librería de funciones		
Descripción de la tarea:	A partir del enunciado de problemas, implementar las funciones necesarias, el programa que las use y probarlo.	Cantidad de problemas planteados:	6
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las funciones que deben implementarse.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y la caracterización de las funciones.	7 minutos	
	Implementar y probar el programa de solución	35 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar soluciones utilizando la programación modular basada en funciones. Realizar implementaciones en lenguaje C de programas modular que usan funciones de las librerías estándar, además de otras funciones creadas para el efecto.		

Tabla 54: Décimo primera Semana, 2º período, actividad complementaria

Considerando que el aprendizaje y el desarrollo de las capacidades referidas a la implementación y el uso de funciones, es otra de las dificultades importantes que presentan los estudiantes en un curso de programación, en esta semana se realizará una segunda sesión de aprendizaje cooperativo respecto este tema (tabla 55), para lo cual se propondrá a los estudiantes problemas de mayor complejidad que la sesión anterior.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Metodología de programación. Concepto de función. Programación modular. Implementación de funciones. Librería de funciones	
Tipo de tarea:	Implementar funciones, el programa que las use y probarlo.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y caracterizar las funciones que deben implementarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y la caracterización de las funciones.	5 minutos
	Individualmente, implementar las funciones, el programa y probarlo.	25 minutos
	Presentar, argumentar y defender la implementación de las funciones y el programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar soluciones utilizando la programación modular basada en funciones. Realizar implementaciones en lenguaje C de programas modular que usan funciones de las librerías estándar, además de otras funciones creadas para el efecto.	

Tabla 55: Décimo primera Semana, 3º período

La actividad anterior se complementa con la que se muestra en la tabla 56 (página siguiente), que se realiza en horario extra clase.

Décimo segunda Semana:

En el primer período de esta semana (tabla 57 en página siguiente), se inicia el tratamiento de los contenidos referidos a las estructuras de datos, para lo cual se deberá primero explicitar el concepto de estructura de datos, lo que se hará partiendo por la estructura de datos más simple, que son los vectores o arreglos unidimensionales.

Luego se explicitarán los beneficios y facilidades que se tienen con el uso de este tipo de estructuras de datos, como por ejemplo para las operaciones de búsqueda y ordenamiento de datos.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Metodología de programación. Concepto de función. Programación modular. Implementación de funciones. Librería de funciones		
Descripción de la tarea:	A partir del enunciado de problemas, implementar las funciones necesarias, el programa que las use y probarlo.	Cantidad de problemas planteados:	6
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las funciones que deben implementarse.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y la caracterización de las funciones.	7 minutos	
	Implementar y probar el programa de solución	35 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Aplicar la metodología de resolución de problemas de procesamiento de datos, Diseñar soluciones utilizando la programación modular basada en funciones. Realizar implementaciones en lenguaje C de programas modular que usan funciones de las librerías estándar, además de otras funciones creadas para el efecto.		

Tabla 56: Décimo primera Semana, 3º período, actividad complementaria

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Concepto de estructura de datos, La estructura de datos de tipo arreglo, Operaciones básicas sobre estructuras de tipo arreglos.

Tabla 57: Décimo segunda Semana, 1º período

A continuación se explicaran las operaciones básicas que se realizan sobre este tipo de estructuras, para finalizar explicitando la forma de definir y declarar variables que se corresponden con esta estructura.

Los objetivos de aprendizaje que deben alcanzar los estudiantes son:

- Describir el concepto de estructura de datos,
- Establecer las características de las estructuras de datos del tipo vector, y
- Describir las operaciones básicas sobre estructuras de datos del tipo vector.

En el segundo período de esta semana (tabla 58 en página siguiente), se presentan y explicitan los algoritmos para realizar las operaciones básicas sobre vectores, partiendo por las operaciones de llenado de un vector, y mostrar los elementos de un vector, para luego tratar con otras operaciones como frecuencia de un cierto elemento, y para el caso de vectores que tienen elementos numéricos, la operación para obtener el promedio de los elementos, el valor máximo y el valor mínimo.

Luego de esto, se presentan y explicitan las implementaciones de estas operaciones en el lenguaje C, para lo cual se trabajará usando los principios de la programación modular, ya tratados.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Diseño algoritmos para realizar operaciones básicas sobre vectores, Implementación en C de algoritmos para realizar operaciones básicas sobre vectores.

Tabla 58: Décimo segunda Semana, 2º período

Para estos efectos se presentará y explicitará la implementación de cada operación como una función, para luego presentar y explicitar la implementación del programa principal que utilice las funciones.

En este punto es importante recalcar que el paso de vectores a las funciones se hace por referencia y no por valor. Para clarificar este aspecto se mostraran algunos ejemplos de traspaso de vectores a funciones mediante punteros explícitos.

Con la presentación y explicitación de estos elementos, se pretenden que los estudiantes logren los siguientes objetivos de aprendizaje:

- Comprender los algoritmos para ejecutar las operaciones básicas de llenar y mostrar el contenido de un vector, encontrar frecuencia de un elemento, promedio de los elementos, y elemento mayor y menor contenido en el vector,
- Implementar como funciones los algoritmos para las operaciones básicas,
- Implementar el programa principal que integre el uso de las funciones para las operaciones básicas.

En el tercer período de esta semana (tabla 59), se presentan y explicitan los algoritmos realizar ordenamientos sobre los elementos de un vector. Se tratarán los algoritmos de ordenamiento de burbuja, de inserción y selección, y de mezcla.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Algoritmos de ordenamiento sobre vectores, Implementación en C de algoritmos de ordenamiento sobre vectores.

Tabla 59: Décimo segunda Semana, 3º período

Se debe mostrar y explicitar la valoración de la eficiencia de cada algoritmo, para lo cual se usará la técnica del ruteo de cada algoritmo para un mismo conjunto de datos.

Se debe mostrar y explicitar las implementaciones de estos algoritmos como funciones, tanto para ordenamientos de elementos numéricos, como para caracteres y cadenas de caracteres.

Con esto se pretenden que los estudiantes logren los siguientes objetivos de aprendizaje:

- Comprender los algoritmos de burbuja, de inserción y selección, y de mezcla.
- Implementar como funciones los algoritmos de ordenamiento,
- Implementar el programa principal que integre el uso de las funciones de ordenamiento.

Décimo tercera Semana:

En el primer período de esta semana (tabla 60), se presentan y explicitan los algoritmos realizar búsquedas de elementos en un vector. Se tratarán los algoritmos de búsqueda secuencial y binaria.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Algoritmos de búsqueda sobre vectores, Implementación en C de algoritmos de búsqueda sobre vectores.

Tabla 60: Décimo tercera Semana, 1º período

En este punto es importante explicitar la eficiencia de cada algoritmo, para lo cual se utilizará la técnica de ruteo de los algoritmos para cada caso y para un mismo conjunto de datos.

Se debe mostrar y explicitar las implementaciones de estos algoritmos como funciones, tanto para la búsqueda de elementos numéricos, de caracteres y de cadenas de caracteres.

Con esto se pretenden que los estudiantes logren los siguientes objetivos de aprendizaje:

- Comprender los algoritmos de búsqueda secuencial y binaria.
- Implementar como funciones los algoritmos de búsqueda,
- Implementar el programa principal que integre el uso de las funciones de búsqueda.
- Se indica a los estudiantes que para preparar la actividad cooperativa del próximo período revisen y estudien los siguientes contenidos:
- Concepto de estructura de datos,
- La estructura de datos del tipo vector y sus algoritmos para realizar operaciones básicas,
- Los algoritmos de ordenamiento de burbuja, de inserción y selección, y de mezcla.
- Los algoritmos de búsqueda secuencial y binaria.

En el segundo período de esta semana, se realiza una actividad de trabajo cooperativo (tabla 61 en página siguiente), en la que se proponen problemas para los cuales se requiere la implementación de operaciones sobre estructuras de datos de tipo vector, que contienen elementos numéricos. Estos problemas deben ser resueltos mediante programación modular usando funciones.

2º Período

Tipo de actividad:	Trabajo cooperativo
Duración:	90 minutos
Aspectos a trabajar:	Operaciones sobre vectores.

	Implementación de funciones y programa principal para realizar operaciones sobre vectores que contienen elementos numéricos.	
Tipo de tarea:	Implementar y probar programas que utilicen funciones para realizar operaciones sobre vectores que contienen elementos numéricos.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y establecer las funciones que deben usarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y las funciones a utilizar.	5 minutos
	Individualmente, implementar y probar el programa.	25 minutos
	Presentar, argumentar y defender la implementación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo vectores, que contienen datos de tipo numéricos. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo vectores que tienen elementos de tipo numérico.	

Tabla 61: Décimo tercera Semana, 2º período

La actividad anterior se complementa con la siguiente (tabla 62), para realizar en horario extra clase es:

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Operaciones sobre vectores. Implementación de funciones y programa principal para realizar operaciones sobre vectores que contienen elementos numéricos.		
Descripción de la tarea:	Implementar y probar programas que utilicen funciones para realizar operaciones sobre vectores.	Cantidad de problemas planteados:	6
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las funciones que se requieren.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la caracterización de las funciones.	7 minutos	
	Implementar y probar el programa de solución	35 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo vectores, que contienen datos de tipo numéricos. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo vectores que tienen elementos de tipo numérico.		

Tabla 62: Décimo tercera Semana, 2º período, actividad complementaria

Para el tercer período de esta semana (tabla 63 en página siguiente), se proponen problemas de mayor complejidad que en la actividad anterior, para los cuales se requiere la implementación de varias operaciones sobre estructuras de datos de tipo vector, que pueden contener elementos del tipo caracteres o cadenas de caracteres. Estos problemas deben ser resueltos mediante programación modular usando funciones.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Operaciones sobre vectores. Implementación de funciones y programa principal para realizar operaciones sobre vectores que contienen elementos tipo carácter y/o cadena de caracteres	
Tipo de tarea:	Implementar y probar programas que utilicen funciones matemáticas y de manejo de cadenas de caracteres.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y establecer las funciones que deben usarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y las funciones a utilizar.	5 minutos
	Individualmente, implementar y probar el programa.	25 minutos
	Presentar, argumentar y defender la implementación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo vectores, que contienen datos de tipo carácter y/o cadena de caracteres. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo vectores que tienen datos del tipo carácter y/o cadenas de caracteres	

Tabla 63: Décimo tercera Semana, 3º período

El complemento de la actividad anterior en la tabla 64, para realizar en horario extra clase.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Operaciones sobre vectores. Implementación de funciones y programa principal para realizar operaciones sobre vectores que contienen datos de tipo carácter y/o cadenas de caracteres.		
Descripción de la tarea:	Implementar y probar programas que utilicen funciones para realizar operaciones sobre vectores.	Cantidad de problemas planteados:	6
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las funciones que se requieren.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la caracterización de las funciones.	7 minutos	
	Implementar y probar el programa de solución	35 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo vectores, que contienen datos de tipo caracteres y/o cadenas de caracteres. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo vectores que tienen elementos de tipo carácter y/o cadenas de caracteres.		

Tabla 64: Décimo tercera Semana, 3º período, actividad complementaria

Décimo cuarta Semana:

En el primer período de esta semana (tabla 65), se presentan y explicitan los conceptos referidos a arreglo bidireccional o matriz, y arreglos tridimensionales, y las

operaciones básicas sobre este tipo de estructuras, partiendo por las operaciones de llenado de la estructura de datos, y mostrar sus elementos, para luego tratar con otras operaciones como frecuencia de un cierto elemento, y para el caso de estructuras que tienen elementos numéricos, la operación para obtener el promedio de los elementos, el valor máximo y el valor mínimo.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Arreglos bidireccionales o matrices y tridimensionales, Operaciones básicas sobre matrices y arreglos tridimensionales, Algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Implementación en C, de algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales.

Tabla 65: Décimo cuarta Semana, 1º período

Luego de esto, se presentan y explicitan los algoritmos para realizar estas operaciones.

Se presenta y explicita las implementaciones de las funciones para las operaciones básicas, y se pedirá a los estudiantes que implementen un programa principal en el que se integren estas funciones.

La competencia a desarrollar en los estudiantes es la capacidad para comprender las implementaciones de las operaciones básicas sobre estos tipos de estructuras, y la capacidad para implementar un programa principal que integre estas funciones.

Con esto se pretende que los estudiantes logren los siguientes aprendizajes:

- Comprender las organizaciones de datos del tipo matrices y arreglos tridimensionales,
- Comprender las operaciones básicas sobre estos tipos de estructuras, y
- Comprender los algoritmos para realizar las operaciones básicas para estos tipos de estructuras de datos.

En el segundo período de esta semana (tabla 66 en página siguiente), se realiza una actividad cooperativa, con la que se pretende que los estudiantes desarrollen las capacidades para la implementación de funciones y los correspondientes programas principales, para realizar operaciones sobre estructuras de datos del tipo matrices y arreglos tridimensionales.

Para ello, se presentarán a los estudiantes problemas referidos al manejo de operaciones sobre este tipo de estructuras, para los cuales deberán implementar cada operación como una función, y consecuentemente, deben también implementar el programa principal que haga uso de estas funciones.

Tipo de actividad:	Trabajo cooperativo
Duración:	90 minutos

Aspectos a trabajar:	Operaciones básicas sobre matrices y arreglos tridimensionales, Algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Implementación en C, de algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales.	
Tipo de tarea:	Para problemas que tratan con operaciones básicas sobre matrices y arreglos tridimensionales, implementar las funciones, el programa principal y probarlo.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y caracterizar las operaciones y funciones que se requieren.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y la caracterización de las operaciones y funciones requeridas.	5 minutos
	Individualmente, implementar las funciones, el programa principal y probarlo.	25 minutos
	Presentar, argumentar y defender la implementación de las funciones y el programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Diseñar e implementar funciones para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Diseñar e implementar programas principales que hagan uso de las funciones. Definir y utilizar casos de pruebas apropiados para evaluar la validez de las implementaciones.	

Tabla 66: Décimo cuarta Semana, 2º período

La actividad de este segundo período se complementa con la mostrada en la tabla 67 (página siguiente), que se debe realizar en horario extra clase.

En el tercer período de esta semana (tabla 68 en página siguiente), y teniendo en cuenta que los estudiantes en general tienen dificultades para tratar con operaciones que requieran el manejo de varias estructuras de datos del tipo matrices y arreglos tridimensionales, se realiza una sesión de trabajo cooperativo, en la que se abordaran los mismos aspectos y competencias que en el período anterior, pero se complejizan las operaciones sobre los tipos de estructuras, con implementaciones de funciones y programas principales que las usen, para tratar con operaciones sobre dos más estructuras de este tipo.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Operaciones básicas sobre matrices y arreglos tridimensionales, Algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Implementación en C, de algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales.		
Descripción de la tarea:	Implementar y probar programas que utilicen funciones para realizar operaciones básicas sobre matrices y arreglos tridimensionales.	Cantidad de problemas planteados:	6
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las operaciones y funciones que se requieren.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la caracterización de las operaciones y funciones.	7 minutos	
	Implementar y probar las funciones y el principal de solución	35 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Diseñar e implementar funciones para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Diseñar e implementar programas principales que hagan uso de las funciones. Definir y utilizar casos de pruebas apropiados para evaluar la validez de las implementaciones.		

Tabla 67: Décimo cuarta Semana, 2º período, actividad complementaria

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Operaciones básicas sobre matrices y arreglos tridimensionales, Algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Implementación en C, de algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales.	
Tipo de tarea:	Construir programas de solución y las correspondientes funciones para resolver problemas que tratan con las operaciones básicas sobre matrices y arreglos tridimensionales.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y establecer las funciones que deben usarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y las funciones a utilizar.	5 minutos
	Individualmente, implementar y probar el programa.	25 minutos
	Presentar, argumentar y defender la implementación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Específicas disciplinares; Capacidad para: Diseñar e implementar funciones para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Diseñar e implementar programas principales que hagan uso de las funciones. Definir y utilizar casos de pruebas apropiados para evaluar la validez de las implementaciones.	

Tabla 68: Décimo cuarta Semana, 3º período

La actividad del tercer período, se complementa con la mostrada en la tabla 69, la que debe realizarse en horario extra clase.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Operaciones básicas sobre matrices y arreglos tridimensionales, Algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Implementación en C, de algoritmos para realizar operaciones básicas sobre matrices y arreglos tridimensionales.		
Descripción de la tarea:	Construir programas de solución y las correspondientes funciones para resolver problemas que tratan con las operaciones básicas sobre matrices y arreglos tridimensionales.	Cantidad de problemas planteados:	6
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las operaciones y funciones que se requieren.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la caracterización de las operaciones y funciones.	7 minutos	
	Implementar y probar las funciones y el programa principal de solución	35 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Diseñar e implementar funciones para realizar operaciones básicas sobre matrices y arreglos tridimensionales. Diseñar e implementar programas principales que hagan uso de las funciones. Definir y utilizar casos de pruebas apropiados para evaluar la validez de las implementaciones.		

Tabla 69: Décimo cuarta Semana, 3º período, actividad complementaria

Décimo Quinta Semana:

En el primer período de esta semana (tabla 70), se presenta y explicita el concepto de registro, junto con las operaciones sobre arreglos de registros y los algoritmos para realizarlas. Con esto, se debe explicar las ventajas y facilidades que presentan este tipo de estructuras para la realización de ciertas operaciones sobre estructuras de datos más complejas.

Tipo de actividad:	Clase expositiva
Duración:	90 minutos
Contenidos a tratar:	Concepto de registro. Operaciones básicas sobre estructuras de datos del tipo arreglos de registros. Algoritmos para realizar operaciones básicas sobre arreglos de registros.

Tabla 70: Décimo quinta Semana, 1º período

Con esto se pretende que los estudiantes:

- Comprendan la organización de datos de tipo registro, y los arreglos de registros,
- Describan las operaciones básicas sobre arreglos de registros, y
- Comprendan los algoritmos para realizar operaciones básicas sobre estructuras del tipo arreglos de registros.

- Se indica a los estudiantes que para la actividad cooperativa del siguiente período se requiere que revisen y estudien los siguientes temas:
- La organización de datos del tipo matrices y arreglos tridimensionales, y sus algoritmos para realizar las operaciones básicas.
- La organización de datos de tipo registro, los arreglos de registros, y sus algoritmos para realizar las operaciones básicas.

En el segundo período de esta semana (tabla 71), se trabajará con problemas que tratan con las operaciones sobre estructuras de datos del tipo arreglos de registros, que deben ser resueltos mediante programación modular usando funciones.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Operaciones sobre estructuras de datos del tipo arreglos de registros. Implementación de funciones y programa principal para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.	
Tipo de tarea:	Implementar y probar programas que utilicen funciones matemáticas y de manejo de cadenas de caracteres.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y establecer las funciones que deben usarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y las funciones a utilizar.	5 minutos
	Individualmente, implementar y probar el programa.	25 minutos
	Presentar, argumentar y defender la implementación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo arreglos de registros. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.	

Tabla 71: Décimo quinta Semana, 2º período

Esta actividad cooperativa del segundo período, se complementa con la que se muestra en la tabla 72 (en página siguiente), la que deben realizar los estudiantes fuera del horario de clases.

El tercer período de esta semana (tabla 73 en página siguiente), es la continuación de la actividad de trabajo cooperativo anterior, donde se pedirá a los grupos cooperativos que resuelvan problemas de complejidad mayor que en la actividad previa, la que además se complementa con la siguiente actividad a realizar por los estudiantes en horario extra clase.

La actividad de trabajo cooperativo del tercer período de esta semana se complementa con la actividad que se muestra en la tabla 74 (en la página siguiente).

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Operaciones sobre estructuras de datos del tipo arreglos de registros. Implementación de funciones y programa principal para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.		
Descripción de la tarea:	Implementar funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros, e implementar y probar el programa principal.	Cantidad de problemas planteados:	6
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las funciones que se requieren.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la caracterización de las funciones.	7 minutos	
	Implementar y probar el programa de solución	35 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	6 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo arreglos de registros. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.		

Tabla 72: Décimo quinta Semana, 2º período, actividad complementaria

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Operaciones sobre estructuras de datos del tipo arreglos de registros. Implementación de funciones y programa principal para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.	
Tipo de tarea:	Implementar y probar programas que utilicen funciones matemáticas y de manejo de cadenas de caracteres.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y establecer las funciones que deben usarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y las funciones a utilizar.	5 minutos
	Individualmente, implementar y probar el programa.	25 minutos
	Presentar, argumentar y defender la implementación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo arreglos de registros. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.	

Tabla 73: Décimo quinta Semana, 3º período

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Operaciones sobre estructuras de datos del tipo arreglos de registros. Implementación de funciones y programa principal para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.		
Descripción de la tarea:	Implementar funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros, e implementar y probar el programa principal.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las funciones que se requieren.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la caracterización de las funciones.	5 minutos	
	Implementar y probar el programa de solución	23 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo arreglos de registros. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.		

Tabla 74: Décimo quinta Semana, 3º período, actividad complementaria

Décimo sexta Semana:

En el primer período de esta semana (tabla 75), se desarrolla una tercera actividad de trabajo cooperativo que es continuación de las dos anteriores, en el sentido que se propondrán tres nuevos problemas referidos a la implementación de funciones y el programa principal para operaciones sobre estructuras del tipo arreglos de registros, en los cuales se trata de operaciones que implican el manejo de dos o más estructuras de datos de este tipo.

Tipo de actividad:	Trabajo cooperativo	
Duración:	90 minutos	
Aspectos a trabajar:	Operaciones sobre estructuras de datos del tipo arreglos de registros. Implementación de funciones y programa principal para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.	
Tipo de tarea:	Implementar y probar programas que utilicen funciones matemáticas y de manejo de cadenas de caracteres.	
Distribución de tiempos:	Explicación de la tarea y los propósitos;	5 minutos
	Trabajo de los grupos cooperativos;	50 minutos
	Presentación y defensa de las soluciones de los grupos;	25 minutos
	Análisis colectivo del desarrollo de la actividad;	10 minutos
Acciones de los estudiantes:	Individualmente, interpretar el enunciado del problema y establecer las funciones que deben usarse.	5 minutos
	Discutir, argumentar y establecer consensos respecto de la interpretación del problema y las funciones a utilizar.	5 minutos
	Individualmente, implementar y probar el programa.	25 minutos
	Presentar, argumentar y defender la implementación del programa.	10 minutos
	Discutir y argumentar elementos para establecer la mejor implementación del programa.	5 minutos
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo arreglos de registros. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.	

Tabla 75: Décimo sexta Semana, 1º período

La actividad complementaria a la del primer período se muestra en la tabla 76, la que se ha de realizar por los estudiantes en horario extra clase.

Tipo de actividad:	Trabajo cooperativo complementario fuera de la clase		
Duración estimada:	360 minutos		
Aspectos a trabajar:	Operaciones sobre estructuras de datos del tipo arreglos de registros. Implementación de funciones y programa principal para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.		
Descripción de la tarea:	Implementar funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros, e implementar y probar el programa principal.	Cantidad de problemas planteados:	8
Acciones de los estudiantes por cada problema:	Lectura interpretativa del enunciado del problema y caracterización de las funciones que se requieren.	5 minutos	
	Discutir, argumentar y establecer consensos respecto de la interpretación del enunciado del problema y de la caracterización de las funciones.	5 minutos	
	Implementar y probar el programa de solución	23 minutos	
	Presentar, argumentar y defender la implementación realizada	7 minutos	
	Discutir y argumentar elementos para elegir la mejor solución.	5 minutos	
Competencias a desarrollar:	Específicas disciplinares; Capacidad para: Usar y diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo arreglos de registros. Implementar y probar programas en basados en funciones para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.		

Tabla 76: Décimo sexta Semana, 1º período, actividad complementaria

En el segundo y tercer períodos de esta, la ultima semana de clases, se realiza la tercera evaluación acumulativa (tabla 77)

2º y 3° Períodos

Tipo de actividad:	Tercera Evaluación Acumulativa
Duración:	180 minutos
Contenidos a evaluar:	Implementación y prueba de programas basados en funciones para: Realizar operaciones básicas sobre estructuras de datos del tipo vector y arreglos bi y tridimensionales, incluidas operaciones de ordenamiento y búsqueda, y Realizar operaciones básicas sobre estructuras de datos del tipo arreglos de registros.

Tabla 77: Décimo sexta Semana, 2º y 3º períodos

En estos términos, las competencias que se medirán en esta evaluación son:

- Diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo vector, matrices y arreglos tridimensionales.
- Utilizar los algoritmos de ordenamiento y búsqueda.
- Diseñar algoritmos para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.
- Implementar las funciones y el programa principal para realizar operaciones sobre estructuras de datos del tipo vector, matrices y arreglos tridimensionales, incluyendo las operaciones ordenamiento y búsqueda.
- Implementar las funciones y el programa principal para realizar operaciones sobre estructuras de datos del tipo arreglos de registros.

5.5. Instrumentos para la recopilación de datos

Teniendo en cuenta que nuestro principal propósito en esta investigación, es mejorar el rendimiento académico de los estudiantes de la asignatura de FP, a través de una metodología que apunte fundamentalmente al aprendizaje, el desarrollo y perfeccionamiento de las habilidades de pensamiento y las estrategias de resolución de problemas de procesamiento de datos, hemos considerado necesario utilizar técnicas e instrumentos para la recopilación de datos en tres momentos del proceso de enseñanza y aprendizaje que nos parecen importante: al inicio del proceso de enseñanza y aprendizaje, durante el propio proceso de enseñanza y aprendizaje, que es el momento en que llevaremos a cabo la intervención, y al final del mismo.

Al inicio del proceso de enseñanza y aprendizaje utilizamos una prueba con el objeto de establecer el nivel de dominio que poseen los estudiantes de los grupos experimental y control respecto del manejo de conocimientos declarativos y procedimentales. Esta prueba constituye el pre-test, y en ella se evaluaron los siguientes cuatro ámbitos:

1.- Cálculo,
2.- Manejo de expresiones algebraicas,
3.- Algebra de Boole, y
4.- Resolución de problemas.

Aunque la asignatura de FP, en las mallas curriculares de las dos universidades de las cuales hemos tomado los grupos experimental y control, no se contemplan requisitos previos, hemos considerado que es importante conocer el dominio que tienen los estudiantes en los cuatro ámbitos señalados.

Desde la perspectiva del propio desarrollo de la asignatura, se presupone que los estudiantes poseen dominio sobre estos conocimientos, sin embargo como analizamos en apartados anteriores donde tratamos con las dificultades que presentan los estudiantes, varias investigaciones en esta línea, señalan que existen dificultades en el manejo de estos. Mediante este instrumento pretendemos conocer cuál es la realidad de nuestros estudiantes a este respecto.

Esta prueba considera 4 ítems para el ámbito Cálculo, 22 para el ámbito de Manejo de expresiones algebraicas, 5 para el ámbito de Algebra de Boole, y 7 para el ámbito de Resolución de problemas.

Durante el proceso de enseñanza y aprendizaje y con el propósito de determinar la influencia de la utilización del método de instrucción con actividades de aprendizaje cooperativo, sobre el grado de logro académico de los estudiantes, se han aplicado tres pruebas a los grupos experimental y grupo control en cada semestre de

intervención. Estas pruebas nos permiten también medir los logros en tres momentos del desarrollo de la asignatura.

Para precisar estas mediciones, se han definido un conjunto de competencias que los estudiantes deberían alcanzar con el desarrollo de la misma.

Para una mejor definición, hemos establecido tres categorías de competencias, y dentro de cada una de ellas, se han definido sub-categorías de orden más específico. Las categorías y sub-categorías son:

1 Capacidad para comprender y explicar los fundamentos teórico-formales de la programación de computadores.
   1.1 Comprender y explicar las bases teóricas de la programación
   1.2 Comprender y explicar el modelo de programación de un computador

2 Capacidad para aplicar la metodología de la programación a la resolución de problemas de procesamiento de datos.
   2.1 Comprender los propósitos de cada etapa de la metodología de la programación
   2.2 Aplicar los procedimientos para desarrollar las etapas de la metodología
   2.3 Utilizar las distintas formas de representación algorítmica
   2.4 Diseñar algoritmos de solución

3 Capacidad para construir e implementar programas computacionales.
   3.1 Declarar y utilizar correctamente los tipos de datos simples
   3.2 Utilizar correctamente la jerarquía de operadores
   3.3 Utilizar correctamente las estructuras de programación
   3.4 Declarar y utilizar correctamente estructuras de datos del tipo arreglos
   3.5 Utilizar algoritmos de ordenamiento sobre arreglos
   3.6 Utilizar algoritmos de búsqueda sobre arreglos
   3.7 Utilizar funciones de la biblioteca estándar
   3.8 Implementar funciones
   3.9 Construir e implementar programas a partir de su representación algorítmica

La intervención acaba cuando finaliza el semestre académico, instancia en la cual, se aplicó un pos-test con el propósito de evaluar las competencias de salida que alcanzaron los estudiantes.

Las competencias que han sido evaluadas en cada prueba semestral y en el post-test son:

- Primera prueba mide las competencias 1.1, 1.2, 2.1, 2.2, 2.3, y 2.4
- Segunda prueba mide las competencias 2.1, 2.2, 2.3, 2.4, 3.1, 3.2, 3.3, y 3.9

- Tercera prueba mide las competencias 2.4, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, y 3.9
- Post-test mide las competencias 2.4, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, y 3.9

Como se muestra en este detalle, existen varias competencias que dada la cronología de desarrollo de los contenidos y la complejidad de las propias competencias, algunas han sido medidas en más de una de estas pruebas. Quizás el caso más evidente es la competencias 2.4 que es evaluada en las tres pruebas ya que ella corresponde a la capacidad para diseñar algoritmos de solución, que es una capacidad que durante el desarrollo de los contenidos de la asignatura, se ha de manifestar en la resolución de problemas de procesamiento de datos de dificultad creciente, conforme a la cronología de desarrollo de los contenidos de la propia asignatura.

Para cada evaluación, y para cada situación problema planteada, el logro de una competencia es medida en la escala de valorización de 1.0 a 7.0. Por su parte, la valorización global del logro de la competencia para una prueba en particular, se ha obtenido promediando los logros parciales de la competencia para cada situación problema que han debido resolver los estudiantes en la prueba.

El tercer instrumento utilizado, es una encuesta para medir el grado de satisfacción de los estudiantes respecto de la metodología de instrucción con actividades de aprendizaje cooperativo.

Este instrumento considera 24 ítems que son valorizados mediante una escala Likert de cinco niveles, el cual fue aplicado a los grupos experimentales en las dos intervenciones.

5.6. Variables y sus medidas

En la presente investigación se tomarán en cuenta las siguientes variables:

1 Forma de conducción del proceso de enseñanza y aprendizaje: Para el grupo experimental, el desarrollo de la asignatura se llevará a cabo integrando de manera intensiva actividades de aprendizaje cooperativo, en tanto para el grupo control, la conducción del proceso de enseñanza aprendizaje se realizará conforme al método tradicional. Para esta investigación, esta es la variable independiente
2 Logro de las competencias: La medida del nivel de logro de las competencias, corresponderá a la variable dependiente.

Las competencias a medir han sido organizadas en tres categorías conforme a la siguiente definición:

1. Capacidad para comprender y explicar los fundamentos teórico-formales de la programación de computadores, la que se descompone en:
   1. 1. Comprender y explicar las bases teóricas de la programación, y
   1. 2. Comprender y explicar el modelo de programación de un computador.

2. Capacidad para aplicar la metodología de la programación a la resolución de problemas de procesamiento de datos, la que descompone en:
   2. 1. Comprender los propósitos de cada etapa de la metodología de la programación,
   2. 2. Aplicar los procedimientos para desarrollar las etapas de la metodología,
   2. 3. Utilizar las distintas formas de representación algorítmica, y
   2. 4. Diseñar algoritmos de solución.

3. Capacidad para construir e implementar programas computacionales, la que se descompone en:
   3. 1. Declarar y utilizar correctamente los tipos de datos simples,
   3. 2. Utilizar correctamente la jerarquía de operadores,
   3. 3. Utilizar correctamente las estructuras de programación,
   3. 4. Declarar y utilizar correctamente estructuras de datos del tipo arreglos,
   3. 5. Utilizar algoritmos de ordenamiento sobre arreglos,
   3. 6. Utilizar algoritmos de búsqueda sobre arreglos,
   3. 7. Utilizar funciones de la biblioteca estándar,
   3. 8. Implementar funciones, y
   3. 9. Construir e implementar programas a partir de su representación algorítmica.

El nivel de logro intermedio de estas competencias se medirá durante el semestre académico, mediante tres pruebas acumulativas. En base a la secuencialidad en el tratamiento de los contenidos, las competencias que se evaluarán en cada prueba son:

- Primera prueba: 1.1, 1.2, 2.1, 2.2, 2.3, y 2.4
- Segunda prueba: 2.1, 2.2, 2.3, 2.4, 3.1, 3.2, 3.3, y 3.9
- Tercera prueba: 2.4, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, y 3.9

El logro final de competencias, se mide mediante una prueba post-test que se aplica al final del semestre académico. Para estos efectos se ha hecho una selección de aquellas competencias que por su definición incluyen necesariamente el logro de otras más específicas. Por tanto en la prueba post-

test se medirán las competencias 2.2, 2.3, 2.4, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, y 3.9:

El nivel de logro de cada competencia será medida en la escala de calificación universitaria chilena, y que corresponde al intervalo comprendido entre 1,0 (uno, cero) y 7,0 (siete, cero), con nota mínima de aprobación 4,0 (cuatro, cero).

3 Grado de satisfacción de los estudiantes: La medida del grado de satisfacción de los estudiantes del grupo experimental, referida a la forma de conducción del proceso de enseñanza y aprendizaje. Para ello se aplicará una encuesta estructurada basada de varios ítems con el propósito de recoger información respecto de la evaluación del grado de satisfacción que hacen los estudiantes con la metodología aplicada para la conducción del proceso de enseñanza y aprendizaje.

5.7. Procedimiento

El diseño del experimento considera el desarrollo de cuatro fases:

Fase I; Evaluación diagnóstica,
Fase II; Aplicación del plan de clases,
Fase III, Pos test, y
Fase IV; Medición del nivel de satisfacción de los estudiantes

A continuación se detallan estas cuatro fases.

FASE I; Evaluación diagnóstica

Aunque en la definición curricular de la carrera de Ingeniería en Informática, tanto de la Universidad de Playa Ancha, como de la Universidad de Valparaíso, que son las dos entidades de las cuales hemos tomado el grupo experimental y los grupos control para esta investigación, no se consideran asignaturas de prerrequisitos para cursar la asignatura de Fundamentos de Programación, y por tanto, curricularmente no se considera necesario el manejo de conocimientos previos, para nosotros, tanto desde la perspectiva pedagógica, como para los propósitos de esta investigación, consideramos que es absolutamente necesario evaluar el manejo de un conjunto de conocimientos, que creemos que son absolutamente necesarios para abordar el aprendizaje de la asignatura.

En esta perspectiva, consideramos que existen un conjunto de conocimientos de tipo declarativos y de tipo procedimentales sobre los cuales, los estudiantes deberían tener dominio para abordar de buena manera esta asignatura.

Como lo hemos señalado, la asignatura de Fundamentos de Programación, en su calidad de primera asignatura que trata con el núcleo de conocimientos de

programación de los planes de estudio de este tipo de carreras, tiene como propósito principal, la enseñanza y el aprendizaje de los fundamentos teóricos y metodológicos de la programación de computadores, ámbito en el cual también se aborda el aprendizaje de los aspectos aplicativos de la metodología aplicada a la resolución de los problemas de procesamiento de datos y de un lenguaje de programación.

En este contexto, el desarrollo la asignatura, trata con los aspectos aplicativos de la metodología de la programación, abordando la resolución de problemas de tipo matemático, por tanto se requiere que los estudiantes manejen conocimientos declarativos y procedimentales que son propios de los contenidos de las asignaturas de matemática que cursaron durante su enseñanza media, que es el nivel educativo anterior a las carreras de pre-grado universitario.

Consideramos dos aspectos del diagnóstico que son relevantes para esta investigación, que son:

A.- Disponer de información relevante para orientar el proceso de enseñanza y aprendizaje de la asignatura, para lo que será necesario diagnosticar el manejo que tienen los estudiantes de los conocimientos y procedimientos que implicados en cada reactivo del test de diagnóstico, y

B.- Disponer de información para determinar la relación de equivalencia entre el grupo experimental y control, considerando que el diagnostico se aplicará a ambos grupos.

Para estos efectos, las áreas de conocimientos matemáticos que se evaluaran mediante el test de diagnóstico son:

1. *Cálculo*; en esta área nos interesa evaluar el manejo de conocimientos:
   - ♦ Declarativos, referidos a:
   - ♦ Razón de proporcionalidad y
   - ♦ Porcentaje.
   - ♦ Procedimentales, referidos al manejo y uso de los procedimientos matemáticos de cálculo de:
   - ♦ Proporcionalidades y
   - ♦ Porcentajes.

   Esta área será evaluada mediante cuatro ítems, y para cada uno de ellos, se evaluará por separado el uso de los conocimientos y de los procedimientos. Esta separación en la evaluación nos permitirá disponer de información más acotada de cada aspecto del ámbito de estos conocimientos.

2. *Expresiones algebraicas;* en esta área nos interesa evaluar el manejo de conocimientos:
   - ♦ Declarativos, referidos a:
   - ♦ Concepto de expresión algebraica,

- Términos de una expresión algebraica,
- Constantes y variables de una expresión algebraica,
- Incógnita en una expresión algebraica,
- Función algebraica,
- Ecuación de primer grado,
- Sistema de ecuaciones de primer grado,
- Algoritmos de resolución de sistemas de ecuaciones de primer grado,
- Ecuación de segundo grado,
- Algoritmo de resolución de ecuaciones de segundo grado.
- Procedimentales, referidos a la:
- Aplicación de los procedimientos matemáticos para reducir y simplificar expresiones algebraicas,
- Aplicación de los procedimientos matemáticos para determinar el valor de la incógnita en una ecuación de primer grado,
- Construcción de relaciones algebraicas a partir del enunciado de problemas,
- Aplicación de los algoritmos para determinar el valor de las incógnitas en sistemas de ecuaciones de primer grado,
- Aplicación de los procedimientos para evaluar una función algebraica para un cierto valor de la variable de la función, y
- Aplicación del algoritmo para resolver ecuaciones de segundo grado.

Esta área será evaluada mediante 23 ítems, y para cada uno de ellos, se evaluará por separado el uso de los conocimientos y de los procedimientos. Esta separación en la evaluación nos permitirá disponer de información más acotada de cada aspecto del manejo de los conocimientos.

Dadas las características de los tipos de problemas que se abordan en un primer curso de programación, estos conocimientos son particularmente importantes para el educado desenvolvimiento por parte de los estudiantes.

Como se ha señalado y ejemplificado en varios de los apartados de esta tesis, parte importante de la práctica referida al diseño de algoritmos y la programación en los primeros cursos de programación, se lleva a cabo utilizando problemas de naturaleza matemática.

3. *Algebra de Boole;* en esta área nos interesa evaluar el manejo de los operadores lógicos y (and) y o (or), y operandos del tipo p=verdadero y q=falso, y su correspondiente determinación del resultado lógico (verdadero o falso), en dos tipos de situaciones de uso de operadores y operandos, en expresiones que involucren:
   - Un operador y dos operandos, y
   - Varios operadores y operandos

El manejo de operadores y operandos de la lógica de primer orden en el ámbito de la lógica proposicional, es conocimiento que los estudiantes debieran desarrollar en los cursos de matemática en el nivel de enseñanza obligatoria anterior a la universidad, y que en el contexto de un primer curso de programación de nivel universitario, es el conocimiento que se requiere para tratar con las proposiciones lógicas que se incluyen en el diseño de los algoritmos de solución de los problemas de procesamientos de datos y en los programas computacionales.

Son el recurso formal para controlar los flujos de secuencias de acciones y operaciones, tanto en un algoritmo como en un programa computacional.

Esta área la evaluaremos con cinco ítems en el ámbito de procedimiento, ya que consideramos que lo fundamental, es la capacidad para determinar el resultado lógico al operar con un operador lógico y dos operandos, y cuando se trata con expresiones lógicas que tienen varios operadores y sus correspondientes operandos.

4. *Resolución de problemas;* en esta área nos interesa evaluar el manejo y uso de capacidades generales para resolver problemas, ámbito en el cual queremos evaluar la capacidad para leer e interpretar el enunciado de problemas matemáticos simples, y la capacidad para describir la secuencias de acciones y operaciones para resolverlos.

   En sentido estricto, nos interesa medir la capacidad que tienen los estudiantes para elaborar un plan de solución para un problema y para describir dicho plan como una secuencia de acciones y operaciones que deben ser ejecutadas para resolver el problema.

   Este aspecto nos interesa, ya que conforme a lo que hemos argumentado en varios de los apartados anteriores, basándonos tanto en las investigaciones como en nuestra experiencia, existen evidencias que muestran que las capacidades generales para resolver problemas, juegan un rol decisivo en el aprendizaje de las capacidades para la resolución de problemas en áreas más específicas, como es el caso de la programación de computadores.

   En esta perspectiva, la capacidad para elaborar y describir un plan de solución para un problema, en este caso matemático, es para nosotros un conocimiento de base fundamental en el aprendizaje de las capacidades para resolver problemas de procesamiento de datos y para programar computadores.

   Para evaluar esta área hemos considerado siete ítems, y para cada uno de ellos evaluaremos tanto el uso de los conocimientos que se hace para cada

situación problema planteado, como la validez de los procedimientos que se describen para resolverlas

FASE II; Desarrollo de la intervención

Tomando en consideración los tipos de contenidos que deben aprender los estudiantes, así como la complejidad de las capacidad que han de desarrollar y perfeccionar, como también los elementos que se deben tener presente para que ellos adquieran las capacidades, cuestiones que hemos presentado y analizado en los capítulos 2 y 3, y atendiendo a la definición que planteamos respecto de los principios básicos que a nuestro juicio deben orientar el proceso de aprendizaje que se sintetizan en:

- Conocer y comprender de manera significativa los fundamentos teórico-formales de la disciplina,
- Adquirir y desarrollar las capacidades para identificar, seleccionar, combinar y organizar los diversos tipos de conocimientos que se implican en la solución de un problema,
- Adquirir, desarrollar y perfeccionar las estrategias cognitivas y de pensamiento que se requieren para el aprendizaje y el uso de los conocimientos teórico-formales, en especial las estrategias cognitivas de regulación, de control y de evaluación de la validez de los aprendizajes, y
- Adquirir y desarrollar estrategias y habilidades para el trabajo en equipo.

Por lo que hemos diseñado un plan de clases multiestructurado, que permita la presentación y explicitación de los contenidos por parte del profesor, pero que también responda de la mejor manera posible a la creación de ambientes de enseñanza y aprendizaje que promueva y favorezca la práctica intencionada y sistemática por parte de los estudiantes, para posibilitar el aprendizaje, el desarrollo y el perfeccionamiento de las capacidades, es decir un espacio intencionado que demande en ellos, la construcción, el uso y aplicación de los conocimientos implicados, favoreciendo el desarrollo de las estrategias cognitivas que demanda la tarea de resolver los problemas.

En esta perspectiva, este plan de clases que es la base de la intervención, considera instancias para el tratamiento de los conocimientos teórico-formales, de los fundamentos, de los procedimientos y las técnicas, pero también considera de manera importante, instancias que favorezcan la adquisición, el desarrollo y el perfeccionamiento de las capacidades y habilidades que los estudiantes necesitan para actuar como buenos resolutores de problemas de procesamiento de datos.

Así, el plan de clases para el desarrollo de la asignatura considera el desarrollo de clases expositivas para el tratamiento y explicitación de los conocimientos, y el desarrollo de actividades basadas en el aprendizaje cooperativo como instancias

para favorecer el aprendizaje, el desarrollo y el perfeccionamiento de las capacidades y habilidades para ejecutar las tareas intelectuales que son propias del uso y aplicación de los conocimientos.

Mediante la puesta en práctica de este tipo de actividades de aprendizaje, pretendemos también que nuestros estudiantes desarrollen y perfeccionen el uso y aplicación de las estrategias de regulación y control que se requieren para el correcto desarrollo de la tarea intelectual de resolución de los problemas de procesamiento de datos.

Por tanto, con las actividades de tipo cooperativo trabajamos los aspectos de uso y aplicación de los conocimientos para la resolución de los problemas, pero por sobre todo trabajamos los aspectos de construcción del conocimiento para la resolución de los problemas y trabajamos también en el desarrollo y el perfeccionamiento de las capacidades y habilidades para estos propósitos.

Mediante las actividades de trabajo cooperativo apuntamos a favorecer condiciones para la construcción del conocimiento, y para el desarrollo y perfeccionamiento de las capacidades y las habilidades requeridas para resolver los problemas.

Con estas actividades generamos condiciones para que los estudiantes organizados en los grupos cooperativos, confronten ideas, con el objeto de favorecer la construcción, organización de sus conocimientos y la expresión y defensa de argumentos en relación con el uso y aplicación de ellos, al tiempo que registran, buscan y comparan información, y piensan que hacer y cómo usar la información que están recibiendo de sus pares, con lo cual, tendrán mejores oportunidades de alcanzar los aprendizajes requeridos.

Además, con el objeto de extender las prácticas de trabajo cooperativo, y con ello disponer de más instancias para el aprendizaje de los conocimientos y las capacidades que demanda la resolución de los problemas de procesamiento de datos, y junto con ello perfeccionar el desempeño de la propia práctica cooperativa, se propondrá a los estudiantes el desarrollo de actividades cooperativas para que sean desarrolladas fuera del horario regular de las clases y actividades propias de la asignatura.

Así entonces, en el diseño del plan de clases hemos considerado los siguientes tipos de actividades:

- Sesiones de clases expositivas; con las cuales se presentan y desarrollan:
  - ♦ Los contenidos referidos a los fundamentos teórico-formales de la disciplina de la programación,
  - ♦ Los contenidos referidos a la metodología para aplicar los fundamentos teórico-formales de la disciplina,

  - ♦ Los ejercicios de ejemplo mediante los cuales se muestran y explicitan los aspectos prácticos de la aplicación de los fundamentos teórico-formales de la disciplina y la metodología, cuando se usan en las tareas de resolución de los problemas.

- Sesiones de aprendizaje cooperativo, compuestas por actividades diseñadas con el objeto de poner a los estudiantes en situaciones de trabajo colectivo y socializado que contribuyan a favorecer y potenciar el aprendizaje, el desarrollo y el perfeccionamiento de las capacidades para resolver problemas de procesamiento de datos y la programación, y
- Sesiones de evaluación de los aprendizajes, que permitan medir los logros del aprendizaje alcanzado por los estudiantes. Además, estas evaluaciones permiten obtener las calificaciones de cada estudiante como parte de las exigencias del desarrollo de la asignatura.

Estas tres categorías de sesiones se han organizado en base a sus intencionalidades y a los propósitos de la investigación, en las 16 semanas de clases que considera curricularmente el desarrollo de la asignatura.

Conforme a la carga académica de la asignatura establecida en el plan de estudios, en cada semana se consideran tres períodos de 1,5 horas de duración, lo que arroja un total de 72 horas para el semestre académico.

Ahora bien, asimilando el modelo europeo (ECTS) de asignación de crédito, a la asignatura de FP le hemos asignado 6 créditos, y usando el mismo referente, estamos considerando que cada crédito equivale a 30 horas (en el modelo europeo se considera 1 crédito equivale a 25 o 30 horas), con lo cual, el equivalente en horas de la asignatura es de 180 horas.

Se han programado un total de 72 horas de actividades presenciales (actividades de aula) que detallamos en el plan de clases a continuación.

Además de esto, en el plan de clases hemos considerado la realización de 20 actividades de trabajo cooperativo que deberán ser realizadas por los estudiantes en horario extra clase.

Para el desarrollo de estas actividades, se entregarán a los estudiantes guías de ejercicios complementarios, para ser desarrolladas preferentemente de manera grupal. Se entregara una de estas guías complementarias después de cada una de las 20 actividades cooperativas que se han considerado dentro del plan de clases.

Hemos ponderado que para desarrollar cada una de estas guías complementarias, los grupos cooperativos requieren aproximadamente de unas 6 horas de trabajo.

La conformación de grupos los cooperativos se corresponde con la definición de grupos formales y de base, con lo cual son grupos permanentes durante todo el semestre académico, en la idea de así favorecer el afiatamiento entre sus integrantes, y de esta forma alcanzar buenos niveles de desempeño del propio grupo, en la realización de las tareas académicas e intelectuales que se demanden, con lo cual apuntamos a crear buenas condiciones y ambientes que permitan optimizar el proceso de aprendizaje de los conocimientos y las capacidades disciplinares.

En lo fundamental pretendemos que los estudiantes dentro de sus grupos cooperativos, participen de manera activa en la:

- Organización de las tareas intelectuales que se requieren para resolver problemas de procesamiento de datos, y
- Discusión, negociación y argumentación colectiva de significados respecto del manejo y aplicación de los conocimientos implicados en la resolución de los problemas.

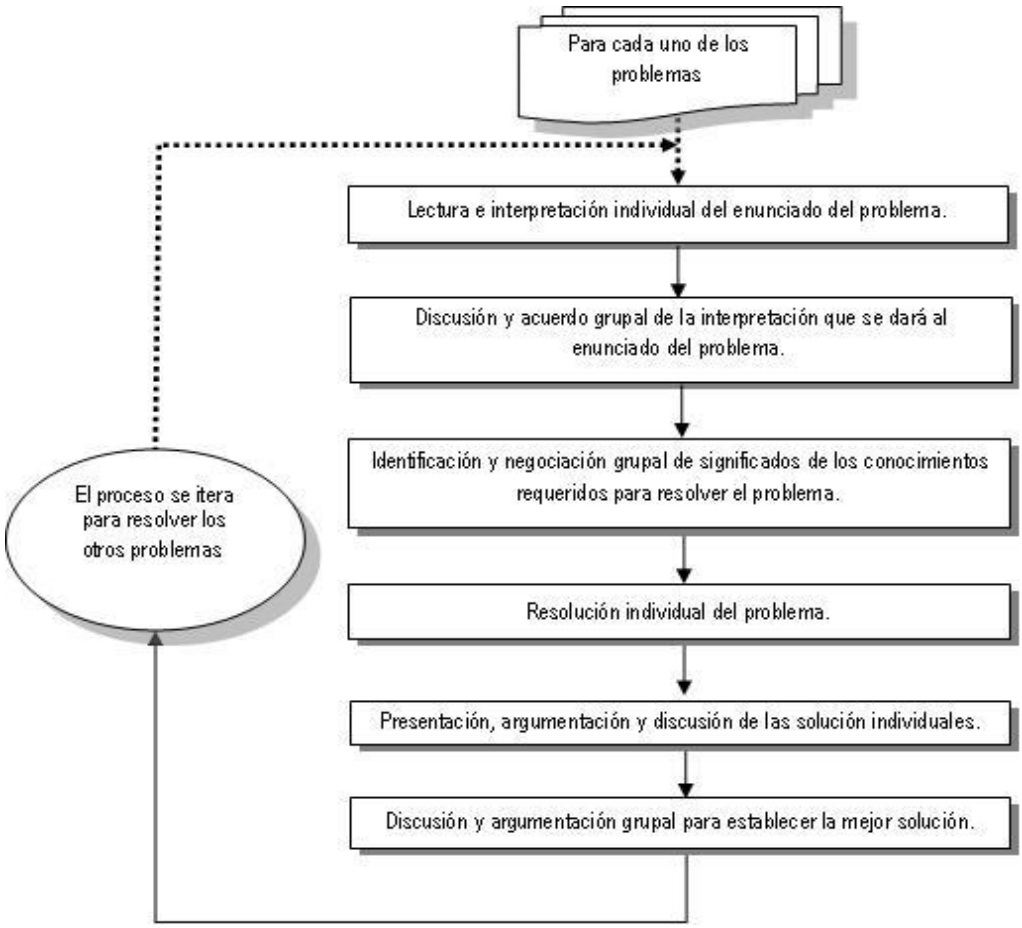

Figura 16: Secuencia general de pasos en las actividades cooperativas

En términos más precisos, con las actividades cooperativas, pretendemos que exista:

- Generación de desequilibrios cognitivos que estimulen el aprendizaje, la creatividad y el desarrollo cognitivo y social que es pertinente a la tarea de resolver los problemas, con lo cual esperamos mejorar la comprensión de los conocimientos disciplinares y la calidad del razonamiento,
- Exposición de ideas variadas de los integrantes del grupo referidas a múltiples perspectivas para entender y abordar la resolución de los problemas, y de la forma de manejar y aplicar los conocimientos, con lo cual esperamos favorecer los aprendizajes, y
- Involucramiento activo hacia una construcción de un pensamiento más elaborado, mediante las acciones de dar y recibir explicaciones, adoptando con más frecuencia puntos de vista personales para discutir respecto del manejo de los conocimientos y su aplicación.

Para conseguir estos propósitos, el abordaje de las situaciones problemas que propondremos para, desarrollar en las actividades cooperativas se organizará en la siguiente secuencia general de pasos

FASE III; Evaluación Post-test

En esta fase se aplica el pos-test con el objeto de evaluar el nivel de logro alcanzado por los estudiantes, en las competencias que hemos establecido como de salida una vez finalizado el desarrollo de la asignatura.

Estas competencias son:

- La competencia 2.4, referida a la capacidad para diseñar algoritmos de solución
- La competencia 3.1, referida a la capacidad para declarar y utilizar correctamente los tipos de datos simples
- La competencia 3.2, referida a la capacidad para utilizar correctamente la jerarquía de operadores
- La competencia 3.3, referida a la capacidad para utilizar correctamente las estructuras de programación
- La competencia 3.4, referida a la capacidad para declarar y utilizar correctamente estructuras de datos del tipo arreglos
- La competencia 3.5, referida a la capacidad para utilizar algoritmos de ordenamiento sobre arreglos
- La competencia 3.6, referida a la capacidad para utilizar algoritmos de búsqueda sobre arreglos
- La competencia 3.7, referida a la capacidad para utilizar funciones de la biblioteca estándar
- La competencia 3.8, y referida a la capacidad para implementar funciones

- La competencia 3.9 referida a la capacidad para construir e implementar programas a partir de su representación algorítmica

Con el objeto de asegurar que la medición de estas competencias refleje de manera lo más fiel posible, los logros en el aprendizaje alcanzado por los estudiantes, se debe poner especial cuidado en que el instrumento considere problemas que no hayan sido planeados a los estudiantes, en ninguna de las instancias de desarrollo de la asignatura, así como también en ninguna de las evaluaciones anteriores.

FASE VI; Encuesta de satisfacción de los estudiantes.

Mediante esta encuesta se persigue el propósito de conocer el nivel de satisfacción de los estudiantes del grupo experimental, respecto de la modalidad usada para el desarrollo de la asignatura y respecto del sentimiento que tienen respecto a si consideran que las actividades cooperativas han sido útiles para crear mejores condiciones para el aprendizaje, y han contribuido a las mejoras en sus aprendizaje.

5.8. Procedimiento y técnicas de análisis de los datos

Dada la naturaleza de la investigación con datos de tipo cuantitativo, evaluaciones de rendimiento, y cualitativo, dinámicas del trabajo cooperativo y satisfacción de los estudiantes, se atenderá en cada caso a las técnicas de esta naturaleza.

En el caso del análisis relativo a la parte experimental y control de sus resultados se tarbajará con competenecias esperadas en el aprendizaje con relación a la asignatura de referencia, por lo que serán considerados indicadores ad hoc y específicos al conocimiento no solo declarativo sino también procedimental.

Por ello, además de verificar si el rendimiento ha sido o no superior en cada caso, medido en términos de resultado final, conviene también considerar la arquitecturas de las propias competencias al establecerse en éstas conocimientos de naturaleza distinta, cognitivo, habilidad, destreza, aplicación, desarrollo, etc. Es decir, valorando si la cohesión de los elementos integradores de las competencias se mantiene, mejora o se deteriora.

Para el tratamiento de la información cuantitativa se ha previsto el uso del paquete estadístico SPSS que facilitará confirmar la fiabilidad de los instrumentos de medida y además realizar los análisis pertinentes, descriptivos, factoriales, agrupamiento-interelación de las competencias mediante dendograma, así como las necesarias correlaciones, coeficientes de Pearson, diferencia de medias y la la verificación de variables asociadas mediante tablas de contigencia al objeto de valorar la aportación de la metodología aplicada en el aprendizaje de los estudiantes y que nos permita comparar con la que veía siendo práctica habitual. Es decir:

a. Un análisis de “Prueba T para medias de grupos independientes” al objeto de valorar las diferencias que pudieran darse entre los grupos experimental y de control respectos de:
    i. No diferencias intergrupales de la media en el pretest
    ii. Si dferencias intergrupales de la media en el postest
    iii. Si o no diferencias intragrupales pretest-postest en cada uno de los grupos

b. Análisis asociados al Pretest:
    i. De los resultados del pre-test (diagnóstico) para las cuatro áreas evaluadas (Cálculo, Expresiones algebraicas, Algebra de Boole y Resolución de problemas). El análisis se realiza teniendo en cuenta las medias y desviaciones típicas alcanzadas por el grupo experimental y control para cada área.
    ii. Análisis factorial para determinar el nivel de explicación que se puede deducir de los resultados del pre-test (diagnóstico).
    iii. Análisis de conglomerados jerárquicos para determinar las agrupaciones de ítems del pre-test (diagnóstico), para el grupo experimental y control.

c. Evaluación contínua durante el desarrollo e implementación del modelo.
    i. Estimación del Chi-cuadrado de Cochran para evaluar la fiabilidad de las mediciones de logros de competencias para la primera, segunda y tercera prueba acumulativa del semestre académico y para el Post-test.
    ii. Análisis de resultados de la evaluación de logro de competencias para la primera, segunda y tercera prueba acumulativa del semestre, y para el post-test, tanto para el grupo experimental y control.
    iii. Análisis de los porcentajes de logro de competencias para la primera, segunda y tercera prueba acumulativa del semestre, y para el post-test, tanto para el grupo experimental y control.
    iv. Análisis respecto de los niveles de logro para el conjunto de indicadores del trabajo cooperativo, para los tres momentos, en el grupo experimental.

d. Otros análisis asoiados al postest:
    i. Análisis de conglomerados jerárquicos para determinar las agrupaciones de ítems del post-test, para el grupo experimental y control.
    ii. Análisis de comparación entre aprobados y reprobados para el grupo experimental y control.

e. Análisis de datos cualitativos y nivel de satisfacción
    i. Análisis de indicadores que definen la consistencia del trabajo cooperativo, derivado de las observaciones de campos de esta dinámica en los grupos.
    ii. Análisis de los resultados de la encuesta de satisfacción aplicada al grupo experimental, en base a las medias y desviaciones típicas de cada ítem

CAPITULO 6

RESULTADOS SU ANÁLISIS Y DISCUSIÓN

6. RESULTADOS SU ANÁLISIS Y DISCUSIÓN

INTRODUCCIÓN

En este apartado, analizaremos y discutiremos en primer término, los resultados que arrojó la evaluación de conocimientos previos, para el grupo experimental y el grupo control.

Como se ha de recordar, la evaluación de estos conocimientos, corresponde al pre-test que se aplicó a ambos grupos (experimental control), con la cual se midió el manejo de conocimientos conceptuales y procedimentales, en los ámbitos de Cálculo, Expresiones algebraicas, algebra de Boole, y Resolución de problemas.

El primer propósito de esta medición, es contrastar la equivalencia entre ambos grupos, para lo cual se determinaran las medias y las desviaciones típicas de los resultados de cada grupo, para cada uno de los cuatro ámbitos.

El segundo propósito en esta medición, es conocer los niveles de logro que cada grupo alcanza en los ámbitos indicados, ya que esto nos entregará información referida al dominio que poseen los estudiantes respecto de los cuatro ámbitos de conocimientos previos, que nos parecen relevantes.

A partir de los resultados, analizamos aquellos aspectos que creemos que son relevantes para el trabajo sobre los contenidos que se tratan, y los aprendizajes que se pretenden en la asignatura de Fundamentos de programación.

Para apoyar nuestro análisis utilizaremos dos herramientas estadísticas: análisis factorial para evaluar el nivel de explicación que se puede deducir de los resultados de la medición; y el análisis de conglomerados jerárquicos mediante su representación como déndograma, lo que nos permitirá analizar las organizaciones de similaridad entre los ítems que han sido medidos.

En segundo término, analizamos los resultados referidos al logro de las competencias. En este contexto se analizan y discuten los resultados alcanzados por el grupo experimental y control, en cada una de las 15 competencias que fueron

definidas, las que se encuentran organizadas en tres categorías: Capacidad para comprender y explicar los fundamentos teórico-formales de la programación de computadores; Capacidad para aplicar la metodología de la programación a la resolución de problemas de procesamiento de datos; y Capacidad para construir e implementar programas computacionales.

El análisis y discusión se realiza respecto de los resultados de tres evaluaciones intermedias, que se aplicaron al grupo experimental y control, durante el semestre académico, y sobre la evaluación pos-test, aplicada al final del semestre académico.

Estas mediciones, su análisis y discusión, tienen carácter fundamental para esta investigación, ya que sus resultados están en directa relación con el principal objetivo de este trabajo.

Este análisis y discusión, se realiza en base a los niveles de logro de las competencias del grupo experimental y control, que fueron medidas en cada una de las evaluaciones indicadas.

En tercer término, se analizan y discuten los resultados referidos a las tasas de Aprobación y Reprobación de los estudiantes, del grupo experimental y control.

Estos resultados tienen una pertinencia, que se relaciona directamente con los resultados en el logro de las competencias, ya que dan cuenta de los resultados globales al finalizar el semestre académico y la intervención, para el caso del grupo experimental, por lo que nos permite, realizar un análisis y discusión, para valorar la eficacia de la propia intervención sobre este grupo.

Luego de lo anterior, realizamos el análisis y discusión de la evolución de los indicadores de calidad del trabajo cooperativo del grupo experimental.

Esto se realiza sobre los resultados de las mediciones de 17 indicadores, organizados en tres categorías: Interdependencia positiva al interior del grupo; Relaciones psicosociales; y Construcción de significados.

El análisis y discusión de cómo estos indicadores dan cuenta de la evolución de la calidad del trabajo cooperativo durante la intervención, nos parece relevante para relacionarlos con los logros en los aprendizajes tanto grupales como individuales, ya que para las intenciones de esta investigación, la forma en que estos indicadores fueron evolucionando durante la intervención, darían cuenta de la evolución de los aprendizaje, reflejándose en el logro de las competencias que alcanzaron los estudiantes de manera individual.

Para estos efectos, el análisis y discusión se ha hecho mediante la organización de las actividades cooperativas, en tres momentos: el primero entre el inicio de la

intervención y la primera evaluación individual de logros de competencias; el segundo, entre la primera y segunda evaluación individual de logros competencias; y el tercero, entre la segunda y tercera evaluación individual de logros de competencias.

Por último, analizamos y discutimos los resultados de la encuesta de satisfacción que se aplicó a los estudiantes del grupo experimental.

Este aspecto nos parece también relevante, toda vez que la disposición de los estudiantes para tomar parte activa y comprometida en un estilo diferente de conducción de la asignatura, está fuertemente condicionada por el grado de satisfacción que ellos tienen con la experiencia de trabajar cooperativamente.

El análisis y discusión se realiza sobre los resultados de la medición de 24 ítems que los estudiantes del grupo experimental evaluaron al final del semestre académico, mediante una escala Likert de cinco categorías.

6.1. Análisis preliminar. Diferencias intergrupos, prueba T pretest-postest

Para este contraste inicial se realiza una prueba T para el análisis de la media de grupos independientes con relación a los rendimientos de los estudiantes en las pruebas Pretest, rendimiento 1, rendimiento 2, rendimiento 3 y Postest.

Como se puede apreciar en la figura 17 las medias son similares para ambos grupos en las pruebas pretest y rendimiento 1, sin embargo se van diferenciando en las pruebas siguientes, incluida la de postest, así como en la acumulada y que aparece denominada promedio de pruebas.

Grupo de pertenencia 1 Experimental 2 Control		N	Media	Desviación típ.	Error típ. de la media
Pretest	1	61	3,7134	,47546	,06088
	2	83	3,7825	,58861	,06461
Nota1	1	61	4,093	1,0942	,1401
	2	83	4,293	1,3234	,1453
Nota2	1	61	4,600	1,0381	,1329
	2	83	3,446	1,3742	,1508
Prueba3	1	61	4,811	1,0944	,1401
	2	83	3,453	1,4905	,1636
Postest	1	61	4,944	1,0697	,1370
	2	83	3,345	1,2721	,1396
Promedio Pruebas	1	61	4,630	,9795	,1254
	2	83	3,651	1,0173	,1117

Figura 17.- Media y desviación típica de las pruebas de rendimientos de los grupos experimental y control

En el análisis de pruebas independientes (figura 18), ya podemos valorar las significatividad de las varianzas y de las medias, hallando que en las pruebas de

pretest ni la diferencia de varianzas ni la de media son significativas, pudiendo afirmar con el 95% de afirmar que ambos grupos, experimental y control son iguales en el momento.

Sin embargo, en las pruebas 2 y tres esas diferencias ya se tornan significativas, tanto para las varianzas como para las medias, pudiendo establecer que los grupos ya se diferencias en estos dos momentos del desarrollo de la asignatura.

Y si bien las varianzas del postest y de la media acumalada de todas las pruebas no dan signficatividad si que se da ésta para las medias de ambas medidas.

		Prueba de Levene para la igualdad de varianzas		Prueba T para la igualdad de medias						
		F	Sig.	t	gl	Sig. (bilateral)	Diferencia de medias	Error típ. de la diferencia	95% Intervalo de confianza para la diferencia	
		Inf	Sup	Inf	Supr	Inferior	Superior	Inferior	Superior	Inferior
Pretest	Se han asumido varianzas iguales	,891	,347	-,754	142	,452	-,06913	,09169	-,25039	,11212
	No se han asumido varianzas iguales			-,779	140,685	,437	-,06913	,08877	-,24463	,10636
Prueba1	Se han asumido varianzas iguales	1,585	,210	-,960	142	,339	-,1993	,2077	-,6100	,2113
	No se han asumido varianzas iguales			-,988	139,978	,325	-,1993	,2018	-,5983	,1997
Prueba2	Se han asumido varianzas iguales	8,452	,004	5,505	142	,000	1,1542	,2097	,7397	1,5687
	No se han asumido varianzas iguales			5,741	141,877	,000	1,1542	,2010	,7568	1,5516
Prueba3	Se han asumido varianzas iguales	6,036	,015	6,022	142	,000	1,3585	,2256	,9126	1,8044
	No se han asumido varianzas iguales			6,307	142,000	,000	1,3585	,2154	,9326	1,7843
Postest	Se han asumido varianzas iguales	1,229	,269	7,966	142	,000	1,5997	,2008	1,2027	1,9967
	No se han asumido varianzas iguales			8,179	139,376	,000	1,5997	,1956	1,2130	1,9864
Promedio Pruebas	Se han asumido varianzas iguales	,002	,964	5,796	142	,000	,9789	,1689	,6450	1,3128
	No se han asumido varianzas iguales			5,829	132,096	,000	,9789	,1679	,6467	1,3111

Figura 18.- Análisis de Diferencias, Prueba T para rendimiento de los estudiantes

Así, por tanto podemos concluir que con este análisis se puede verificar y en consecuencia, afirmar que los grupos experimental y control son diferentes en su rendimiento una vez concluido el desarrollo de la asignatura, por lo que podrá establecerse que la metodología ha tenido una incidencia positiva en el rendimiento del grupo al que le ha sido aplicada.

No obstante el estudio, a pesar de confirmar nuestra H1, sugiere profundizar en el análisis del comportamiento, tanto en lo que supone el propio trabajo cooperativo y las dinámicas de interacción intragrupo, como con relación a los diferentas dominios de las competencias de la asignatura, sí como el proceso evolutivo del aprendizaje con relación a los elementos que integran tales competencias, cuestión ésta que se abordará en los siguientes apartados de este capítulo.

6.2. Resultados referidos a la evaluación de los conocimientos previos

La estimación del Chi-cuadrado de Cochran para la fiabilidad del test de conocimientos arroja un alfa de 0,8569.

En la siguiente tabla muestra los resultados de la prueba de diagnóstico para los grupos experimental (GE) y el grupo control (GC) para las cuatro áreas evaluadas. Los valores de las medias deben ser considerados en la escala de notas que se utiliza en Chile para la evaluación universitaria, la cual está determinada por la escala de 1.0 a 7.0, considerando que la nota 4.0 es la mínima para la aprobación.

	GE		GC	
Área	Media	Desv. tip	Media	Desv. tip
Cálculo	4,535	0,780	4,700	1,103
Expresiones Algebraicas	4,360	0,605	4,420	0,730
Algebra de Boole	3,555	1,000	3,446	1,503
Resolución de Problemas	1,585	0,450	1,550	0,463

Tabla 78: Resultados del diagnóstico para las cuatro áreas

Como se observa en la tabla 77, tanto los grupos experimentales como control muestran resultados equivalentes en medias, solo existen variaciones un poco más significativas para el caso del área de Cálculo, lo que implica que los grupos pueden tener resultados más o menos agrupados en torno a la media del grupo.

Estas misma tabla muestra que para las áreas de Cálculo y Expresiones Algebraicas los grupos alcanzan valores que son poco más que los mínimos para la aprobación, y que para el área de Algebra de Boole los resultados están levemente por debajo del nivel de aprobación, sin embargo para el área de Resolución de Problemas el nivel de logro para ambos grupos es similar, y alcanzando valores muy por debajo del mínimo de aprobación.

Los resultados para el área de Algebra de Boole, muestran en general valores más altos de sus desviaciones típicas, lo que podríamos interpretar como la existencia de mayores grados de heterogeneidad de parte de los estudiantes, en el manejo de los conocimientos de esta área.

Por su parte, los resultados del área de Resolución de Problemas, son coincidentes con los diagnósticos que han mostrado las investigaciones referidas a las capacidades de resolución de problemas que poseen los estudiantes que ingresan a la educación superior, y son coincidentes también, con las dificultades que se han señalado como una de las causas importantes del fracaso de los estudiantes en los primeros cursos de programación de computadores.

Nuestra experiencia con la asignatura de FP, coincide también con estas evidencias, en el sentido de que hemos constatado que el manejo de las capacidades generales para resolver problemas de orden matemático, influye de manera importante en el desempeño de los estudiantes en la asignatura.

Con el propósito realizar una estimación respecto de cuál es el nivel de explicación que podemos deducir a partir de los resultados de esta evaluación diagnóstica, hemos realizado un análisis factorial respecto de ellos, incluyendo tanto al grupo experimental como al grupo control.

Para estos efectos, hemos determinado excluir el ítem 4.7 en el aspecto de procedimiento, ya que todos sus resultados tienen puntuaciones 1, con lo cual esta medición no presenta variabilidad.

Considerando esto, la tabla de Varianza total explicada muestra que para 21 componentes se alcanza un 79,049% de explicación de los resultados.

Por otra parte, con el objeto de identificar conglomerados jerárquicos (clústeres) que agrupen los ítems de la evaluación diagnostica a partir de los resultados de la misma, hemos construido déndogramas que nos permitan identificar organizaciones de similaridad entre los ítems que hemos evaluados, y partir de esto interpretar las organizaciones jerárquicas relacionándolas con los hallazgos de las investigaciones, y con nuestra propia experiencia en el desarrollo de la asignatura.

En el déndongrama se aprecian 12 niveles de clústeres, los guales se agrupan de la siguiente forma, desde el nivel inferior hacia el nivel superior.

En el nivel de mayor similaridad, que denominaremos nivel 1 se encuentran 3 clústeres. En el primero de ellos se agrupan a los ítems 4.2; 4.6; 4.7, del ámbito manejo conceptual y los ítems 4.1; 4.2; 4.3; 4.5; 4.6; 4.7; del ámbito manejo procedimental. Estos ítems están referidos a la evaluación de la categoría Resolución de problemas, y con ellos pretendemos medir las capacidades generales de resolución de problemas de los estudiantes, en los dos ámbitos señalados.

En los ítems de esta categoría se presentan los enunciados de 7 problemas, que se diseñaron cuidando que en su solución se impliquen conocimientos declarativos y

procedimentales que se encuentren dentro del ámbito de conocimientos que los estudiantes deberían manejar, ya que corresponden a conocimientos que forman parte de los contenidos que se tratan en los cursos de enseñanza media, que son previos a la enseñanza universitaria.

En estos términos, nuestro propósito fundamental es medir las capacidades de los estudiantes para identificar, seleccionar, y organizar los conocimientos que se encuentran implicados en la resolución de cada problema, entendidas estas como capacidades generales.

Respecto de los resultados del diagnóstico, para el ítem 4.2 se tiene un 83.3% de estudiantes con puntuación cero, para el ítem 4.6 un 86.8%, y para el ítem 4.7 un 84.0%, todos ellos en el ámbito conceptual.

En tanto para el ítem 4.1 se tiene un 79.9% de estudiantes con puntuación cero, para el ítem 4.2 un 99.3%, para el ítem 4.3 un 95.8%, para el ítem 4.5 un 87.5%, para el ítem 4.6 un 93.8%, y para el ítem 4.7 un 100%, todos ellos del ámbito del manejo procedimental.

En varios informes de investigaciones referidas al aprendizaje de la programación de computadores se hace mención de que si bien el disponer de las capacidades generales para la resolución de problemas, podría considerarse a priori como gravitante para el éxito del aprendizaje de la programación de computadores, los resultados indican que en la práctica esto no es determinante. Sin embargo estos mismos resultados han mostrado que las capacidades matemáticas generales si son determinantes para el éxito académico.

Nuestra experiencia docente es concordante con estos hallazgos. Creemos que la razón de esto es que si bien en los cursos de FP se trata con la metodología de la resolución de problemas y con la aplicación de la misma, los tipos de problemas son bastante específicos, ya que son problemas de procesamiento de datos y donde además la solución final se expresa como un programa computacional. Respecto de esto, complementariamente podemos comentar, que hemos tenido algunos estudiantes que aun viniendo de una educación previa de tipo humanista, consiguen aprobar la asignatura, y de forma inversa, el que un estudiante provenga de una educación de tipo científica, no asegura que vaya a probar la asignatura.

En el segundo clúster de este nivel se agrupa el ítem 2.10 tanto en el ámbito del manejo conceptual como procedimental. Este es un ítem referido al manejo de Expresiones algebraicas, para el cual se pide al estudiante describir la secuencia de acciones y operaciones que se deben realizar para encontrar las raíces que son solución de una ecuación de segundo grado, con lo cual, se está midiendo la

capacidad para describir el algoritmo de solución, que suponemos conocido por los estudiantes.

En términos de los resultados del diagnóstico para este ítem, para el manejo conceptual se obtiene un 68.1% de estudiantes con puntuación cero, y para el manejo procedimental del mismo ítem, se obtiene un 75.7% de estudiantes con puntuación cero.

Este es un resultado que nos parece relevante destacar. En los ítems 2.9.1, 2.9.2, 2.9.3. 2.9.4, y 2.9.5 presentamos a los estudiantes 5 expresiones algebraicas de distinta complejidad, y que corresponden formulaciones de ecuaciones de segundo grado, para las cuales les pedimos que encuentren sus soluciones. Conforme a los resultados cuantitativos, se verifica que los estudiantes en un número cercano al 60% logran realizar con éxito la tarea, lo implica que en términos generales, ellos además de conocer los procedimientos algebraicos que se requieren para manejar este tipo de expresiones matemáticas, pueden utilizar con éxito la fórmula para encontrar las dos soluciones, sin embargo existe un alto porcentaje de ellos que no son capaces de describir correctamente la secuencia de pasos y operaciones que se debe realizar para encontrar las soluciones de la ecuación.

En el tercer clúster se agrupa al ítem 2.9.5 tanto en el ámbito del manejo conceptual como procedimental. Este es un ítem referido al manejo de Expresiones algebraicas que corresponde a una expresión de segundo grado que contiene términos racionales, para la cual se pide encontrar las raíces que son solución de la expresión, por lo que estamos evaluando tanto el manejo de los conceptos y la capacidad para usar y aplicar el algoritmo de solución que existe para este tipo de expresiones. Para este ítem el diagnostico arrojó que sobre el 60% tuvo éxito en el desarrollo de este ítem, con lo cual podemos concluir que si bien los estudiantes pueden utilizar con cierto éxito un algoritmo que les es conocido, tienen dificultades importantes para describir el funcionamiento de dicho algoritmo.

En síntesis, las jerarquizaciones de estos tres clústeres para el primer nivel de agrupamiento tienen un denominador común en el sentido de que en todos ellos se está evaluando la capacidad de los estudiantes para describir secuencias de acciones y operaciones (algoritmos) para resolver determinadas situaciones, ámbito en el cual se evidencian índices importantes de fracaso, sin embargo, conforme a nuestra experiencia y los reportes de investigaciones en el área, esto no necesariamente puede considerarse como un indicativo a priori del fracaso que podrían experimentar los estudiantes en el curso de FP.

En el segundo nivel de similaridad de los ítems se encuentran 12 clústeres en los cuales se aprecia principalmente agrupaciones de ítems de la segunda categoría, con excepción de tres clústeres. En uno de estos se agrupan los ítems 4.3 y 4.5 en

el ámbito del manejo conceptual, los que como señalado anteriormente, corresponden a ítems de la categoría de Resolución de problemas. Conforme a los resultados del diagnóstico, estos dos ítems presentan un alto porcentaje de puntuación cero (83.3% y 72.9% respectivamente), lo que es consecuente con los resultados señalados anteriormente con respecto a los ítems de esta misma categoría.

En el segundo y tercero de estos clústeres de excepción se agrupan al ítem 1.1 y el ítem 1.2 ambos en los ámbitos del manejo conceptual y procedimental. Estos ítems pertenecen a la categoría de Cálculo, y el primero está referido a un cálculo simple de razón de incremento, en tanto el segundo está referido a un cálculo de porcentaje de carácter simple. Los resultados de la evaluación muestran valores de logro cercanos al 60% en el ámbito conceptual, y cercanos a 50% en el ámbito procedimental.

Para aprender respecto de la resolución de problemas de procesamiento de datos, los conocimientos que evaluamos de la categoría de cálculo, constituyen una fuente de conocimientos fundamentales, ya que en rigor, en el primer curso de programación tratamos principalmente con problemas matemáticos que implican cálculos.

Sin embargo, aunque los resultados de la evaluación del resto de los ítems de esta categoría muestran niveles de logro equivalentes a los ítems 1.1 y 1.2, en los reportes de investigación no se considera que esta sea una causa del fracaso de los estudiantes. Conforme a nuestra experiencia tampoco lo vemos como una causa importante de fracaso, y podría contribuir a la mejora en este aspecto el hecho de que junto con la asignatura de FP, los estudiantes generalmente están cursando la asignatura de Introducción a la ingeniería, que en una parte importante trata con cálculos matemáticos.

En los nueve clústeres restantes del segundo nivel, se agrupan la mayoría de los ítems pertenecientes a la categoría de expresiones algebraicas, tanto en el ámbito del manejo conceptual como procedimental. En esta categoría hemos incluido ejercicios de reducción de términos algebraicos comunes, determinación del valor de funciones para un valor dado de la variable, de encontrar incógnitas en ecuaciones de primer grado, de sistemas de dos ecuaciones de primer grado y de ecuaciones de segundo grado, a partir de expresiones de distinta complejidad algebraica.

Los resultados cuantitativos del diagnóstico para los estos tipos de ítems muestra un desempeño que fluctúa entre el 50% y el 65% de éxito en las respuestas correctas.

Hemos incluido también en esta categoría enunciados de situaciones que para ser resueltas requieren la formulación de una ecuación matemática y de su correspondiente solución, en cuyo caso los resultados del desempeño bajan para

situarse entre el 25% y el 30%, lo que indica que si bien, pueden aplicar con mediano éxito los procedimientos para resolver ecuaciones de primer y segundo grado, tienen dificultades cuando se les pide que a partir de un enunciado construyan y resuelvan ecuaciones. La razón de esto podría estar en la dificultad para modelar y resolver matemáticamente el enunciado de una situación problema.

Por último, también hemos incluido en esta categoría dos ítems en los cuales pedimos a los estudiantes que describan las secuencias de acciones y operaciones necesarias para encontrar los valores de las incógnitas en un sistema de dos ecuaciones (ítem 2.6), y para encontrar las raíces que son solución de una ecuación de segundo grado (ítem 2.10).

Para el ítem 2.7, aproximadamente un 40% de los estudiantes obtuvo puntuación cero, y para el ítem 2.10 un 75% obtuvo puntuación cero. Teniendo en cuenta que las puntuaciones que alcanzan los estudiantes cuando se les pide resolver sistemas de ecuaciones y ecuaciones de segundo grado, son bastante mejores, creemos que estos resultados reflejan la dificultad que tienen los estudiantes para expresar procedimientos que conocen y aplican, como una secuencia de acciones y operaciones.

En síntesis, este nivel, con excepción del clúster 4 que agrupa a los ítems 4.3 y 4.5 de la categoría resolución problemas, se jerarquizan la mayoría de los ítems de la categoría de manejo de expresiones algebraicas.

Como ya hemos señalado los ítems de la cuarta categoría son los que presentan el mayor índice de fracaso en la evaluación diagnostica de los estudiantes, y que son los ítems donde les pedimos que describan la secuencia de acciones y operaciones que se requieren realizar para encontrar un cierto resultado numérico a partir del enunciado de una determinada situación problema.

En tanto para los ítems de la categoría de manejo de expresiones algebraicas se observan valores de rendimiento que se encuentra entre el 50% y 60% con excepción de aquellos ítems en que les pedimos que describan ciertos procesos de para resolver las situaciones que planteamos (ítem 2.6 y 2.10), en cuyos casos el rendimiento disminuye de manera importante.

En el tercer nivel del déndograma se agrupan 6 clústeres. Uno de ellos lo conforman los ítems 2.10 y 4.1 en el ámbito conceptual, que aunque ambos pertenecen a categorías diferente, para cada uno de ellos pedimos a los estudiantes describir secuencias de acciones y operaciones.

En un segundo clúster se agrupa al clúster 3 y el ítem 4.4 en el ámbito conceptual y procedimental, que como ya sabemos es de la categoría de resolución de problemas,

para los cuales pedimos a los estudiantes que describan la secuencia de acciones y operaciones para encontrar el valor de la incógnita en una ecuación de primer grado de tres términos

El tercer clúster de este nivel grupa a los ítems 2.3, 2.4, y 2.5.3 todos en el ámbito procedimental, los que como se recordará son ítems de la categoría expresiones algebraicas. Para el caso del primer ítem presentamos a los estudiantes una relación de diferencias entre dos números y les pedimos determinar estos dos números. El ítem 2.4 es un problema similar pero trata con la relación entre tres números. En el ítem 2.5.3 presentamos a los estudiantes un sistema de dos ecuaciones de primer grado con dos incógnitas y les pedimos determinar los valores de esas dos incógnitas.

En el cuarto se agrupa al clúster 9 y el ítem 1.3 en el ámbito conceptual. En este ítem de cálculo pedimos a los estudiantes calcular un porcentaje a partir de un cierto número.

El quinto clúster agrupa al clúster 10 y a los ítems 2.8.1 y 2.8.2 en el ámbito procedimental. Para estos dos ítems pedimos a los estudiantes calcular ciertas expresiones dado el valor de sus variables.

En el sexto clúster se agrupan a los clústeres 11, 12, 13, 14, y 15 y los ítems 2.1.1 en el ámbito conceptual y el ítem 2.2.1 en los ámbitos conceptual y procedimental. Estos ítems pertenecen a la categoría de expresiones algebraicas. En el primero de ellos pedimos a los estudiantes reducir términos semejantes en una expresión algebraica de 6 términos, en tanto en el segundo de ellos, pedimos determinar el valor de la incógnita para una ecuación de primer grado.

En síntesis, con excepción de uno de los ítems del primer clúster y de los únicos ítems del segundo y cuarto clúster, todos los ítems del resto de los clústeres de este tercer nivel, agrupan ítems de la categoría de expresiones algebraicas. En algunos casos referidos fundamentalmente al manejo de expresiones de este tipo, y en otros referidos a la resolución de ecuaciones de primer grado ya sea como ecuaciones individuales o como sistemas de dos ecuaciones con dos incógnitas.

El déndograma agrupa en los niveles noveno, décimo y undécimo a los ítems de la tercera categoría la que se relaciona con el tema algebra de Boole, con excepción del ítem 3.1.5 de esta misma categoría que ha sido agrupado en el sexto nivel del déndograma.

El álgebra de Boole en el ámbito de la lógica de primer orden, constituyen el recurso formal para describir las condicionantes bajo las cuales debe operar un algoritmo o

parte de un algoritmo, y consecuentemente el programa computacional, para responder a la generalización de las soluciones.

Los ítems 3.1.1 y 3.1.2 se agrupan en un único clúster del noveno nivel, y la evaluación de ellos muestra que para los dos ítems se alcanza un 27.1% de estudiantes con puntuación cero. En estos ítems presentamos una expresión lógica conteniendo un solo operador y dos operandos. En el primer caso se encuentra el operador or y en el segundo el operador lógico and.

Por su parte el ítem 3.1.4 se encuentra agrupado en el décimo nivel junto a los clústeres agrupados en el octavo nivel. En este ítem pedimos a los estudiantes determinar si una expresión lógica que contiene agrupación de operadores lógicos, es verdadera o falsa. Un 35.4% de los estudiantes tuvo puntuación cero en este ítem.

El ítem 3.1.3 se encuentra agrupado en el nivel undécimo con un 21.5% de puntuación cero.

Por otra parte, al obtener el déndograma conglomerando los casos, se observa que no existe agrupación de estos en base a los grupos de pertenencia de los estudiantes, lo cual nos indica que en términos de la evaluación diagnostica, no existen diferencia entre los grupos utilizados para el estudio.

6.3. Resultados referidos al logro de las Competencias

A continuación, presentamos los resultados de la evaluación realizada, respecto de los niveles de dominio alcanzado sobre el conjunto de competencias que se definieron.

Como se recordará, estas competencias fueron definidas en los siguientes ámbitos:

- Conceptual; referidas a los conocimientos de todos aquellos elementos que conforman la base teórico-formal de la disciplina. Se clasifican en esta categoría todos los conocimientos disciplinares de tipo declarativos, como por ejemplo el concepto de algoritmo, las construcciones de la lógica de primer orden, el concepto de lenguaje de programación entre otros.
- Procedimental; referido al manejo de los procedimientos que definen la forma de hacer algo. Se sitúa en este ámbito el manejo de los pasos de la metodología de la programación.
- Habilidades y destrezas; referidas a la capacidad para relacionar los elementos conceptuales y procedimentales. Se sitúan en este ámbito las habilidades y destrezas para relacionar por ejemplo, el concepto de algoritmo con sus formas de representación y con la propia metodología de la programación.

- Estratégicas; referidas a la capacidad para integrar los distintos tipos de conocimientos con el propósito de diseñar e implementar soluciones a los problemas de procesamiento de datos. Su manifestación es prueba de que el estudiante ha desarrollado la capacidad para manejar y aplicar estrategias que le permiten construir las soluciones.

Como indicamos, estas competencias se organizaron en tres categorías generales, y dentro de ellas se definieron sub-categorías de carácter más específico, esto con el propósito de valorar de manera más precisa cada competencia.

Por tanto, serán las competencias definidas en las sub-categorías, las que utilizaremos como indicadores para medir los niveles de logro que han alcanzado los estudiantes. Las categorías y sub-categorías se organizan como:

1. Capacidad para comprender y explicar los fundamentos teórico-formales de la programación de computadores.
   1.1. Comprender y explicar las bases teóricas de la programación
   1.2. Comprender y explicar el modelo de programación de un computador
2. Capacidad para aplicar la metodología de la programación a la resolución de problemas de procesamiento de datos.
   2.1. Comprender los propósitos de cada etapa de la metodología de la programación
   2.2. Aplicar los procedimientos para desarrollar las etapas de la metodología
   2.3. Utilizar las distintas formas de representación algorítmica
   2.4. Diseñar algoritmos de solución
3. Capacidad para construir e implementar programas computacionales.
   3.1. Declarar y utilizar correctamente los tipos de datos simples
   3.2. Utilizar correctamente la jerarquía de operadores
   3.3. Utilizar correctamente las estructuras de programación
   3.4. Declarar y utilizar correctamente estructuras de datos del tipo arreglos
   3.5. Utilizar algoritmos de ordenamiento sobre arreglos
   3.6. Utilizar algoritmos de búsqueda sobre arreglos
   3.7. Utilizar funciones de la biblioteca estándar
   3.8. Implementar funciones
   3.9. Construir e implementar programas a partir de su representación algorítmica

Estas competencias fueron evaluadas en dos instancias. La primera corresponde a mediciones llevadas a cabo en tres evaluaciones acumulativas durante el semestre de desarrollo de la asignatura, y la segunda corresponde a la medición desarrollada en el post-test.

Para la primera instancia, y conforme a la cronología de desarrollo de la asignatura, la distribución de competencias medidas en cada prueba corresponde a la siguiente:

- Primera prueba se miden las competencias: 1.1, 1.2, 2.1, 2.2, 2.3, y 2.4
- Segunda prueba se mide las competencias: 2.1, 2.2, 2.3, 2.4, 3.1, 3.2, 3.3, y 3.9
- Tercera prueba se miden las competencias: 2.4, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, y 3.9

En tanto en la evaluación post-test, que se aplicó una vez terminado el desarrollo de la asignatura, se han medido las competencias 2.4, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, y 3.9 en su estado de logro final.

La escala de medición para todas las evaluaciones esta basa en la escala de calificaciones usada tanto en la Universidad de Playa Ancha como en la Universidad de Valparaíso, la que corresponde a notas en el rango de 1.0 a 7.0, siendo 4.0 la nota mínima de aprobación.

La fiabilidad de estas mediciones se estimó mediante el Chi-cuadrado de Cochran, calculado para el conjunto de los grupos experimental y control, arrojando los siguientes resultados sobre 144 casos:

- Primera prueba: alfa=0,986
- Segunda prueba: alfa=0,993
- Tercera prueba: alfa=0,982
- Post-test: alfa=0,980

A continuación presentamos las tablas de los estadísticos descriptivos para cada evaluación.

Teniendo en consideración que el propósito central de esta investigación es mejorar el aprendizaje de los estudiantes en la asignatura, comentaremos las diferencias que presentan los resultados de los desempeños en las competencias, en las diferentes pruebas aplicadas a los grupos, para las dos intervenciones. Estas diferencias estarán dadas en términos de puntuaciones en la escala de notas que hemos señalado.

Primera Evaluación

GE	N	Mínimo	Máximo	Media	Desv. típ.
p1_c1.1	61	2,5	6,8	4,807	1,0954
p1_c1.2	61	2,3	6,6	4,656	1,0871
p1_c2.1	61	2,0	6,8	4,234	1,1581
p1_c2.2	61	1,4	6,5	3,861	1,1191
p1_c2.3	61	1,0	6,3	3,607	1,1939
p1_c2.4	61	1,0	6,2	3,328	1,2107

Tabla 79: Resultados de la primera evaluación GE

GC	N	Mínimo	Máximo	Media	Desv. típ.
p1_c1.1	83	1,5	6,9	4,690	1,2865
p1_c1.2	83	1,2	6,9	4,577	1,3100
p1_c2.1	83	1,5	6,8	4,312	1,2890
p1_c2.2	83	1,4	6,9	4,239	1,3757
p1_c2.3	83	1,3	6,8	4,179	1,3647
p1_c2.4	83	1,0	6,7	3,728	1,4024

Tabla 80: Resultados de la primera evaluación GC

En la tablas 78 y 79 se muestran los resultados para las competencias evaluadas en la primera prueba, no se aprecian diferencias significativas en las Medias entre cada grupo experimental y sus grupos control.

Sin embargo nos parece importante destacar que:

- Para las competencias de la primera categoría, las que están referidas a la capacidad para comprender y explicar los fundamentos teórico-formales de la programación de computadores, los grupos experimentales presentan valores de media levemente superiores a los grupos control.
- Para el caso de la competencia de la segunda categorías, que están referidas a la capacidad para aplicar la metodología de la programación a la resolución de problemas de procesamiento de datos, sin bien las diferencias entre cada grupo experimental y sus correspondientes grupos control no son significativas, son los grupos control los que obtienen mejor puntuación.
  Estos resultados son consecuentes con las investigaciones referidas a las mediciones iniciales de los efectos del aprendizaje cooperativo, en comparación con los métodos tradicionales de enseñanza, y se justificarían por el hecho de que los estudiantes que intentan aprender de manera cooperativa, requieren de un tiempo de acomodación y también de un tiempo para adquirir las habilidades para aprender cooperativamente, lo que llevado a la práctica en esta asignatura, significa que los estudiantes de los grupos experimentales han debido trabajar sobre un aprendizaje adicional además del aprendizaje de las capacidades de la segunda categoría, que al ser competencias de orden práctico, requieren de mayores tiempos para su ejercitación que aquellas competencias de la primera categoría. Así, los tiempos de ejercitación para los grupos experimentales han sido afectados por la demanda de aprender y acomodarse al aprendizaje cooperativo.
- Sin embargo, nos parece importante destacar que los valores de desviación típica para todas las competencias evaluadas, muestran valores menores, en el grupo experimental que en el grupo control, lo que significa que los resultados para el primer grupo están más cohesionados respecto de la media, que para el caso del grupo control.

Se debe recordar que las competencias de la primera categoría están referidas a un tipo de conocimientos más bien de orden teórico-conceptual, en tanto las competencias de la segunda categoría son de orden práctico-aplicativo.

Estos mismos elementos de análisis se evidencian en el siguiente gráfico, donde mostramos una comparación entre los porcentajes de niveles de logro alcanzados en cada competencia por cada grupo.

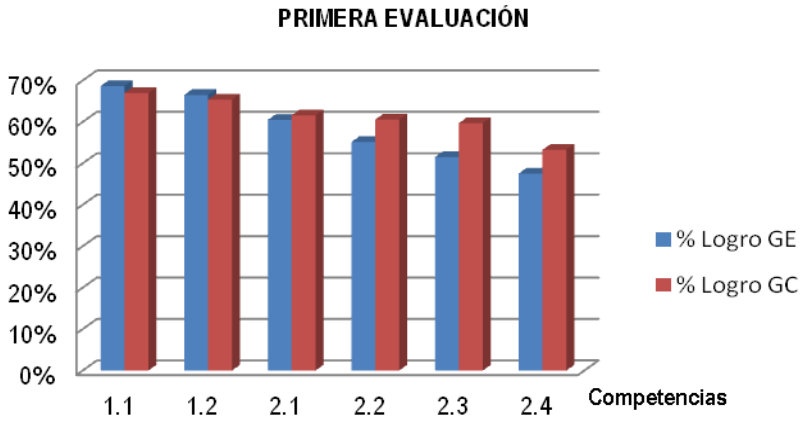

Gráfico 1: Resultados de la Primera evaluación

Segunda Evaluación

GE	N	Mínimo	Máximo	Media	Desv. típ.
p2_c2.1	61	3,7	6,9	5,236	0,9089
p2_c2.2	61	3,4	6,9	5,025	0,9690
p2_c2.3	61	3,0	6,7	4,772	0,9647
p2_c2.4	61	1,7	6,5	4,020	1,2128
p2_c3.1	61	3,0	6,8	4,687	1,0251
p2_c3.2	61	2,0	6,5	4,423	1,1727
p2_c3.3	61	2,0	6,5	4,289	1,1489
p2_c3.9	61	2,0	6,6	4,202	1,2095

Tabla 81: Resultados de la Segunda evaluación para el GE

GC	N	Mínimo	Máximo	Media	Desv. típ.
p2_c2.1	83	1,0	6,7	3,909	1,3853
p2_c2.2	83	1,0	6,7	3,834	1,4227
p2_c2.3	83	1,0	6,5	3,685	1,4368
p2_c2.4	83	1,0	6,0	2,759	1,3621
p2_c3.1	83	1,0	6,6	3,554	1,3778
p2_c3.2	83	1,0	6,3	3,474	1,4467
p2_c3.3	83	1,0	6,1	3,326	1,3789
p2_c3.9	83	1,0	6,0	2,969	1,4143

Tabla 82: Resultados de la Segunda evaluación para el GC

De estos resultados de la segunda evaluación se puede señalar que:

- En todas las competencias evaluadas, el grupo control evidencia mejores niveles de logro que el grupo control, apreciándose mayores diferencias a favor del grupo experimental para las competencias 2.4, 3.3 y 3.9.
- En las tablas se muestra que el grupo control en las competencias de la segunda categoría (2.1, 2.2, 2.3, y 2.4) que en la medición de la primera evaluación obtuvo mejor evaluación que el grupo experimental, en esta segunda evaluación obtiene resultados de menor nivel que el grupo experimental, lo cual daría cuenta de los efectos que están teniendo las actividades cooperativas sobre las mejoras en el grupo experimental.
- Se aprecia también que para el grupo control, la calificación mínima de cada una de las competencias está en el valor de la nota mínima, lo que evidencia que existieron estudiantes de ese grupo que no alcanzaron a sobrepasar la nota mínima.
- Consecuente con lo anterior, se aprecian diferencias significativas en las desviaciones típicas de cada competencia evaluada, de lo que se observa, el grupo experimental muestra desviaciones típicas menores en todas las competencias respecto del grupo control, destacando las difererencias en este indicador, en las competencias 2.1, 2.2, y 2.3.
- De esto podemos deducir que en esta segunda evaluación se aprecian beneficios como efectos de las actividades cooperativas, que han sido incorporadas al desarrollo de la asignatura del grupo experimental.

Creemos importante destacar también, que estas diferencias en el logro de los desempeños, se consiguen en competencias que demandan el manejo de los aspectos aplicativos de conocimientos de FP, cuestión que constituye un objetivo importante de esta asignatura.

De estos resultados se desprende que mientras el grupo experimental mejora su evaluación en las competencias de la segunda categoría, los grupos control bajan en sus niveles de logro. Para nosotros estos resultados son consecuencia de que si bien se han evaluado las mismas competencias que en la primera prueba, en esta segunda prueba, los estudiantes han sido puestos en situaciones problemas de mayor complejidad, por lo que las mejoras en su logro para los grupos experimentales las interpretamos como el efecto de la intervención con las actividades de aprendizaje cooperativo, para las cuales además, ya tienen un mayor grado de adaptación, lo que les permite sacar un mejor provecho de las mismas.

Para el caso de las competencias de la tercera categoría las que están en general referidas a la capacidad para construir e implementar programas computacionales, estas también muestran resultados de logro superiores para los grupos experimentales que para los grupos control. La competencia 3.1 está referida a la

capacidad para declarar y utilizar correctamente los tipos de datos simples, la 3.2 a la capacidad para utilizar correctamente la jerarquía de operadores, la 3.3 a la capacidad para utilizar correctamente las estructuras de programación, y la 3,9 a la capacidad para construir e implementar programas a partir de su representación algorítmica. Estas competencias han sido evaluadas también en la tercera prueba en el contexto de la resolución de problemas de mayor complejidad y del uso de recursos del lenguaje de programación y técnicas de programación, también más complejos.

Por otra parte, los valores de las desviaciones típicas de estas tablas muestran que para las dos intervenciones, los estudiantes de los grupos experimentales están más agrupados en torno a la media de su grupo, lo que se hace más evidente para la segunda intervención.

En el gráfico 2 en la página siguiente, se muestran los porcentajes de logro parciales alcanzados por los grupos experimentales y control, en las competencias que fueron medidas en la segunda prueba.

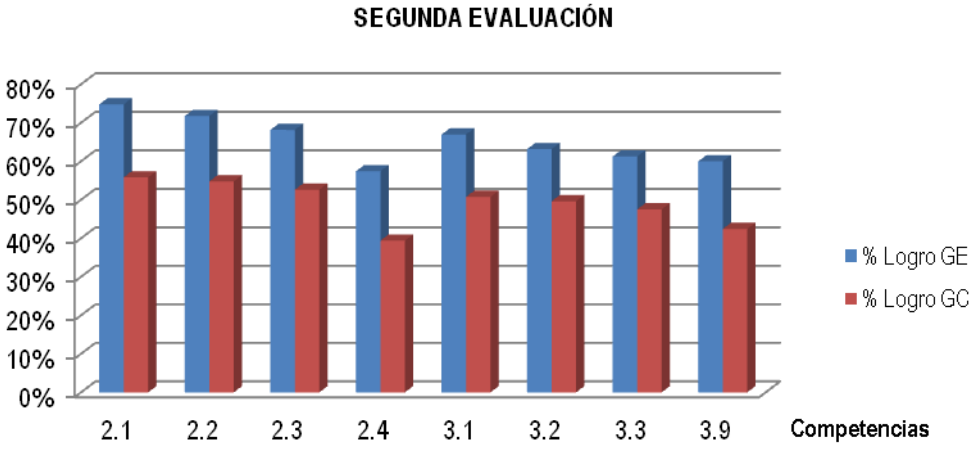

Gráfico 2: Comparativo GE y GC para la segunda evaluación

Tercera Evaluación

GE	N	Mínimo	Máximo	Media	Desv. típ.
p3_c2.4	61	2,5	6,6	4,461	1,1578
p3_c3.1	61	3,0	6,9	5,298	0,9537
p3_c3.2	61	2,7	6,7	4,836	1,1042
p3_c3.3	61	2,3	6,8	4,690	1,1681
p3_c3.4	61	2,4	6,9	4,848	1,2084
p3_c3.5	61	2,9	6,7	4,716	1,0999
p3_c3.6	61	2,8	6,9	5,015	1,1117
p3_c3.7	61	2,8	6,9	5,077	1,0665
p3_c3.8	61	3,0	6,8	4,870	1,0760
p3_c3.9	61	2,8	6,9	4,757	1,0475

Tabla 83: Resultados Tercera evaluación GE

GC	N	Mínimo	Máximo	Media	Desv. típ.
p3_c2.4	83	1,0	7,0	2,890	1,4298
p3_c3.1	83	1,0	7,0	3,724	1,5900
p3_c3.2	83	1,0	7,0	3,745	1,5947
p3_c3.3	83	1,0	7,0	3,734	1,5372
p3_c3.4	83	1,0	7,0	3,608	1,5106
p3_c3.5	83	1,0	7,0	3,375	1,4972
p3_c3.6	83	1,0	6,6	3,512	1,3126
p3_c3.7	83	1,0	6,7	3,514	1,3358
p3_c3.8	83	1,0	6,7	3,248	1,2830
p3_c3.9	83	1,0	7,0	3,084	1,4607

Tabla 84: Resultados Tercera evaluación GC

Como muestran los resultados de las tablas 82 y 83, para esta tercera prueba, en cada una de las competencias evaluadas, el grupo experimental muestra resultados significativamente mejores que el grupo control. Las medias para cada competencia en el grupo experimental son significativamente mayores que en grupo control, y además el grupo experimental no registró ningún mínimo inferior a 2,3 en los logros de las competencias medidas en esta evaluación, en tanto el grupo control muestra mínimos de 1,0 en todas las competencias evaluadas.

Además de lo anterior, nuevamente se aprecia que las desviaciones típicas para el grupo experimental son menores que en el grupo control, siendo en algunos casos, significativamente menores que este último, lo que da cuenta que el grupo experimental presenta una mayor aglomeración en los logros de las competencias, en torno a la media de su grupo, respecto de lo que ocurre con el grupo control.

Estos resultados muestran que aun cuando la competencia 2.4 muestra un valor superior para el grupo experimental sus medias son menores que las obtenidas en la segunda prueba, cuestión que también se evidencia para el grupo control. Esta disminución se explica por la complejidad creciente de los problemas que debieron resolver en la tercera evaluación.

Por su parte para las competencias de la tercera categoría medidas en esta prueba se aprecian diferencias significativas en sus medias entre el grupo experimental y el grupo control. Así por ejemplo, la menor diferencia en las medias de los dos grupos se tiene en la competencia 3.3 que corresponde a 0.955 a favor del grupo experimental. En tanto, la máxima diferencia se tiene en la competencia 3.9 y corresponde a 1.672 a favor del grupo experimental. Particularmente el logro de la competencia 3.9 constituye también un objetivo altamente relevante de la asignatura ya que está referido a la capacidad para construir e implementar programas a partir de su representación algorítmica.

En el gráfico 3 de la página siguiente se muestran los porcentajes de logro parciales alcanzados por los grupos experimental y control, en las competencias que fueron medidas en la tercera prueba.

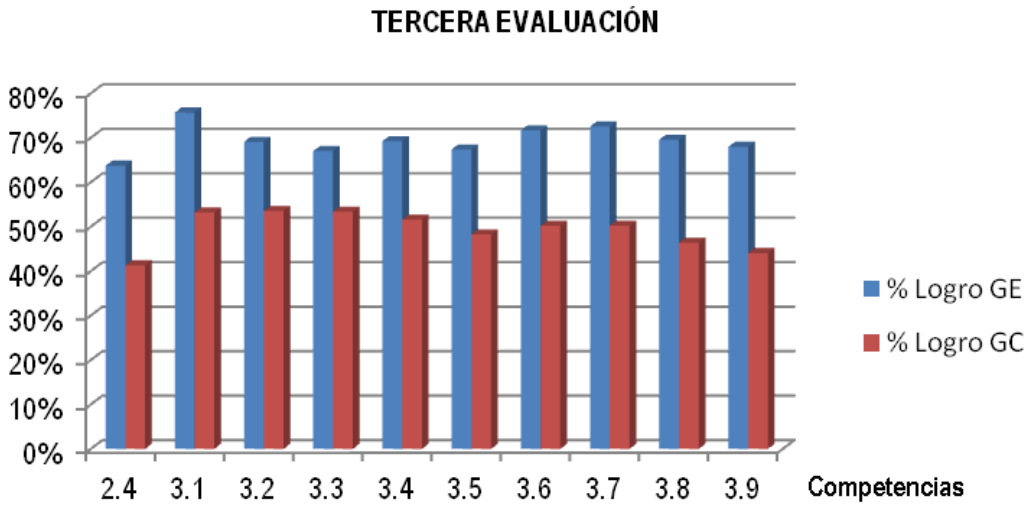

Gráfico 3: Comparativa entre el GE y GC para la Tercera evaluación

Post-test

El instrumento Post-test ha sido diseñado teniendo en consideración la medición de las competencias que reflejan la globalidad de los objetivos de la asignatura, conforme a lo se han considerado un total de 12 competencias cuyos resultados de logro para el grupo experimental y control respectivamente, mostramos en las siguientes tablas.

GE	N	Mínimo	Máximo	Media	Desv. típ.
pt.2.2	61	3,2	6,7	4,559	0,9604
pt.2.3	61	2,9	6,5	4,306	0,9724
pt.2.4	61	2,4	6,3	4,265	1,0563
pt.3.1	61	3,2	6,8	5,054	0,9272
pt.3.2	61	3,1	6,7	5,002	0,9616
pt.3.3	61	2,3	6,6	4,729	1,1034
pt.3.4	61	2,4	6,9	4,848	1,2084
pt.3.5	61	2,9	6,7	4,716	1,0999
pt.3.6	61	2,8	6,9	5,015	1,1117
pt.3.7	61	2,8	6,9	5,077	1,0665
pt.3.8	61	3,0	6,8	4,870	1,0760
pt.3.9	61	2,8	6,6	4,614	1,0168

Tabla 85: Resultados Post-test GE

GC	N	Mínimo	Máximo	Media	Desv. típ.
pt.2.2	83	1,2	6,3	3,996	1,1720
pt.2.3	83	1,1	6,5	3,883	1,1800
pt.2.4	83	1,0	5,7	2,928	1,0201
pt.3.1	83	1,2	6,8	3,656	1,3450
pt.3.2	83	1,0	5,9	3,644	1,1724
pt.3.3	83	1,1	6,1	3,552	1,1571
pt.3.4	83	1,0	7,0	3,608	1,5106
pt.3.5	83	1,0	7,0	3,375	1,4972
pt.3.6	83	1,0	6,6	3,512	1,3126
pt.3.7	83	1,0	6,7	3,514	1,3358
pt.3.8	83	1,0	6,7	3,248	1,2830
pt.3.9	83	1,0	5,5	3,033	1,1026

Tabla 86: Resultados Post-test GC

De los resultados de esta medición se observa que se mantienen las diferencias en los niveles de logro entre el grupo experimental y control, respecto de los cuales nos parece importante comentar que:

- La competencia 2.4 que está referida a la capacidad para diseñar algoritmos de solución, y que ha sido evaluada en cada una de las pruebas, para esta medición post-test muestra una media de logro que es 1.2763 superior que en el grupo control. Esto nos parece un resultado relevante ya que como se recordará, esta es una capacidad altamente relevante de lograr para la asignatura. Como se recordará situaciones de diseño de algoritmos de mayor complejidad, incrementa su media en el grupo experimental y la disminuye en el grupo control.
- Nos parece también importante destacar que en las tablas se observa una diferencia importante en el nivel de logro de la competencia 3.3 la cual está referida a la capacidad para utilizar correctamente las estructuras de programación, objetivo que también es relevante de conseguir en una asignatura de este tipo.
- En la misma línea destacamos también los resultados de la competencia 3.9 que está referida a la capacidad para construir e implementar programas a partir de su representación algorítmica, la que para el grupo experimental a muestra un valor en su media de que es 1.5809 mayor que en el grupo control, y que al igual que la competencia 2.4 está también constituye un objetivo importante a lograr en la asignatura.
- En relación con los niveles de logro en todas las competencias de la tercera categoría que están referidas a la capacidad para construir e implementar programas computacionales, el grupo experimental muestra valores significativamente superiores que el grupo control. Así en todas las competencias de esta categoría, el grupo experimental muestra diferencias

que van desde los 1.177 (competencia 3.3) hasta diferencias de 1.622 (competencia 3.8).

Otro aspecto que destacamos es que a diferencia del grupo experimental, el grupo control en varias de las competencias medidas (2.4, 3.4, 3.5, 3.6, 3.7, 3.8, y 3.9) alcanzó la valoración mínima de la escala de calificación (1,0). Para el caso del grupo experimental la competencia que alcanzó la mínima valoración fue la 3.3 con valor 2.3

Al observar los valores de las desviaciones típicas para ambos grupos se aprecia que para todas las competencias medidas en este post-test, los estudiantes del grupo experimental están más agrupados en torno a la media del grupo que los del grupo control.

En el siguiente gráfico se muestran una comparativa de los porcentajes de logro alcanzados por los grupos experimental y control, en las competencias que fueron medidas en el post -test.

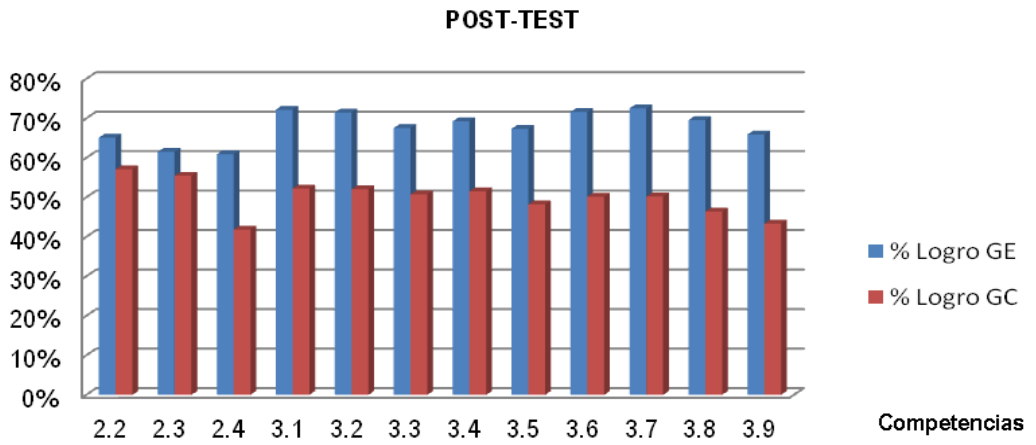

Gráfico 4: Comparativa entre GE y GC para los resultados del Post-test

Así, estos resultados reflejan diferencias significativas en el logro de aquellas competencias que, conforme a lo que hemos analizado y discutido en los apartados anteriores, se consideran fundamentales y esenciales de alcanzar por los estudiantes en un primer curso de programación como es el caso de la asignatura de FP.

En este contexto, reiteramos que una de estas competencias corresponde a la 2.4 la que está referida a la capacidad para diseñar algoritmos de solución para los problemas de procesamiento de datos.

Como ha sido planteado, una de las capacidades que cualquier futuro Ingeniero en Informática debe poseer, es la habilidad para diseñar soluciones a los problemas de procesamiento de datos, como se ha señalado anteriormente, un profesional de este

tipo es por naturaleza un diseñador de soluciones a problemas de procesamiento de información. Si bien esta es una competencia que no completa su desarrollo en esta asignatura, con esta se han de desarrollar las bases metodológicas y estratégicas que sirvan al estudiantes para abordar el aprendizaje de soluciones más complejas en los cursos superiores, como por ejemplo en los cursos de modelamiento de bases de datos, de diseño de sistemas de información, y en los cursos de inteligencia artificial.

En razón de esto, consideramos que esta mejora en el logro de esta capacidad es importante no sólo para la propia asignatura de FP, sino que para varias otras asignaturas que requieren de ella.

En estos resultados también se aprecian diferencias importantes en los niveles de logros alcanzados por los grupos experimentales en comparación con los grupos control, en todas las competencias de la tercera categoría, las cuales están referidas a las capacidades para construir e implementar programas computacionales.

Desde la perspectiva del conocimiento práctico de la asignatura, esta son un conjunto de competencias complementarias a las competencias de la segunda categoría, ya que estas se relacionan con las capacidades para usar los recursos de un lenguaje de programación para implementar los programas computacionales de solución.

Como hemos planteado, dentro de la malla curricular de estas carreras se encuentran varias asignaturas relacionadas de manera directa con la programación, ya sea como asignaturas donde se trata con recursos más avanzados de los lenguajes, asignaturas que tratan con otros modelos de programación, y asignaturas que tratan con otros lenguajes de programación.

Además de esto, varias otras asignaturas del plan curricular, como lo hemos señalado, requieren del manejo de estas capacidades, ya sea para entender ciertos algoritmos programados, como también para implementar los programas para ciertos algoritmos clásicos de procesamiento de datos.

En un ámbito más global, como ha sido planteado en las recomendaciones curriculares a las cuales hemos hecho referencia en apartados anteriores, la habilidad para programar computadores, es considerada como una habilidad ineludible con la cual debe contar cualquier titulado de una carrera del área de las Ciencias de la Computación.

Con el propósito de complementar el análisis de los resultados de la evaluación post-test de ambos grupos, hemos realizado un análisis factorial respecto de los resultados alcanzados por el grupo experimental y el grupo control por separado.

Así, al aplicar el procedimiento de análisis para reducción de datos mediante factorial usando SPSS se obtiene que para el grupo experimental, con una componente se acumula un porcentaje de varianza explicada de 89.505%.

Para el mismo análisis en el grupo control se obtiene que con dos componentes se acumula un porcentaje de varianza explicada de 90.107%.

Teniendo en cuenta que los conocimientos con que trata esta asignatura, deben ser aprendidos de manera integral y coherente, hemos aplicado el procedimiento de análisis para la clasificación de datos, mediante conglomerados jerárquicos usando SPSS, con el objeto de visualizar las organización jerárquica que existen entre las competencias para los resultados del post-test. Las organizaciones jerárquicas nos permitirán conocer además las relaciones de similaridad que se dan entre estas competencias, con lo cual podremos analizar la integridad y coherencia en el logro de las competencias.

En esta perspectiva, también creemos que es interesante detectar si existen diferencias en las organizaciones jerárquicas que resultan de las relaciones de similaridad entre las competencias, entre el grupo experimental y control.

Para visualizar las relaciones de similaridad y las organizaciones jerárquicas de las competencias, se han obtenido los déndogramas por separado del grupo experimental y del grupo control. En el siguiente gráfico se muestra el correspondiente al grupo experimental.

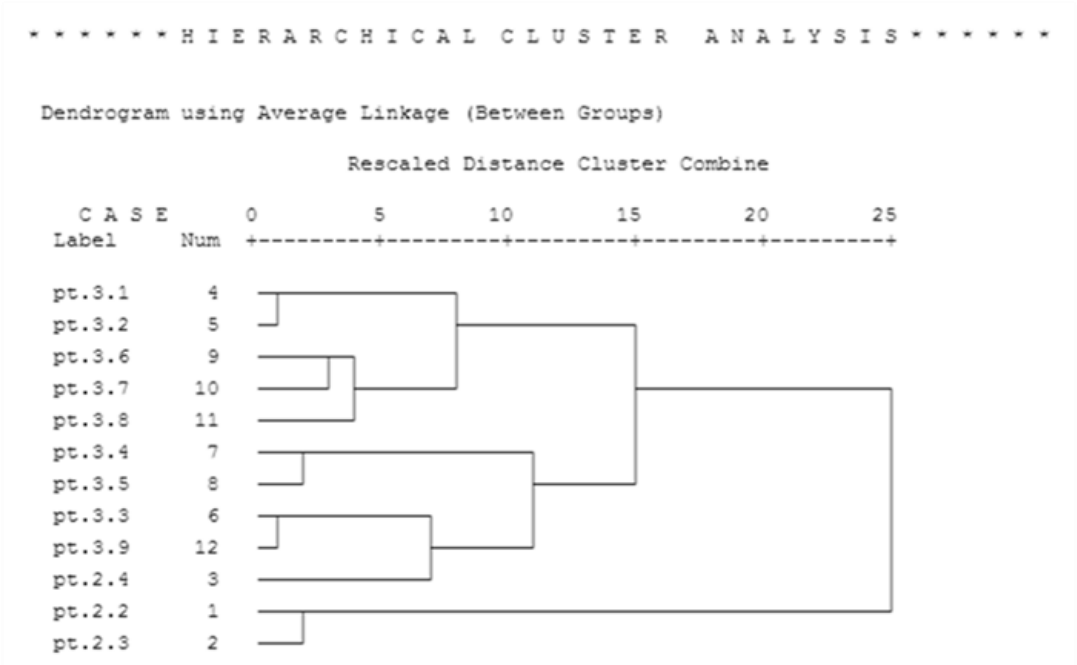

Gráfico 5: Dendograma Post-test para el GE

Respecto de lo que muestra el gráfico 5 nos parece importante destacar que:

- Existen 8 niveles a agrupación.
- Las competencias de la segunda categoría con excepción de la 2.4 forman un solo clúster.
- La competencia 2.4 se agrupa con un clúster formado por las competencias 3.3 y 3.9. Esta agrupación de similaridad indicaría la relación que existe entre la competencia referida a la capacidad para diseñar algoritmos de solución (2.4) con la competencia referida a la capacidad para usar correctamente las estructuras de programación (3.3) y la capacidad para construir e implementar programas a partir de su representación algorítmica.

Así también, al aplicar el mismo procedimiento, para los resultados del post-test del grupo control, se obtiene la distribución de clústeres que se muestra en el siguiente déndograma.

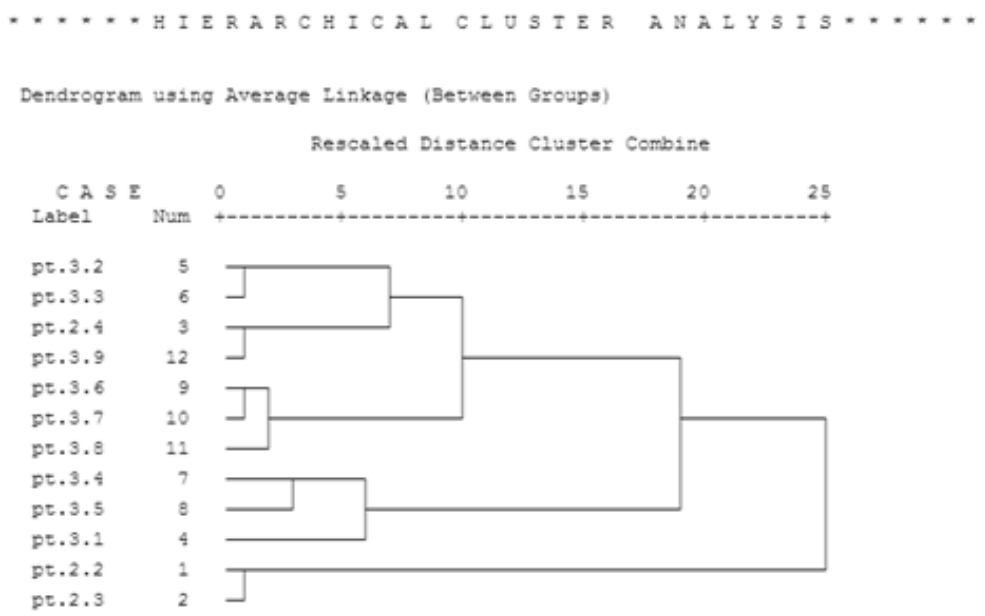

Gráfico 6: Dendograma Post-test para el GC

A partir del gráfico 6, nos parece importante destacar que:

- Al igual que la organización de clústeres para el grupo experimental, acá existen 8 niveles a agrupación.
- También en esta organización de clústeres las competencias de la segunda categoría con excepción de la 2.4 forman un solo clúster.
- En este caso, la competencia 2.4 forma un único clúster con la competencia 3.9. y en una agrupación de mayor nivel, este clúster se agrupa con el formado por la competencia 3.3.

Conforme a esto, si bien no se da la misma relación directa entre estas competencias que en caso del grupo experimental, dicha relación para el grupo control se da en el nivel superior.

Como otra forma de evaluar el impacto que han tenido las prácticas cooperativas, sobre el logro final de las 12 competencias medidas en el Post-test, se han calculado los coeficientes de correlación de Pearson para el conjunto de ellas.

Nuestra intención es determinar la relación lineal entre parejas de competencias, que nos permitan realizar otra mirada de los resultados comparativos de competencias de salida en la evaluación Post-test, entre el grupo experimental y control.

Si bien la determinación de los coeficientes de correlación de Pearson, lo hemos realizado sobre el conjunto de las 12 variables que dan cuenta de las mediciones de las competencias de salida, queremos destacar los resultados de los coeficientes de correlación entre la competencia 2.4 y las 11 restantes, y entre la competencia 3.9 y las 11 restantes.

La competencia 2.4 es particularmente importante para nosotros, ya que conforme a su definición, da cuenta de la capacidad para diseñar algoritmos de solución para los problemas de procesamiento de datos. Es una competencia que en sus términos, da cuenta de la capacidad para utilizar los conocimientos conceptuales, procedimentales y metodológicos, y aplicarlos de manera integrada y organizada, para diseñar el algoritmo de solución de un problema.

El reflejo de esta competencia, no sólo da cuenta de la capacidad para usar y aplicar los conocimientos señalados, sino que además, refleja la capacidad para usar y aplicar las estrategias de pensamiento y de resolución de esta clase de problemas, en la propia tarea de diseñar sus soluciones, actuaciones que como hemos argumentado, son difíciles de transmitir y enseñar usando sólo las clases expositivas. Son precisamente el aprendizaje, el desarrollo y perfeccionamiento de estas capacidades, lo que hemos tratado de favorecer con las actividades de trabajo cooperativo, por tanto, el efecto de las mismas, es relevante para esta investigación.

La tabla general de los valores de coeficientes de correlación de la competencia 2.4 con el resto de las competencias, para el grupo experimental y control es la siguiente:

	Competencia 2.4	
Competencias	Experimental	Control
2.2	0,860	0,687
2.3	0,855	0,692
2.4	1,000	1,000
3.1	0,961	0,927
3.2	0,965	0,966
3.3	0,975	0,965

3.4	0,891	0,847
3.5	0,924	0,857
3.6	0,917	0,684
3.7	0,907	0,708
3.8	0,899	0,718
3.9	0,990	0,992
Casos	61	83

Tabla 87: Correlaciones de la Competencias 2.4 con las 11 restantes para GE y GC

Como se observa, los coeficientes de correlación para el grupo experimental son mayoritariamente superiores que para el grupo control, mostrándose diferencias significativas entre estos dos grupos.

A partir de los valores de esta tabla, hemos categorizados por rangos, los coeficientes de correlación encontrados al emparejar la competencia 2.4 con el resto de las competencias, tanto para el grupo experimental como para el grupo control, con lo cual se obtiene la siguiente tabla:

COMPETENCIA 2.4		
Rangos de Correlaciones	CON LAS COMPETENCIAS DEL GRUPO:	
	EXPERIMENTAL	CONTROL
>=0,9 y <1	3.1, 3.2, 3.3, 3.5, 3.6, 3.7 y 3.9	3.1, 3.2, 3.3 y 3.9
>=0,8 y < 0,9	2.2, 2.3, 3.4, y 3.8	3.4 y 3.5
>=0,7 y < 0,8		3.7 y 3.8
>=0,6 y < 0,7		2.2, 2.3 y 3.6

Tabla 88: Distribución de Correlaciones de la Competencias 2.4 con las 11 restantes para GE y GC

Como muestra esta última tabla, para el caso del grupo experimental existen 7 competencias (todas de la tercera categoría), con valores de coeficiente de correlación con la competencia 2.4, en el primer rango, en tanto para el grupo control se encuentran solo 4 competencia (de la misma categoría), en este rango.

Para el segundo rango de correlaciones (>=0,8 y < 0,9), en el grupo experimental se tienen 4 competencias y sólo 2 para el caso del grupo control.

De esto se concluye que los coeficientes de correlación de la competencia 2.4 con las 11 restante se encuentran en los dos primeros rangos de coeficientes para el caso del grupo experimental, en tanto para el caso del grupo control, se observa que existen correlaciones hasta en los dos rangos inferiores.

En resumen, para el caso del grupo experimental, en la competencia 2.4 se observan un agrupamiento compacto de las medidas de relación lineal con las demás competencias que determina que todos sus valores de coeficientes son iguales o superiores a 0,8, en cambio para el caso del grupo control, se observa una disgregación de estos valores, mostrándose sólo para 6 competencias, valores

iguales o superiores a 0,8, en tanto se tienen 3 competencias con valores iguales o superiores a 0,7, otras tres con valores iguales o superiores a 0,6.

Por su parte, la competencia 3.9 se relaciona con la capacidad para construir e implementar programas a partir de su representación algorítmica, lo que implica que para su logro se debe disponer de la capacidad para diseñar las soluciones, y además, de la capacidad para usar y aplicar los recursos del lenguaje de programación, para implementar los programas de solución.

Conforme a los argumentos que se han dado en apartados anteriores, ésta también es una capacidad difícil de transmitir y enseñar usando sólo clases expositivas y/o, mostrando a los estudiantes ejemplos de códigos de programas resueltos.

La experiencia señala que para aprender, desarrollar y perfeccionar esta capacidad, los estudiantes deben involucrarse de manera directa, en las tareas de interpretación de los algoritmos y su correspondiente implementación como programas, aspecto que también hemos tratado de favorecer, mediante las actividades de trabajo cooperativo.

La tabla general de los valores de los coeficientes de correlación de la competencia 3.9 con el resto de las competencias, para el grupo experimental y control es la siguiente:

	Competencia 3.9	
Competencias	Experimental	Control
2.2	0,842	0,703
2.3	0,836	0,713
2.4	0,990	0,992
3.1	0,958	0,930
3.2	0,961	0,971
3.3	0,970	0,970
3.4	0,884	0,844
3.5	0,915	0,853
3.6	0,922	0,689
3.7	0,917	0,715
3.8	0,911	0,720
3.9	1,000	1,000
Casos	61	83

Tabla 89: Correlaciones de la Competencias 3.9 con las 11 restantes para GE y GC

Nuevamente es posible observar que los coeficientes de correlación para el grupo experimental son también mayoritariamente superiores que para el grupo.

A partir de los valores de esta última tabla, se han categorizado por rangos, los coeficientes de correlación encontrados al emparejar la competencia 3.9 con el resto

de las competencias, tanto para el grupo experimental como para el grupo control, con lo cual se obtiene la siguiente tabla:

COMPETENCIA 3.9		
Rangos de Correlaciones	CON LAS COMPETENCIAS DEL GRUPO:	
	EXPERIMENTAL	CONTROL
>=0,9 y <1	2.4, 3.1, 3.2, 3.3, 3.5, 3.6, 3.7 y 3.8	2.4, 3.1, 3.2 y 3.3
>=0,8 y < 0,9	2.2, 2.3 y 3.4	3.4 y 3.5
>=0,7 y < 0,8		2.2, 2.3, 3.7 y 3.8
>=0,6 y < 0,7		3.6

Tabla 90: Distribución de Correlaciones de la Competencias 3.9 con las 11 restantes para GE y GC

Como se observa, para el caso del grupo experimental, esta competencia tiene una relación lineal entre 0,9 y menos de 1 para con 8 competencias, en contraste con solo 4 para el caso del grupo control.

Para el segundo rango, se tienen 3 competencias para el grupo experimental, y dos para el grupo control.

Para el rango de coeficientes mayor o igual a 0,7 y menor que 0,8 se tienen 4 competencias para el grupo control, y una competencia en el último rango de categorías.

Conforme a esto, nuevamente el grupo experimental muestra una compactación mayor de los valores de correlación, a diferencia del grupo control.

6.4. Resultados referidos a las tasas de Aprobación y Reprobación

Aun cuando el principal objetivo de este trabajo es mejorar el aprendizaje, el uso y la aplicación de los conocimientos de FP, y en este contexto nuestra motivación principal ha sido crear mejores condiciones para que nuestros estudiantes adquieran, desarrollen y perfeccionen el conjunto de competencias que hemos definido, consideramos que un efecto directo de estas mejoras debiera traducirse también, en mejoras en las tasas de aprobación de los estudiantes que cursan la asignatura.

En el grafico 7 se evidencia que el grupo GE obtuvo una tasa de Aprobados de un 75.59%, en cambio los grupos GC tiene una tasa de aprobación de 39.84%.

Respectos de estos resultados queremos destacar que en los cinco semestres anteriores a estas intervenciones, el curso de FP en las dos universidades consideradas en este estudio, presentan tasas de Reprobación que fluctúan entre un 61% y hasta un 74%, con lo cual estos resultados para nuestro grupo experimental son bastante satisfactorio, ya que invierten la situación histórica de la asignatura.

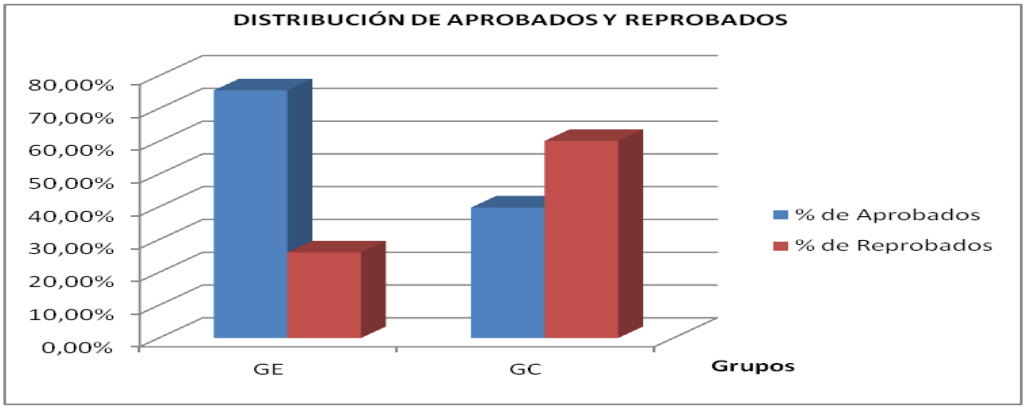

Gráfico 7: Comparativa de aprobados y reprobados entre GE y GC

6.5. Resultados referidos a la evolución de los indicadores del trabajo cooperativo.

Teniendo en cuenta que las actividades de trabajo y aprendizaje cooperativo, constituyen el elemento esencial con el cual hemos intentado mejorar el aprendizaje y el desarrollo de las capacidades para resolver los problemas de programación de nuestros estudiantes, es que a continuación presentamos los resultados de la evolución de las valorizaciones del desarrollo de estas actividades, las cuales resultan del registro de observaciones que hemos llevado a cabo durante el desarrollo de las mismas.

Para estos efectos, se han establecido tres ámbitos de valorización: (1) Interdependencia positiva al interior de los grupos; (2) Relaciones psicosociales; y (3) Construcción de significados.

Con el propósito de realizar una valoración lo más objetiva posible de cada uno de estos ámbitos, se definieron varios indicadores para cada uno de ellos, y para cada indicador, se establecieron formas de manifestación de ellos, y la correspondiente escala de cualificación para los mismos.

En la tabla 91 (Indicadores para la cualificación de las actividades cooperativas), en la página siguiente, se presentan los detalles de estas definiciones.

Luego se presentan los gráficos que muestran las cualificaciones de los indicadores para las distintas actividades cooperativas que fueron desarrolladas en la intervención.

Ámbito	Indicador	Formas en que se puede manifestar el indicador	Escalas de valorización para la observación del indicador
1. Interdependencia positiva al interior del grupo	1.1 Responsabilidad individual	• Posee sus apuntes de clase completos y ordenados • Da muestras de saber de qué tratan los conceptos vistos en clases • Da muestra de haber cumplido los compromisos personales para con la tarea grupal • Muestra compromiso en la realización de la tarea grupal	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de al menos dos de las formas de manifestación del indicador. Medio: Sólo dos de los integrantes dan muestra de al menos dos de la formas de manifestación del indicador Alto: Todos los integrantes del grupo dan muestra de al menos tres formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de las cuatro formas de manifestación del indicador.
	1.2 Propuestas de organización y método de trabajo	• Propone una forma de organizar el trabajo grupal • Explica al resto su propuesta de organización del trabajo grupal • Se muestra abierto a analizar y discutir otras propuestas de organización del trabajo grupal	Nulo Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de las dos primeras manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de al menos dos de la formas de manifestación del indicador Alto: Todos los integrantes del grupo dan muestra de al menos dos formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de las tres formas de manifestación del indicador, y además se implican en reformulaciones de las propuestas de organización y método de trabajo
	1.3 Interpelación de responsabilidades	• Denuncia el no cumplimiento de responsabilidades de uno o más miembros del grupo • Reprocha de manera manifiesta el no cumplimiento de responsabilidades de uno o más miembros del grupo • Exige el cumplimiento de responsabilidades de uno o más miembros del grupo	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de algunas de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de al menos dos de la formas de manifestación del indicador Alto: Todos los integrantes del grupo dan muestra de al menos una forma de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de una de las formas de manifestación del indicador.
	1.4 Preguntas de organización o método de trabajo	• Realiza una o más preguntas referidas a la organización o método de trabajo acordado, ya sea a un integrante del grupo en particular o a todo el grupo. • Pregunta a un integrante o al grupo en general si se ha entendido a cabalidad el método de trabajo que se acordó utilizar en la tarea cooperativa.	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de algunas de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de algunas de las manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos una forma de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de una de las formas de manifestación del indicador.

	1.5 Aclaración y/o complementación de aspectos de organización del trabajo	• Solicita a un integrante que se le aclaren un aspecto de la organización del trabajo. • Solicita al grupo que se le aclare un aspecto de la organización del trabajo. • Solicita a un integrante o al grupo en general que se le aclaren más de un aspecto de la organización del trabajo.	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de algunas de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de algunas de las manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos una forma de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de una de las formas de manifestación del indicador.
2. Relaciones psicosociales	2.1 Refuerzo y/o aprobación	• Expresa acuerdo y/o aprobación con las aportaciones de uno de los integrantes del grupo • Expresa acuerdo y/o aprobación con las aportaciones del resto de los integrantes del grupo	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de alguna de las manifestaciones del indicador Alto: Todos los integrantes dan muestra de alguna de las formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de las dos formas de manifestación del indicador de manera reiterativa.
	2.2 Estimulación y/o animación del grupo	• Estimula y/o anima el trabajo de uno de los integrantes del grupo • Estimula y/o anima el trabajo de todos los integrantes del grupo	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de alguna de las manifestaciones del indicador Alto: Todos los integrantes dan muestra de alguna de las formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de las dos formas de manifestación del indicador de manera reiterativa.
	2.3 Gratitud	• Da muestras de reconocimiento y/o aprecio por los aportes de uno de los integrantes del grupo. • Da muestras de reconocimiento y/o aprecio por los aportes de todos los integrantes del grupo.	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de alguna de las manifestaciones del indicador Alto: Todos los integrantes dan muestra de alguna de las formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de las dos formas de manifestación del indicador de manera reiterativa.

	2.4 Diálogo social	• Respeta las opiniones y/o aportes de los demás integrantes del grupo • Presta atención e interés cuando los demás integrantes del grupo intervienen • Permite y/o facilita que el resto de los integrantes del grupo emitan sus opiniones y/o juicios • Favorece y/o anima la participación en la discusión y el trabajo grupal	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de al menos dos formas de manifestaciones del indicador Alto: Todos los integrantes dan muestra de a lo menos tres formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de las formas de manifestación del indicador de manera reiterativa.
3. Construcción de significados	3.1 Explicación y/o argumentación	• Explica con sus palabras uno o más conceptos que están siendo utilizados en la tarea de resolución • Explicita a uno o más integrantes del grupo la idea general que se usará en la resolución de un problema. • Explica uno o más procedimientos. • Expone argumentos para justificar el uso de una idea para resolver un problema • Fundamenta el uso de ideas, conceptos y/o procedimientos intentando convencer al grupo sobre su relevancia para resolución de un problema.	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de a lo menos tres manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos tres formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de tres de las formas de manifestación del indicador.
	3.2 Reformulación y/o síntesis	• Reformula uno o más conceptos implicados en la solución del problema • Reformula uno o más procedimientos implicados en la solución del problema • Sintetiza varias ideas respecto de la resolución del problema • Sintetiza varios procedimientos de solución en uno solo • Sintetiza varias propuestas de solución para un problema	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de a lo menos dos manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos tres formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de tres de las formas de manifestación del indicador.

	3.3 Preguntas de contenido y/u opinión	• Pregunta a un integrante en particular, con el objeto de clarificar el uso de uno o más conceptos, procedimientos, o ideas implicadas en la solución del problema • Pregunta al grupo, con el objeto de clarificar el uso de uno o más conceptos, procedimientos, o ideas implicadas en la solución del problema • Solicita a un integrante del grupo la opinión respecto de la interpretación que hace de un procedimiento o idea implicada en la solución del problema. • Solicita al grupo la opinión respecto de la interpretación que hace de un procedimiento o idea implicada en la solución del problema.	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de a lo menos dos manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos dos formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de dos de las formas de manifestación del indicador.
	3.4 Justificaciones	• Aporta evidencias para fundamentar el uso de uno o más conceptos, procedimientos o ideas para resolver el problema • Justifica el uso de uno o más conceptos o procedimiento para resolver el problema • Justifica y/o argumenta el uso de una idea para resolver el problema • Justifica el uso de uno o más concepto o procedimientos, desde su propia experiencia de resolución de problemas • Justifica y/o argumenta desde su propia experiencia, el uso de una idea para resolver el problema	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de a lo menos dos manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos dos formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de dos de las formas de manifestación del indicador.
	3.5 Planteamiento de discrepancias	• Plantea discrepancias con el uso de uno o más contenidos en la solución del problema • Plantea discrepancia con el uso de uno o más procedimientos en la solución del problema • Plantea discrepancia con el uso y/o aplicabilidad de una o más ideas respecto a la resolución del problema	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de alguna de las manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos dos formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de dos de las formas de manifestación del indicador.

	3.6 Aclaración y/o complementación de contenidos	• Aclara uno o más contenidos a uno de los integrantes del grupo • Complementa uno o más contenidos a uno de los integrantes del grupo • Aclara uno o más contenidos al resto del grupo • Complementa uno o más contenidos al resto del grupo • Aclara y/o complementa una o más ideas o propuestas implicadas en la solución del problema	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de alguna de las manifestaciones del indicador Alto: Todos los integrantes del grupo dan muestra de al menos dos formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de más de dos de las formas de manifestación del indicador.
	3.7 Explicación de tarea	• Explicita la tarea a desarrollar a uno de los integrantes de un grupo • Explicita la tarea a desarrollar al resto de los integrantes del grupo	Nulo: Ninguno de los integrantes del grupo muestra alguna de las manifestaciones del indicador Bajo: Sólo uno de los integrantes da muestra de alguna de las manifestaciones del indicador. Medio: Sólo dos de los integrantes dan muestra de alguna de las manifestaciones del indicador Alto: Todos los integrantes dan muestra de alguna de las formas de manifestación del indicador. Muy Alto: Todos los integrantes dan muestras de alguna de las formas de manifestación del indicador de manera reiterativa.
	3.8 Calidad de la tarea	• La solución se presenta de manera ordenada • En la solución se usan correctamente los conceptos • En la solución se usan correctamente los procedimientos • En la solución se usa correctamente la metodología • En la solución se usa correctamente el lenguaje algorítmico y/o las estructuras de programación. • El algoritmo y/o el programa de solución es eficaz y eficiente	Nulo: La solución no presenta ninguna de las manifestaciones del indicador Bajo: La solución presenta a lo menos dos de las manifestaciones del indicador Medio: La solución presenta lo menos tres de las manifestaciones del indicador Alto: La solución presenta a lo menos cuatro de las manifestaciones del indicador. Muy Alto: La solución presenta todas las manifestaciones del indicador.

Tabla 91: Indicadores para la cualificación de las actividades cooperativas

A continuación presentamos el análisis de la evolución de los indicadores de los tres ámbitos (Interdependencia positiva al interior del grupo; Relaciones psicosociales; y Construcción de significados), para los tres momentos del desarrollo de las actividades de trabajo y aprendizaje cooperativo para el semestre académico que son:

- *Primer momento;* con 3 actividades, que se desarrollaron entre el inicio del tratamiento de los contenidos y la primera evaluación acumulativa, que cronológicamente va desde el tercer período de la segunda semana de clases (s2-p3), al tercer período de la tercera semana de clases (s3-p3).
- *Segundo momento;* con 8 actividades, que se desarrollaron entre la primera y segunda evaluación acumulativa, que cronológicamente van desde el primer período de la quinta semana (s5-p1), al segundo período de la novena semana (s9-p2), y
- *El tercer momento*; con 10 actividades, que se desarrollaron cronológicamente entre el tercer período de la décima semana (s10-p3), y el primer período de la décimo sexta semana (s16-p1).

1.- Observaciones referidas al ámbito de Interdependencia positiva al interior del grupo

Los siguientes 3 gráficos dan cuenta de los niveles de logro para el indicador de *Responsabilidad individual* (1.1), para cada uno de los tres momentos de actividades cooperativas.

Este indicador hace referencia al cumplimiento del compromiso personal de cada integrante del grupo frente a la tarea grupal, cuestión que es fundamental no sólo para la integridad del grupo y para que los integrantes se sientan parte de él, sino que también es importante para el propio desempeño del grupo como equipo de trabajo.

Como se observa en el gráfico 8 en la página siguiente, en la primera sesión de actividad cooperativa (s2-p3) en 5 grupos, no se registraron formas de manifestación de este indicador, en tanto en 15 de ellos, observamos que al menos uno de sus integrantes dio muestras de tener sus apuntes de clases completos y ordenados, o que al menos uno de ellos sabían de que trataban los contenidos vistos en las clases anteriores, y en algunos casos, apreciamos acciones que se corresponden con el compromiso de realización de la tarea grupal.

En la segunda actividad (s3-p2) no evidenciamos cambios respecto de la anterior, en tanto para la tercera actividad apreciamos que en diez grupos, en al menos dos de sus integrantes, se observó que, disponían de los apuntes de clases completos y ordenados, sabían de qué trataron los contenidos vistos en clases, y/o dieron

muestras de compromiso con la realización de la tarea grupal. En cuatro grupos no registramos evidencias de algunas de las manifestaciones de este indicador.

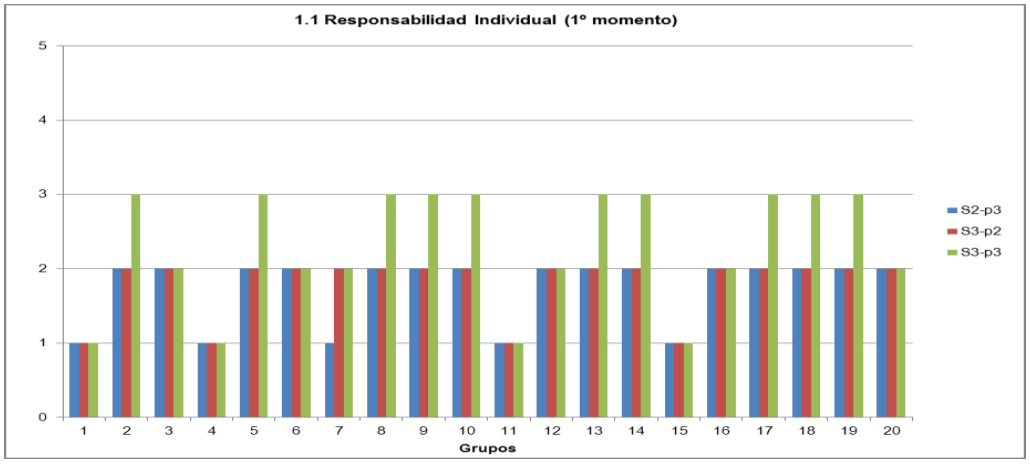

Gráfico 8: Indicador 1.1; 1er Momento

Pensamos que el lento avance en este aspecto se debe principalmente a que los estudiantes generalmente no están acostumbrados al trabajo en equipos, por cuanto, no tienen el hábito de cumplir con sus responsabilidades individuales en tareas académicas, frente a sus pares.

Para el segundo momento el que considera el desarrollo de 8 actividades cooperativas (gráfico 9), apreciamos mejoras importantes, ya que para la primera y segunda actividades (s5-p1, s5-p3) en 6 grupos todos sus integrantes mostraron varias formas de manifestación del indicador.

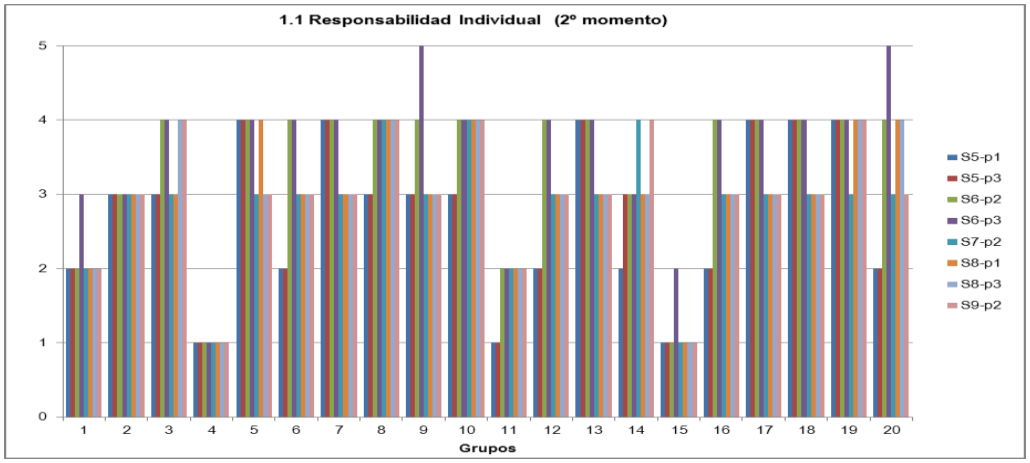

Gráfico 9: Indicador 1.1; 2º Momento

Para la tercera y cuarta actividad (s6-p2 y s6-p3), se verificó un cambio importante puesto que para 13 de los 20 grupos, se evidenció que todos sus integrantes poseían apuntes ordenados y completos, dieron muestras de saber de qué trataban los

conceptos vistos en clases, y mostraron compromiso por la realización de la tarea grupal.

Sin embargo, apreciamos una disminución en el número de grupos que alcanzan estos buenos niveles entre la quinta y octava actividad.

De este gráfico se aprecia también que regularmente en todas las actividades de este segundo momento, a lo menos 2 grupos no mostraron evidencias de que sus integrantes cumplieran con el compromiso individual frente a la tarea grupal, lo que se constata nuevamente en varios de los indicadores del tercer ámbito (Construcción de significados), reflejándose en el bajo y en ocasiones nulo desempeño de estos grupos en las tareas de resolución de los problemas.

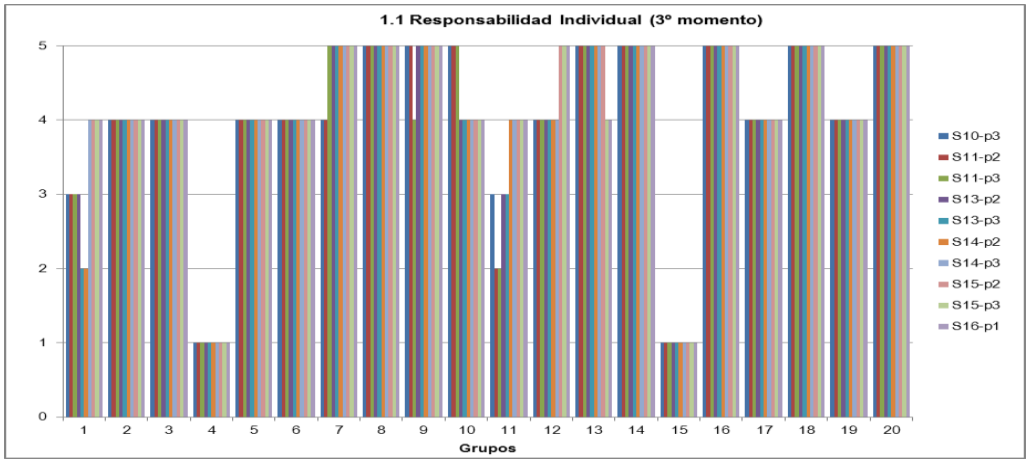

Gráfico 10: Indicador 1.1; 3er Momento

Para el caso del tercer momento (grafico 10), se observó una mejora que consideramos importante, en el nivel de logro de este indicador, ya que apreciamos muy buenas evidencias de cumplimiento del compromiso individual en alrededor de 18 grupos, sin embargo, a pesar de los esfuerzos que hicimos, en dos de los grupos, sus integrantes se mantuvieron sin dar manifestaciones de este tipo.

Desde la práctica constatamos que cuando en los grupos se dio un buen nivel de cumplimiento de las responsabilidades individuales, esto influyó positivamente en la generación de un clima apropiado para el trabajo grupal, ya que apreciamos que en éstos grupos se consiguió un ambiente de trabajo muy cordial, dónde sus integrantes sentían que el trabajo en equipo les ayudaba en su aprendizaje y de paso disminuía la sensación de ansiedad frente al fracaso, que es propio en este tipo de asignaturas.

Los gráficos 11, 12 y 13 dan cuenta de los niveles de logro para el indicador de *Propuesta de organización y método de trabajo* (1.2), el que se refiere a las acciones de planteamiento de ideas para coordinarse mutuamente al interior del grupo, así como también a la formulación de las mismas respecto de maneras para llevar a cabo la tarea grupal

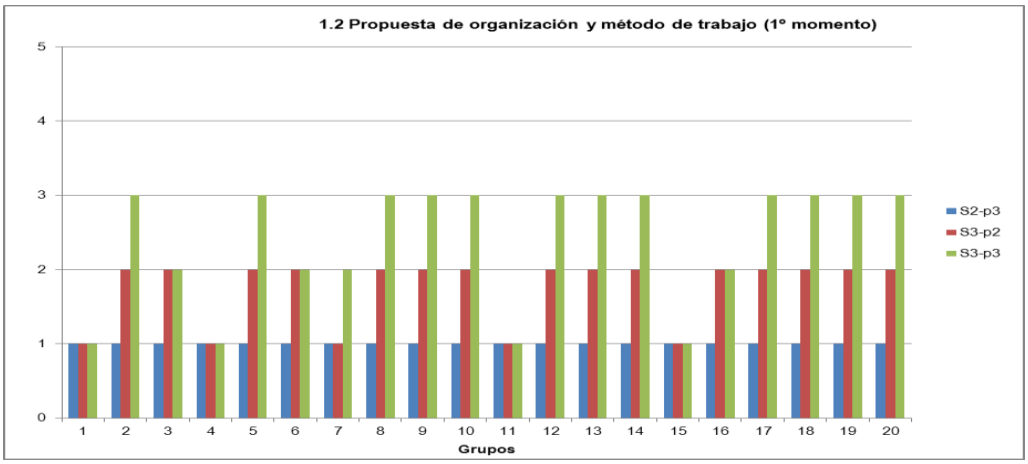

Gráfico 11: Indicador 1.2; 1er Momento

Para el primer momento (gráfico 11), en la primera actividad (s2-p3) en ninguno de los 20 grupos pudimos constatar manifestaciones de este indicador. Apreciamos que la mayoría de los grupos trabajaron de manera muy inorgánica, sin una estructura de organización o método de trabajo claro, lo que influyó fuertemente en sus desempeños.

En la segunda actividad (s3-p2) ya se apreció un cambio significativo, ya que en a lo menos la mitad de los grupos se observó que al menos un estudiante planteó a sus compañeros una propuesta para organizar el trabajo del grupo.

Para 5 grupos, se recogió evidencias de que estas propuestas fueron planteadas por a lo menos dos de sus integrantes, y en algunos casos, estas fueron analizadas y discutidas al interior del grupo. Esto mejoró en la tercera actividad (s3-p3) donde se pudo observar a lo menos12 grupos en esta condición, aunque tuvimos 4 grupos en los que no apreciamos este tipo de propuestas.

Para el segundo momento de actividades (gráfico 12 en la página siguiente), en las dos primeras actividades (s5-p1 y s5-p3) se apreció una cantidad importante de grupos en los que todos sus integrantes propusieron formas de organización del trabajo grupal, y además se involucraron en el análisis y discusión de ellas para establecer consensos respecto de cuál era la más apropiada.

A partir de la tercera actividad de este momento (s6-p2) comenzamos a evidenciar grupos en los que las propuestas de organización y método de trabajo fueron

revisadas y reformuladas durante el desarrollo del trabajo, lo que da cuenta de que al interior de estos grupos se llevó a cabo una (o más) evaluación del método de trabajo puesto en práctica, y que como resultado de ello, se hicieron cambios, lo que para nosotros es una muestra de la calidad del trabajo que dio en algunos de los grupos.

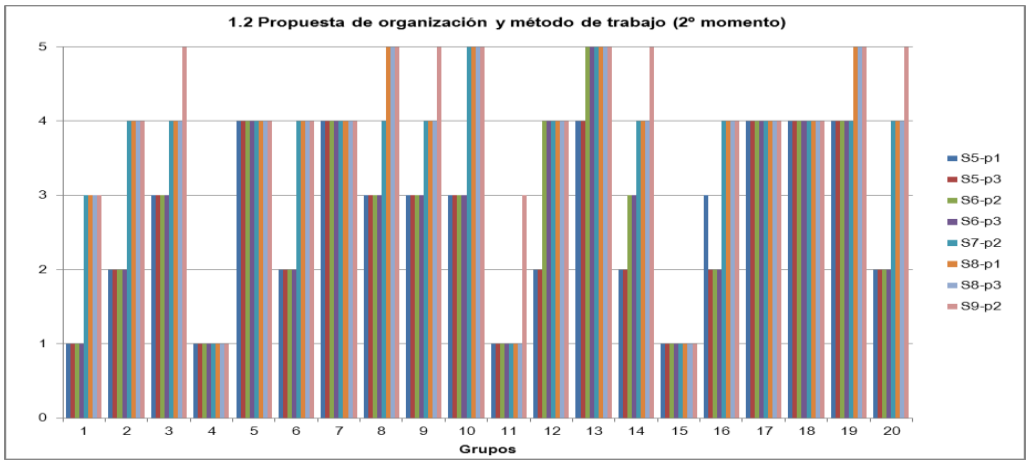

Gráfico 12: Indicador 1.2; 2º Momento

Desde la quinta actividad de este bloque (s7-p2) en promedio 16 grupos alcanzaron un muy buen nivel, que se expresa que todos sus integrantes plantean propuestas de organización y método de trabajo, que estas son analizadas para acordar la mejor de ellas, y que en algunos casos, estas propuestas fueron evaluadas y reformuladas durante el desarrollo de la propia actividad. Sin embargo, los mismos grupos (4 y 15) que dieron muestras de tener nulas manifestaciones de responsabilidad individual (indicador 1.1), finalizan este momento de actividades en esta misma condición para este indicador.

En la mayoría de los casos brindamos apoyo para que los grupos que presentaban dificultades en la organización, pudiesen mejorar esta condición, sin embargo en los dos grupos señalados, no fue posible mejorar esta condición, lo que es una consecuencia de que en estos grupos se diera también, una nula responsabilidad individual y compromiso por la realización de la tarea grupal por parte de sus integrantes.

Estos resultados se mantienen en los mismos niveles para las 6 primeras actividades del tercer momento (s10-p3, s11-p2, s11-p3, s13-p2, s13-p3, y s14-p2) como se muestra en el gráfico 13 (en la página siguiente), y mejoran a partir de la séptima actividad (s14-p3), donde en 18 de los grupos se observó un buen nivel de propuestas de organización, las que en muchos casos fueron revisadas y modificadas durante el desarrollo de la tarea cooperativa, sin embargo, se mantienen los dos mismos grupos en el nivel nulo.

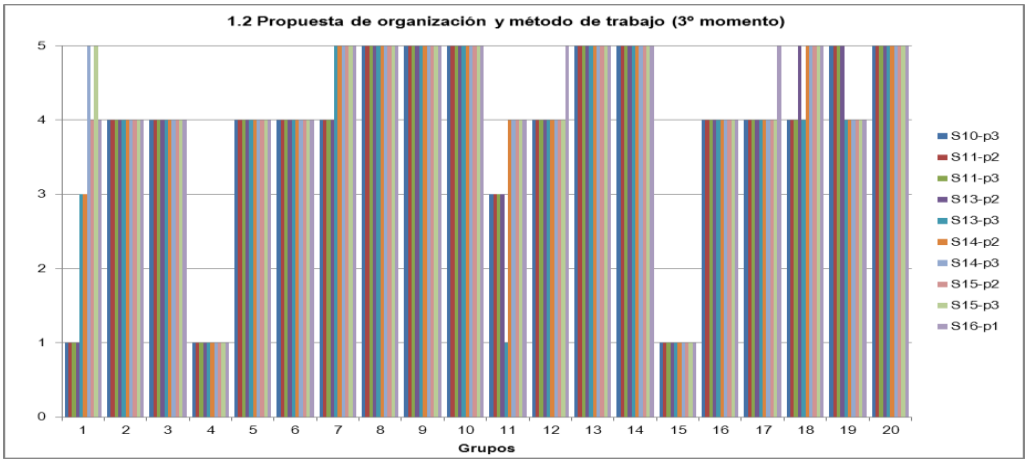

Gráfico 13: Indicador 1.2; 3er Momento

El hecho de que los integrantes de cada grupo hagan propuestas de organización y método de trabajo, evidencia también el nivel de responsabilidad individual que se da al interior de ellos, siendo esta también una forma en que se manifiesta el compromiso con el trabajo grupal.

La evolución de este indicador para 18 de los grupos da cuenta de la habilidad que desarrollaron para organizarse y establecer sus propios métodos de trabajo, cuestión que es reconocida como esencial para el buen desempeño de ellos. Esto nos parece un logro importante, en especial cuando tratamos con estudiantes que no están habituados a formas de trabajo colectivo.

Los siguientes tres gráficos muestran los niveles de logro para los tres momentos del indicador *Interpelación de responsabilidades* (1.3), el que se refiere a las demandas que realiza uno o más integrantes del grupo, para que otros asuman el compromiso personal de trabajo dentro del grupo.

Del gráfico 14 (en la página siguiente), se deduce que entre tres y cuatro grupos, hasta dos de sus integrantes denunciaron el no cumplimiento de responsabilidades de uno o más miembros de su grupo, y en algunos de estos casos, registramos acciones de reproche hacia quienes no cumplían con ello, lo que se observó en la primera y segunda actividad del primer momento (s2-p3 y s3-p2).

Estas situaciones generaron algunas dificultades en las relaciones interpersonales al interior de estos grupos, lo que nos obligó a intervenir en ellos.Para la primera actividad en 15 grupos no se registraron manifestaciones de este indicador, y en dos de ellos sólo uno de sus integrantes denunció incumplimiento de responsabilidades de algunos de sus compañeros de grupo. Para la tercera actividad de este momento (s3-p3), en 8 grupos se observó que solo uno de los integrantes realizó acciones de interpelación de responsabilidades a uno o más integrantes de su grupo, lo que

también se manifestó en la existencia de ciertas dificultades en el desempeño de estos grupos.

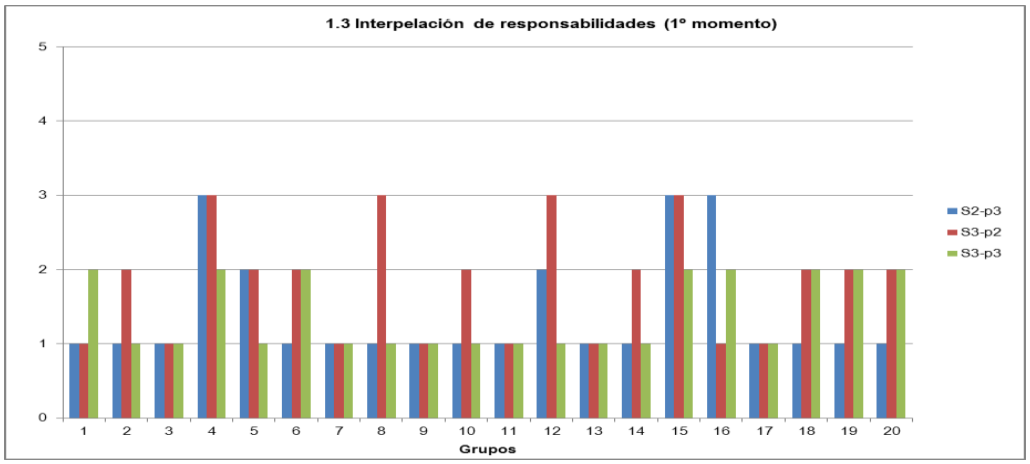

Gráfico 14: Indicador 1.3; 1er Momento

Sin embargo destaca el hecho de que para la tercera de estas actividades (s3-p3) en 12 de los grupos no existió interpelación de responsabilidades, aunque en 8 de ellos, al menos uno de sus integrantes aún hacía reclamaciones y/o reproches para que sus compañeros asumieran de mejor manera sus responsabilidades al interior de grupo.

Para el segundo momento (gráfico 15), se verificó un aumento de los grupos que presentan altos niveles de interpelación de responsabilidades, en especial en la primera y segunda actividad de este bloque, lo que determinó que un número importante de grupos presentaran problemas en sus desempeños.

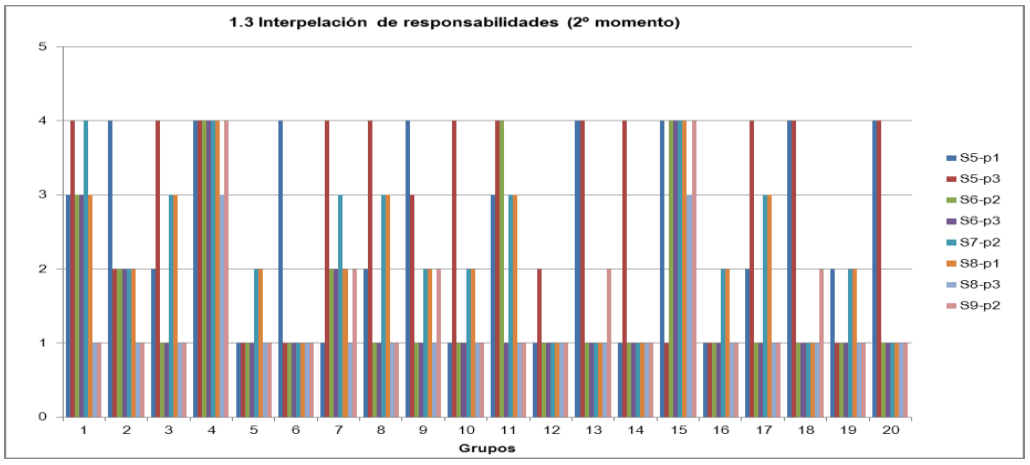

Gráfico 15: Indicador 1.3; 2º Momento

De hecho en la primera actividad de este bloque (s5-p1) se observa que existieron 8 grupos donde las demandas por asumir las responsabilidades se hicieron colectivas, con recriminaciones mutuas.

En la segunda (s5-p3) se apreciaron 12 grupos en esta condición, lo que nos obligó a intervenir para buscar las causas de ello, y proponer acciones y actitudes que permitiesen reducir y minimizar los conflictos dentro de los grupos por esta causa.

Como se aprecia en el gráfico 15 (en la página anterior), para la tercera actividad de este bloque se consiguió mejorar este aspecto, disminuyendo de 12 a sólo 3 grupos con este tipo de dificultades, y que en 14 de los grupos no se observaron acciones de interpelación de responsabilidades.

Para la quinta y sexta actividades de este momento (s7-p2, s8-p1) nuevamente se aprecia un empeoramiento de este indicador ya que se tienen 8 y 7 grupos respectivamente, que evidencian dificultades por esta razón, lo que determinó que nuevamente debiéramos realizar acciones de intervención para los grupos que estaban teniendo las dificultades.

Para la séptima y octava actividad (s8-p3 y s9-p2) se consiguió mejorar, ya que se tuvieron 18 y 14 grupos respectivamente, que no presentaron problemas por la interpelación de responsabilidades al interior de los grupos., aunque aún teníamos 2 grupos que mantenían dificultades.

Para las actividades del tercer momento (gráfico 16), se consiguió que en promedio 18 grupos no tuvieran acciones de interpelación de responsabilidades, y que en dos grupos se mantuvieran las denuncias y los reproches por el incumplimiento de responsabilidades, expresadas en a lo menos por uno de los integrantes de los grupos 4 y 15, que son los que habían presentado dificultades de manera reiterativa y que no conseguimos mejorar.

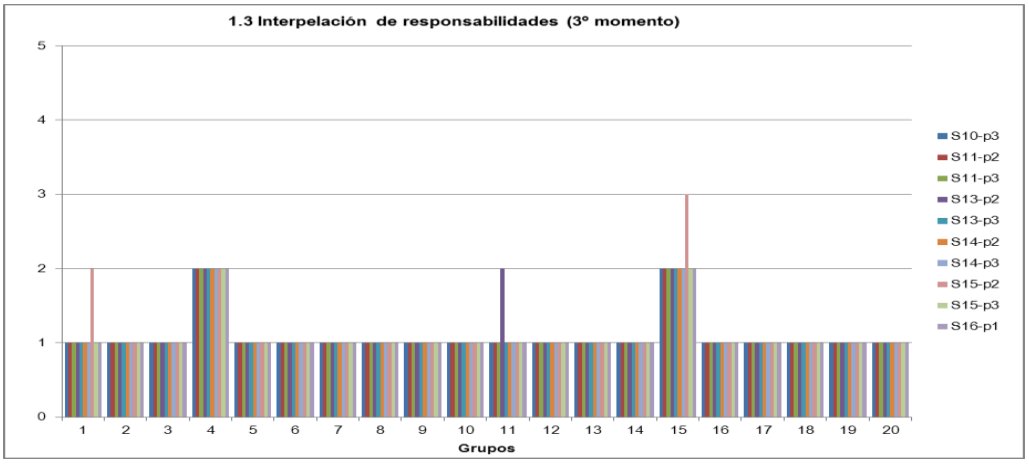

Gráfico 16: Indicador 1.3; 3er Momento

Consideramos que esta evolución hacia niveles nulos de interpelación de responsabilidades es positiva para el buen funcionamiento de los grupos, ya que esto es una condición que genera las confianzas necesarias entre los integrantes del grupo.

Esta condición es una muestra de que los integrantes de un grupo siente que el resto está asumiendo sus responsabilidad en el desarrollo de la tarea grupal, lo que influye positivamente en la actitud para asumir la responsabilidad y el compromiso individual con la tarea grupal, y que para nosotros, consecuentemente ha de influir en la calidad del desempeño de los grupos.

En los gráficos 17, 18 y 19 que presentamos a continuación se muestran los niveles de logro para el indicador, *Preguntas de organización o método de trabajo* (1.4), el que está referido a la existencia de manifestaciones de dudas sobre la organización o forma de llevar a cabo la tarea o actividad grupal, durante el desarrollo de las actividades cooperativas.

En la primera actividad cooperativa (s2-p3, en el gráfico 17), no se observó en ninguno de los 20 grupos la existencia de preguntas de organización o método de trabajo, lo que es consecuente con lo que se apreció para el indicador 1.2 donde en los 20 grupos no existieron propuestas de organización y método de trabajo.

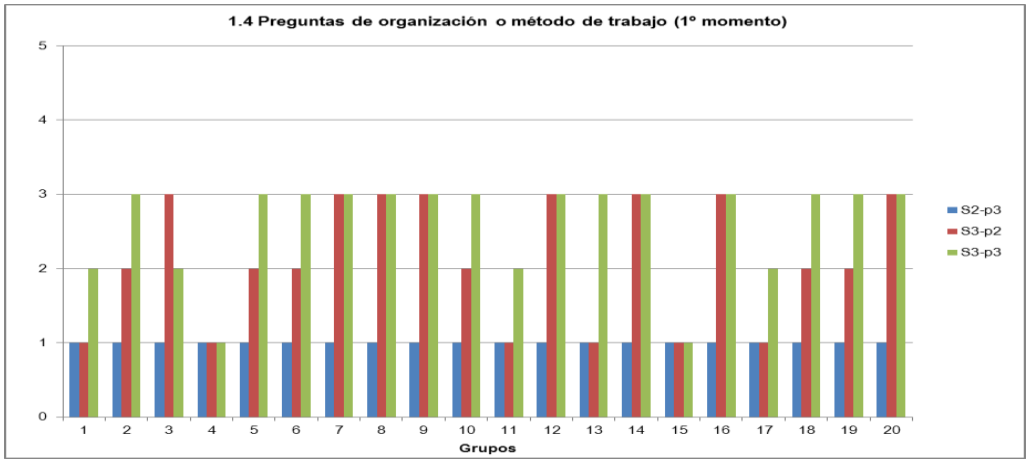

Gráfico 17: Indicador 1.4; 1er Momento

Esto se mantuvo en 6 grupos para la segunda actividad (s3-p2), en tanto, en 14 grupos constatamos que al menos uno de sus integrantes planteó a sus compañeros preguntas referidas a la organización y/o método de trabajo, incluyendo casos en que uno de los integrantes del grupo, preguntó al resto si entendían el método de trabajo que se pondría en práctica. Se apreció también que en algunos de estos grupos, las acciones de este tipo fueron realizadas por a lo menos dos de sus integrantes.

Para la tercera actividad de este momento (s3-p3) registramos que en 14 grupos, a lo menos dos de sus integrantes se implicaron en preguntas de este tipo, y en dos grupos no existieron este tipo de preguntas.

Para el segundo momento (gráfico 18) en las dos primeras actividades (s5-p1, y s5-p3) se constató que a lo menos en 6 grupos, todos sus integrantes realizaron preguntas referidas a la organización del trabajo, ya sea para clarificar el método de trabajo o para constatar si el resto tenía claridad respecto del modo en que se había organizado el trabajo. En tres de los grupos no existieron preguntas de este tipo en las dos primeras actividades.

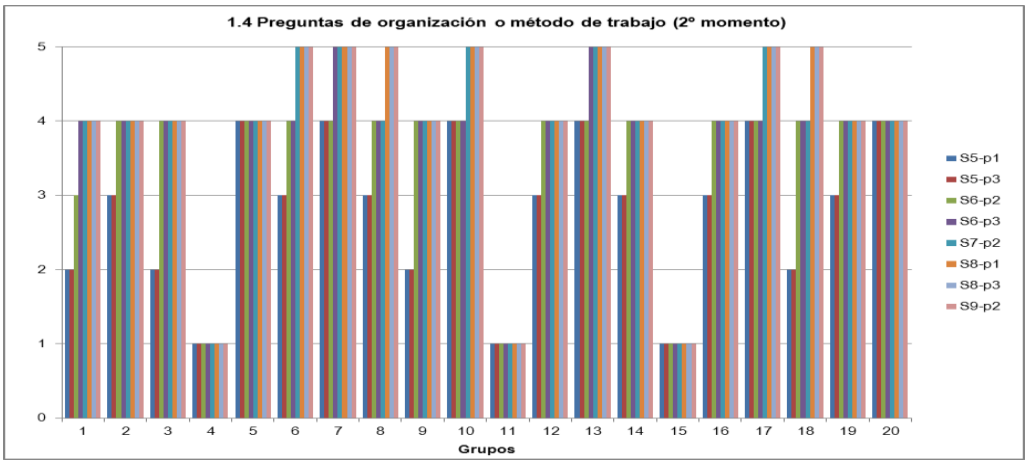

Gráfico 18: Indicador 1.4; 2º Momento

Para la tercera y cuarta actividades de este bloque (s6-p2 y s6-p3) se apreció un incremento del número de grupos (15 a 16), en los cuales este tipo de acciones fueron realizadas por todos los integrantes del grupo, aunque en las siguientes actividades se dio una disminución del número de grupos en esta condición, terminado este segundo momento de actividades, con 10 grupos en dicha condición, y 3 grupos en los que no se observaron preguntas de este tipo.

Hacia las últimas actividades de este momento, se vio que en al menos 7 grupos este tipo de preguntas, además de ser realizadas por todos los integrantes (ya sea para responder a sus dudas o para constatar si los demás tenían claridad respecto de la forma de organización del trabajo), en especial las del segundo tipo, fueron hechas de manera reiterativa durante el trabajo grupal.

Para el tercer momento (gráfico 19, en la página siguiente), se observó que de los tres grupos que no dieron muestra de preguntas de este tipo sólo se mantienen dos de ellos en esta condición, y en un número no menor de 8 grupos estas acciones las realizaron todos los integrantes y de manera reiterativa.

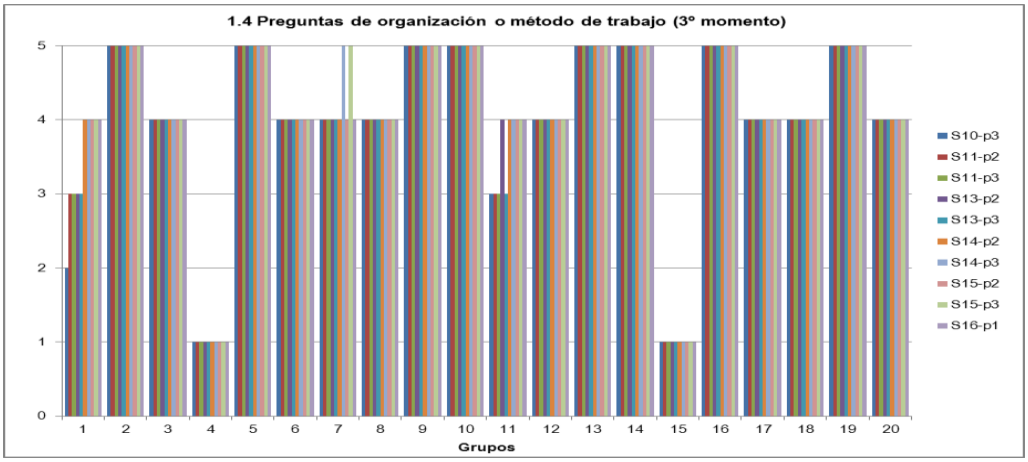

Gráfico 19: Indicador 1.4; 3er Momento

A partir de la sexta actividad de este bloque (s14-p2 en adelante) registramos 18 grupos en la condición señalada en el párrafo anterior, y se mantiene los mismos dos grupos en que no observamos preguntas o acciones de este tipo.

Este es otro de los aspectos importantes del trabajo grupal, puesto que la existencia de preguntas de este tipo denota la necesidad que tienen los integrantes de un grupo, primero por entender cómo se organizará el trabajo que han de realizar, como un aspecto de la clarificación de la responsabilidad individual para con el éxito del trabajo grupal, y segundo, para buscar y establecer consensos respecto de la mejor forma de organizarse, lo que de paso, podría responder a la preocupación por alcanzar un buen desempeño en el resultado de la tarea grupal.

En los siguientes tres gráficos (20, 21, y 22) se muestran los niveles alcanzados en el indicador, *Aclaración y/o complementación de aspectos de la organización* (1.5), el que se refiere evidencias de acciones dirigidas a la clarificación y/o complementación de ideas o propuestas para organizarse o realizar la tarea grupal.

Este indicador se relaciona con el anterior, puesto que refleja la existencia de respuestas respecto de las preguntas de organización o método de trabajo (1.4).

Al igual que lo ocurrido con los indicadores 1.2 y 1.4, en la primera actividad cooperativa (gráfico 20 en la página siguiente), en los 20 grupos no observamos acciones de aclaración y/o complementación de aspectos de organización del trabajo grupal.

Para la segunda actividad de este primer momento (s3-p2) apreciamos que en 10 de los grupos solo uno de sus integrantes realizó acciones como pedir a uno de los demás integrantes que aclarase uno o más aspectos de la organización del trabajo, y algunos casos en que esta misma solicitud fue hecha por el resto de los integrantes

del grupo. En tanto, en 5 grupos, registramos que este tipo de acciones fueron realizadas por a lo menos 2 de los integrantes de cada grupo.

Apreciamos una mejora en la tercera actividad de este momento (s3-p3) ya que vimos que en 11 de los grupos a lo menos dos estudiantes solicitaron aclaraciones de este tipo al interior de sus grupos.

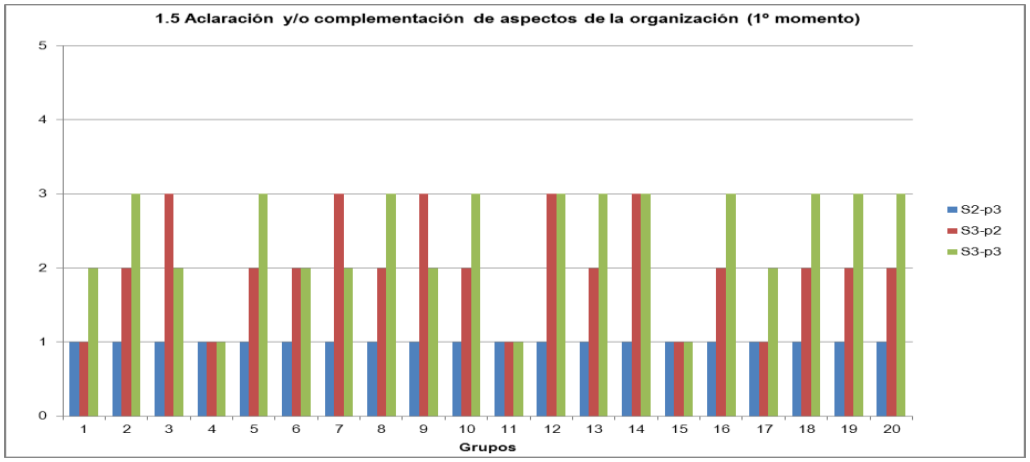

Gráfico 20: Indicador 1.5; 1er Momento

Para el segundo momento (gáfico 21), en las dos primeras actividades (s5-p1 y s5-p3) se observó que en 6 grupos todos sus integrantes se implicaron en acciones de aclaración y complementación de aspectos de organización.

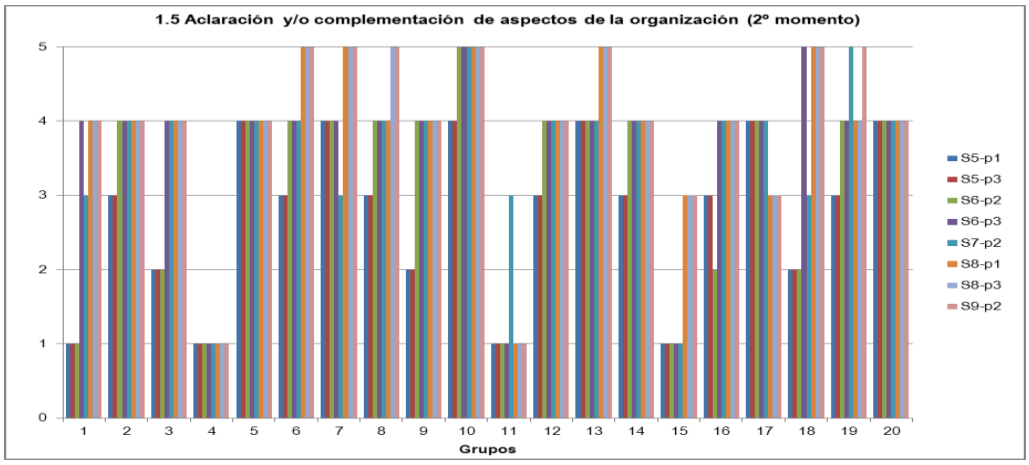

Gráfico 21: Indicador 1.5; 2º Momento

Entre la tercera y sexta de estas actividades, registramos que entre 11 y 15 grupos todos sus integrantes solicitaron que se le aclarasen uno o más aspectos de la organización, ya sea a uno de sus compañeros en particular, o a su grupo en general, pero entre 2 y 4 de los grupos no evidenciamos acciones de este tipo.

A partir de la tercera actividad de este bloque en algunos de los grupos se apreció que todos sus integrantes solicitaron aclaración de estos aspectos en más de una ocasión durante el desarrollo de la actividad, en especial cuando dentro del grupo se modificó la organización de la tarea durante el desarrollo de la misma.

Para el tercer momento (gráfico 22), pudimos observar que en promedio en 18 de los grupos, todos sus integrantes, en una o más ocasiones pidieron que se les aclarasen uno a más aspectos de la organización, durante el desarrollo de la actividad, en tanto, se mantuvieron los dos grupos (4 y 15), en los que no evidenciamos acciones de este tipo.

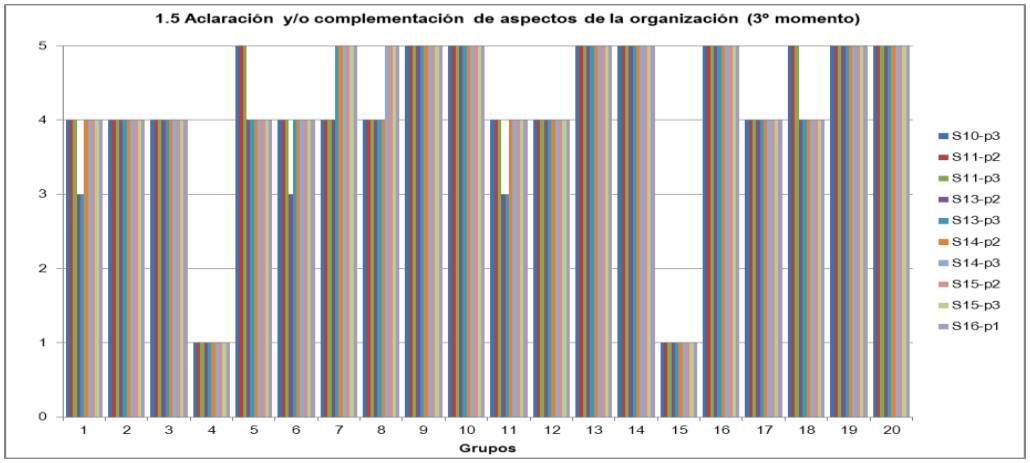

Gráfico 22: Indicador 1.5; 3er Momento

Para nosotros, el desarrollo de este tipo de acciones es importante, ya que es una manifestación de que los integrantes de los grupos se involucran de manera efectiva y reflexiva en la organización del trabajo grupal, aportando con ideas respecto de cómo puede optimizarse el desarrollo de la tarea grupal, y el propio desempeño del grupo.

2.- Observaciones referidas al ámbito de *Relaciones psicosociales*

Con el segundo ámbito de indicadores, abarcamos otros de los aspectos relevantes del trabajo cooperativo, que se relaciona con el ambiente psicosocial que tiene lugar al interior de los grupos, mientras de desarrolla el trabajo cooperativo.

Para este ámbito se establecieron 4 indicadores, cuyos resultados para los tres momentos de actividades cooperativas, pasamos a analizar a continuación.

Tratamos en primer término con el indicador, *Refuerzo y/o aprobación* (2.1), el que está referido a las manifestaciones de acciones que expresen acuerdo con las aportaciones o contenidos que se hacen al interior del grupo durante el trabajo cooperativo.

En los siguientes tres gráficos se muestran los resultados de lo observado para los tres momentos.

Para la primera actividad de este momento (s2-p3 en gráfico 23), en ninguno de los 20 grupos se apreciaron manifestaciones de refuerzo y/o aprobación, ya sea hacia alguien en particular o hacia el resto de los integrantes de grupo.

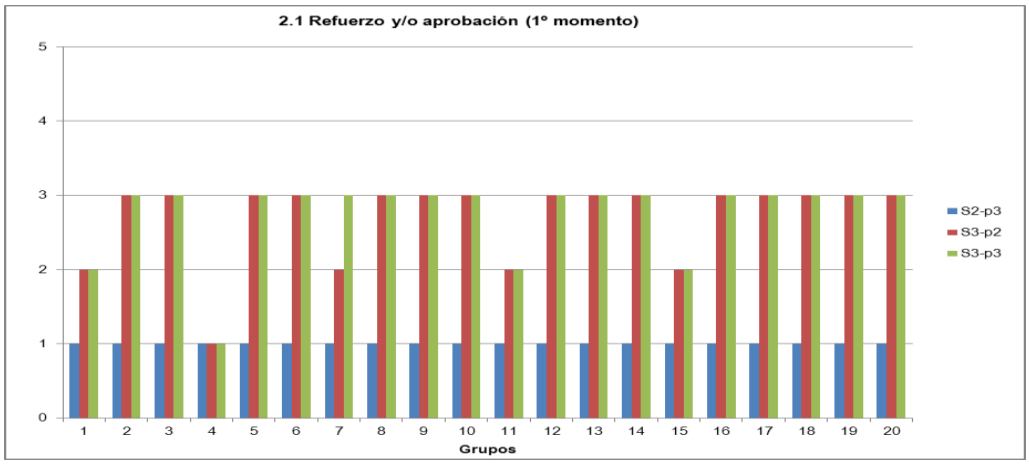

Gráfico 23: Indicador 2.1; 1er Momento

Para la segunda actividad (s3-p2), en 15 de los grupos se verificó que al menos dos de los integrantes de estos grupos tuvieron palabras de refuerzo y/o aprobación por las aportaciones de sus compañeros de grupo durante el desarrollo del trabajo cooperativo, en tanto en uno de los grupos aún no observamos acciones de este tipo.

En la tercera actividad de este primer momento (s3-p3) se pudo observar que en 16 grupos se dieron expresiones de refuerzo y/o aprobación por los aportes de uno o más compañeros de sus grupos, en tanto se mantenía un grupo en que ninguno de sus integrantes dio muestras de este tipo de manifestaciones.

Para la primera actividad del segundo momento (s5-p1 en gráfico 24, en la página siguiente), se verificó que en 16 grupos a lo menos dos de sus integrantes reforzaron y aprobaron las ideas para resolver el problema aportadas por sus compañeros de grupos, sin embargo se incrementó a 4 el número de grupos en que no registramos acciones de este tipo.

Para la segunda (s5-p3) comenzamos a apreciar grupos donde todos los integrantes se implicaron en acciones de refuerzo y/o aprobación con las aportaciones del resto de los integrantes de sus grupos, especialmente cuando se discutían las distintas ideas respecto de cómo elaborar la solución al problema en el que trabajaban.

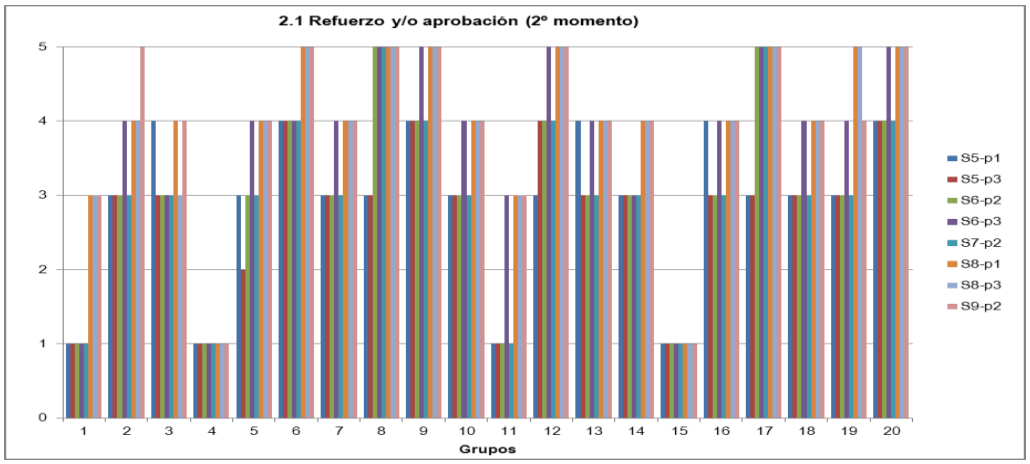

Gráfico 24: Indicador 2.1; 2º Momento

A partir de la tercera actividad y hasta la octava (última de este momento) registramos mejoras, ya que apreciamos que en un número creciente de grupos se dan este tipo de acciones durante las discusiones y el establecimiento de consensos respecto de cómo resolver el problema, llegando a un número de 17 grupos en esta condición, pero se mantienen los dos grupos en los cuales no observamos este tipo de acciones.

Para el tercer momento de actividades (gráfico 25), en la primera actividad (s10-p3) registramos sólo 5 grupos en los cuales observamos que todos sus integrantes se implicaban en acciones como las que hemos señalado, incluso observamos que en algunos de estos grupos, este tipo de acciones se dio de manera reiterativa durante el desarrollo de la tarea de resolución de problemas.

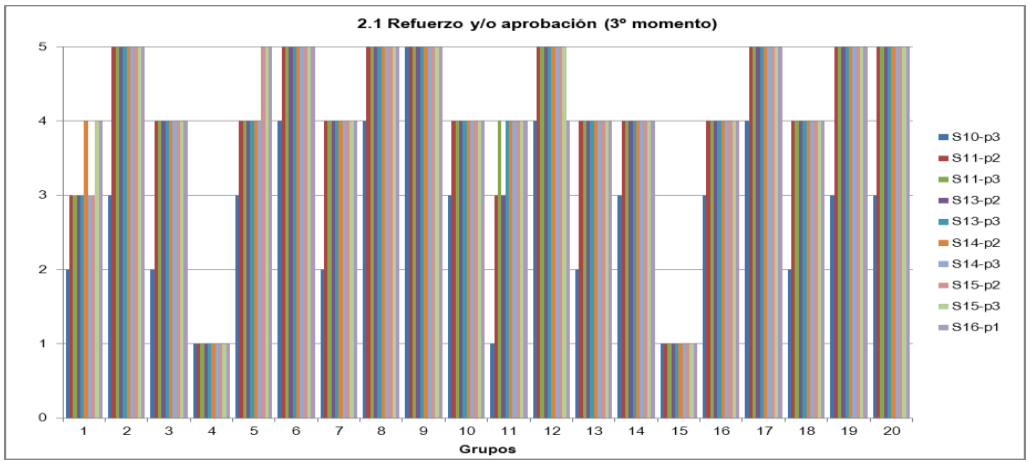

Gráfico 25: Indicador 2.1; 3er Momento

En la segunda actividad de este momento (s11-p2), se observó que en 16 grupos, durante las discusiones para trabajar en la solución del problema, todos los integrantes reconocían las aportaciones de sus pares de grupo, y en la mayoría de los casos dichas aportaciones eran consideradas importantes para construir la

solución a los problemas. En tanto en dos grupos, que son el 4 y el 15 no se apreciaron acciones de este tipo, ya que fundamentalmente hacia un trabajo individual.

A partir de la tercera actividad y hasta la décima se mantuvieron cerca de 18 grupos en esta condición, al igual que los dos mismos grupos que no manifestaron acciones de este tipo.

Para nosotros la existencia de acciones de refuerzo y/o aprobación tienen dos efectos sobre la calidad del trabajo grupal, el primero referido con el establecimiento de los consensos en el manejo de los conocimientos, y el segundo se relaciona con un aspecto de calidad de las relaciones al interior del grupo, siendo ambas dimensiones beneficiosas para los resultados del aprendizaje grupal.

Los siguientes gráficos (26, 27, y 28), muestran los resultados de nuestras observaciones para los tres momentos para el indicador *Estimulación y/o animación del grupo* (2.2), el que está referido a las acciones dirigidas a motivar y animar el trabajo grupal y el propio aprendizaje.

Para la primera actividad del primer momento (s2-p3, en el gráfico 26), en 12 grupos se constató que uno de sus integrantes estimuló y animó el trabajo de sus compañeros de grupos durante el desarrollo de la tarea cooperativa, y en 8 grupos no observamos acciones de este tipo.

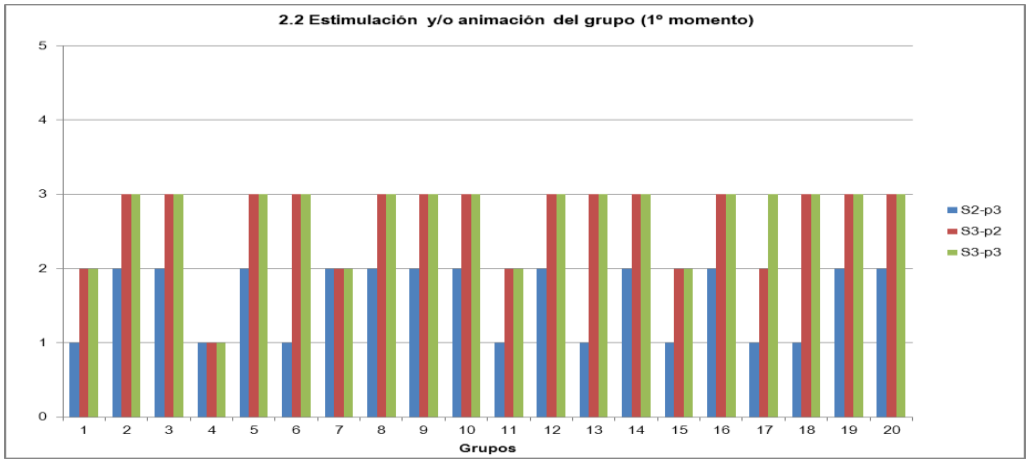

Gráfico 26: Indicador 2.2; 1er Momento

Para la segunda actividad (s3-p2), en 14 grupos registramos que a lo menos dos de los sus integrantes estimularon y/o animaron a sus compañeros, lo que incluyó en algunos caso, estímulos para realizar el trabajo y las expresiones de argumentos respecto del beneficio que esto podría representar para el aprendizaje de cada integrante del grupo.

Para la tercera y última actividad de este primer momento (s3-p3) tuvimos un grupo más en el que observamos acciones como las señaladas en el párrafo anterior, y se mantenía el grupo 4 en el cual ninguno de sus integrantes realizó acciones de este tipo.

Para la primera actividad del segundo momento (s5-p1 en gráfico 27), apreciamos un aumento de la cantidad de grupos en que sus integrantes no dieron muestras de acciones de este tipo, pero en cuatro de los ellos pudimos evidenciar un alto nivel de acciones de estimulación y/o animación lo que significa que todos los integrantes de estos grupos expresaban palabra y/o frases que ayudaban a estimular al resto para asumir la tarea al interior de sus respectivos grupos, a participar de las discusiones, a realizar defender sus aportes, y a tener actitudes que favorecieran el aprendizaje individual y grupal.

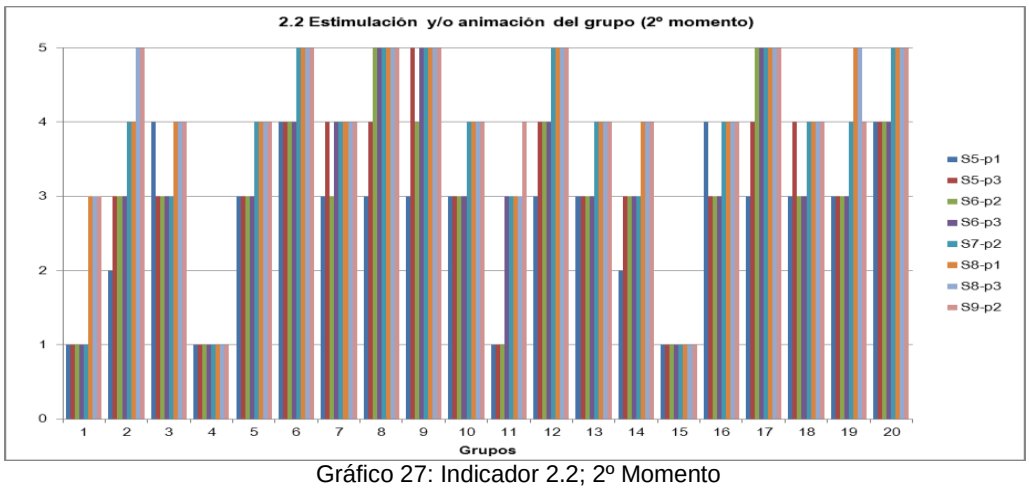

Gráfico 27: Indicador 2.2; 2º Momento

A partir de la segunda actividad de este segundo momento (s5-p3) comenzamos a observar que en este tipo de acciones al interior de los grupos, no sólo tomaban parte todos los integrantes, sino que estas acciones se daban de manera reiterativa durante las sesiones de trabajo cooperativo, esto determina que en la octava actividad observamos 17 grupos en esta condición, en tanto, los mismos dos grupos que habían presentado dificultades también en otros indicadores, se mantenían sin mostrar evidencias de este tipo de acciones.

Los buenos niveles para este indicador se mantuvieron en las 10 actividades del tercer momento (gráfico 28 en la página siguiente), con lo cual en la última actividad (s16-p1) registramos 18 grupos en los que apreciamos muy buenos niveles de manifestaciones de este indicador, y los mismos dos grupos no daban muestras de acciones asociadas a este indicador.

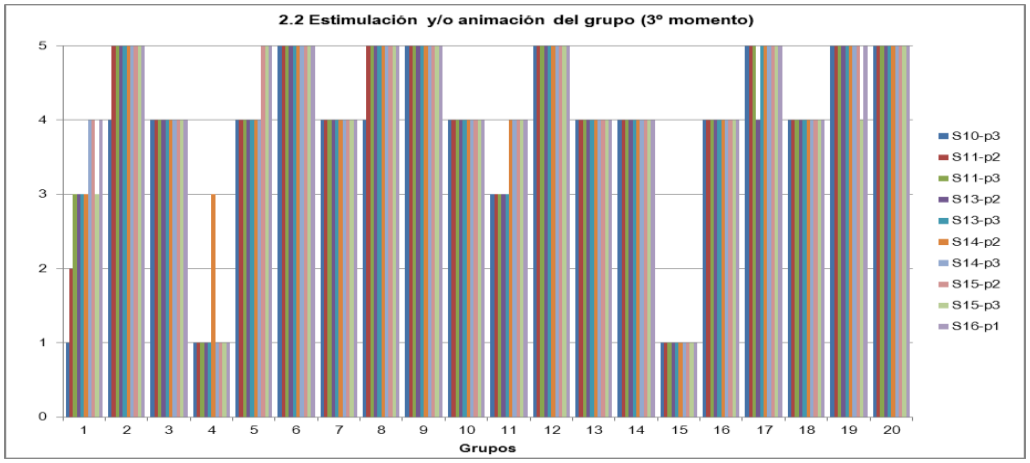

Gráfico 28: Indicador 2.2; 3er Momento

Para nosotros, las acciones de estimulación y/o refuerzo dan cuenta del ambiente de animación que se dio al interior de los grupos, lo que sin duda favorece tanto la calidad de las relaciones entre sus integrantes, como el sentido de identidad y pertenencia de los integrantes, lo que de paso, creemos que ha favorecido la disposición para participar, aportar, y defender ideas y reflexiones propias de la tarea de resolución de los problemas, mejorando el desempeño de los grupos, pero por sobre todo, influyendo en los aprendizajes individuales de los estudiantes.

En los gráficos 29, 30, y 31 se presentan los resultados de la cualificación del indicador *Gratitud* (2.3), el que está referido a las acciones de manifestación de reconocimiento y aprecio por las aportaciones de otros, para los tres momentos de actividades cooperativas.

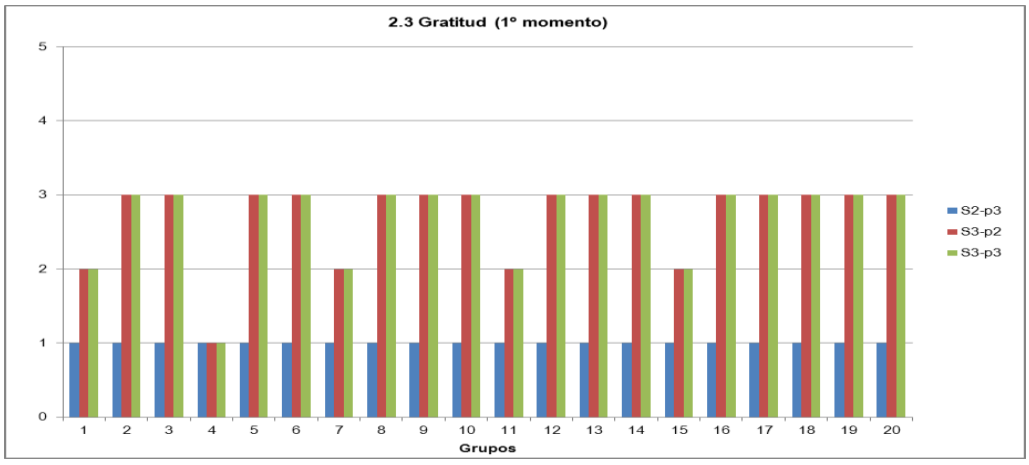

Gráfico 29: Indicador 2.3; 1er Momento

En la primera actividad del primer momento (s2-p3, gráfico 29), en los 20 grupos no se registraron manifestaciones de gratitud entre sus integrantes, sin embargo para la segunda actividad (s3-p2), en 16 de los grupos pudimos apreciar, que a lo menos

dos estudiantes de estos grupos expresaron hacia uno o más de sus compañeros, palabras de gratitud y aprecio ya sea por los aportes y/o compromiso mostrado para con el trabajo grupal y el desarrollo de la tarea de resolución de problemas. En algunos casos estas expresiones de gratitud se plantearon en términos de que los aportes y compromisos representaban una ayuda importante para el proceso de aprendizaje.

Situación similar ocurrió en la tercera y última actividad de este primer momento. En el grupo 4 no se apreciaron acciones de este tipo en ninguna de las tres actividades de este primer momento.

Para la primera y segunda actividad del segundo momento (s5-p1 y s5-p3 en gráfico 30), se observaron 4 grupos donde todos sus integrantes, especialmente al final de la tarea, se daban las gracias mutuamente por los aportes y el compromiso en el trabajo, aunque en 4 de los grupos no observamos acciones de este tipo.

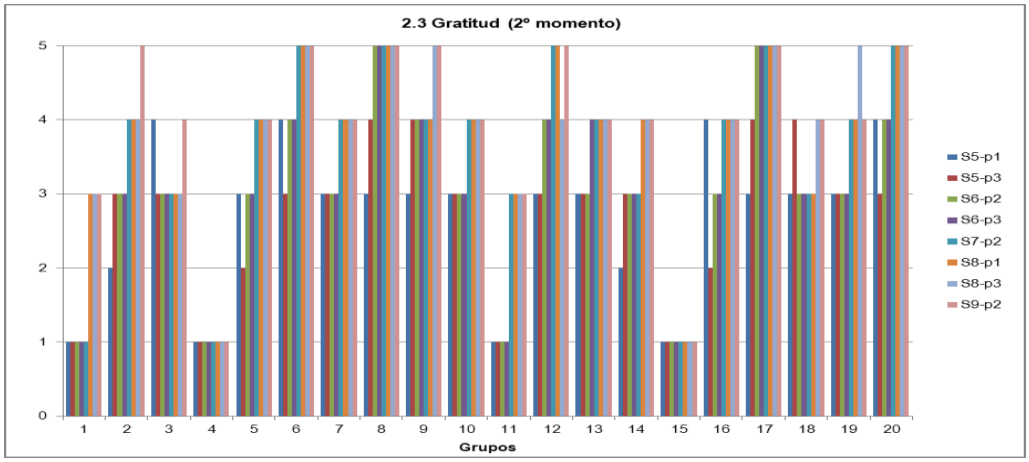

Gráfico 30: Indicador 2.3; 2º Momento

A partir de la tercera actividad de este momento (s6-p2) se evidenció mejora en el nivel de este indicador, en el sentido de que para al menos dos grupos, las expresiones de gratitud y reconocimiento se observaron de manera reiterativa durante el trabajo cooperativo.

Las mejoras en los indicadores determinaron que en la última actividad de este segundo momento, en 16 de los grupos registramos manifestaciones colectivas de gratitud y reconocimientos por los aportes realizados durante el desarrollo de la tarea de resolución de los problemas.

Como se aprecia en el gráfico, los grupos 4 y 15 concluyeron este segundo momento sin manifestaciones de expresiones de este tipo.

Para las actividades del tercer momento (gráfico 31), se mantuvo el buen nivel de este indicador, y en la última actividad apreciamos muestras recurrentes de acciones como las que hemos señalado, aunque los mismos dos grupos terminaron sin dar muestras de acciones de este tipo.

Si bien las acciones de gratitud y reconocimiento, no tienen una influencia directa sobre el aprendizaje, consideramos que este tipo de acciones han influido en el clima de cordialidad y el sentimiento de que el trabajo grupal, los aportes y la disposición de cada integrante para discutirlos y defenderlos durante la tarea de resolución de los problemas, es una importante contribución de los otros para el aprendizaje personal.

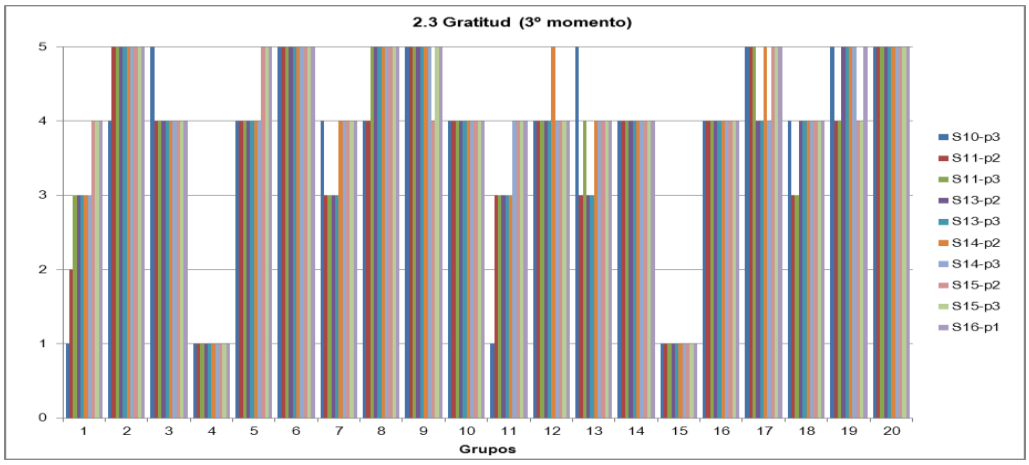

Gráfico 31: Indicador 2.3; 3er Momento

El último indicador de este segundo ámbito es el de *Dialogo social* (2.4), el que está referido a las expresiones que favorecen la participación en la discusión y el trabajo grupal.

En los gráficos 32, 33, y 34 se presentan los niveles de logro para este indicador y para los tres momentos de actividades cooperativas.

Para la primera actividad (s2-p3, en gráfico 32 en la página siguiente), constatamos un bajo nivel de diálogo social en 18 de los grupos. Se observaron grupos donde la comunicación no era del todo fluida, con algunos estudiante que no respetaban los momentos de dialogo de sus compañeros, y que trataban de imponer sus propias opiniones al resto de sus compañeros de grupos.

Ya en la segunda actividad, y considerando algunas instrucciones que les dimos para mejorar este aspecto, se apreció que en a lo menos 16 grupos el dialogo fue más ordenado, y respetando los tiempos y opiniones al interior de cada grupo durante el trabajo y la reflexión cooperativa.

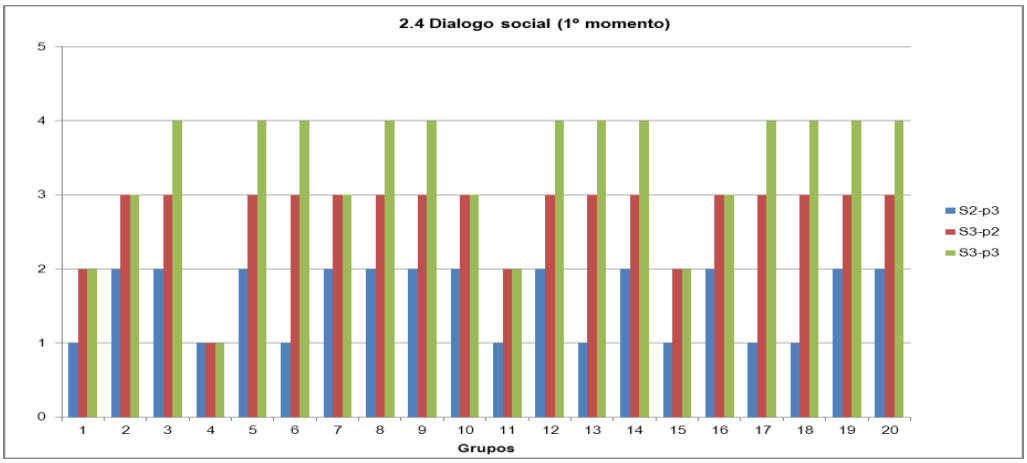

Gráfico 32: Indicador 2.4; 1er Momento

Ya en la tercera actividad de este momento, apreciamos que en 12 de los grupos se dio lo que consideramos como un buen nivel de diálogo social, ya que no sólo se respetaba la participación de todos en el dialogo, sino que cuando alguno de ellos realizaba menos intervenciones que el resto, se le anima a emitir sus opiniones o a refutar o cuestionar la de los otros.

Para las actividades del segundo momento (gráfico 33), se observó que para las primeras cuatro actividades, en 4 de los grupos, el dialogo entre ellos fue de muy mala calidad, no respetaban los momentos de cada uno, en general solo uno o dos querían controlar la participación e intervenir en las opiniones y las reflexiones colectivas solo ellos. Esto hizo que apreciáramos algunos conflictos lo que nos obligó a intervenir para intentar mejoras en este aspecto.

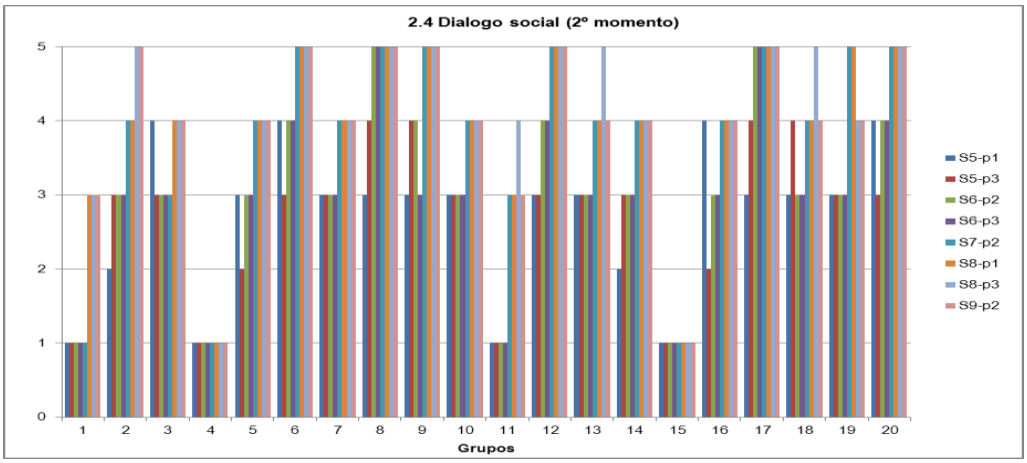

Gráfico 33: Indicador 2.4; 2º Momento

Para la quinta actividad de este momento conseguimos que en al menos un grupo estas dificultades pudieran minimizarse y alcanzar un dialogo al interior del grupo más armónico y participativo.

Como contraparte, en a lo menos 15 grupos se constató una muy buena calidad de dialogo al interior del grupo, con orden, respetando y animando la participación de todos.

Para la última actividad de este momento, se evidenciaron 16 grupos en la buena condición que señalamos en el párrafo anterior, sin embargo se mantenían los conflictos en dos de los grupos.

Para la primera actividad del tercer momento (s10-p3 en el gráfico 34), se observó un retroceso ya que a los dos grupos que habían presentado conflictos en la comunicación, se sumaron otros dos grupos, lo que nos obligó a intervenir nuevamente en ellos.

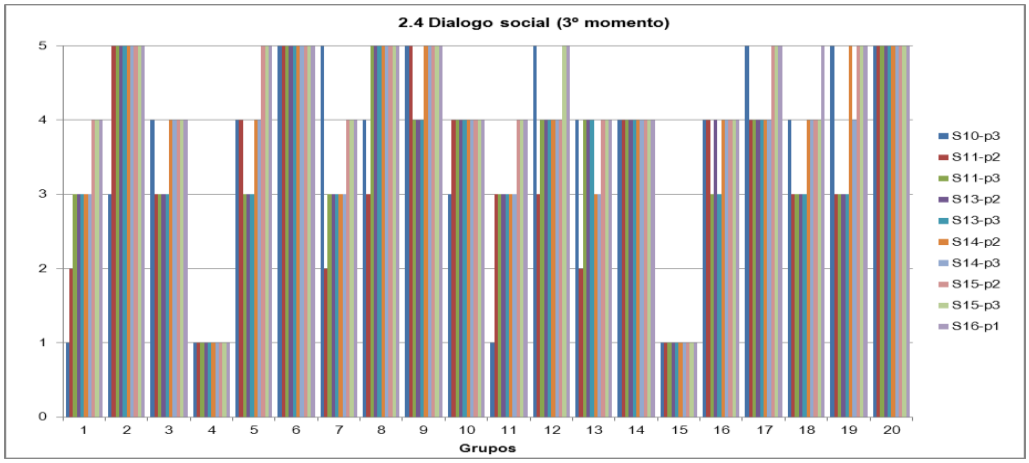

Gráfico 34: Indicador 2.4; 3er Momento

A partir de la tercera actividad (s11-p3) se constatan mejoras en este aspecto, ya que apreciamos a lo menos 10 grupos con muy buena calidad de dialogo al interior de sus grupos.

Como se aprecia en gráfico, en las siguientes actividades se va dando una evolución positiva de este indicador, que determina que a partir de la octava actividad (s15-p2) registramos que en 18 de los grupos el nivel del dialogo y la comunicación al interior de estos grupos fue muy bueno, ya que se apreciaba respeto por las opiniones de los otros, actitud de atención cuando los otros hablan, facilitación del dialogo dando las oportunidades para que todos participen de él, y animando a la participación y expresión de las ideas de aquellos estudiantes que resultaban más retraídos para intervenir en los diálogos y reflexiones durante el desarrollo de la tarea cooperativa. Como contraparte, se mantuvieron los dos grupos con conflictos internos, por lo que

calidad del dialogo, la participación y el propio desempeño de estos grupos (4 y 15), terminó siendo muy malo.

La evolución positiva de este indicador, con excepción de los dos grupos ya señalados, es para nosotros, concordante con la evolución de los otros indicadores, en el sentido de que de alguna forma, es una consecuencia de la evolución de los otros.

Un buen nivel de diálogo social al interior de los grupos, nos parece que contribuyó al buen clima dentro de los grupos de trabajo, favoreciendo el aprendizaje individual y la calidad del desempeño personal y colectivo de los integrantes de los grupos.

3.- Observaciones referidas al ámbito de *Construcción de significados.*

Los indicadores de este tercer ámbito nos permiten dimensionar el efecto que tienen las actividades cooperativas sobre el aprendizaje de los contenidos, y el desarrollo de las competencias para aplicar y usar los conocimientos en las tareas de resolución de problemas.

Conforme a la definición de los indicadores, con ellos intentamos cualificar las participaciones individuales de los integrantes de los grupos, en aquellas acciones que consideramos que tienen directa relación con las actuaciones que son intelectualmente relevantes, durante el proceso de resolución de los problemas, actuaciones que al producirse deberían tener una influencia directa sobre el aprendizaje de cada estudiante.

En este contexto, se han establecido 8 indicadores, cuyos resultados para los tres momentos de actividades cooperativas, pasamos a analizar a continuación.

Tratamos en primer término con el indicador, *Explicación y/o argumentación* (3.1), el que está referido a las acciones dirigidas a hacer comprensible una idea, expresando el propio punto de vista, apoyado en razones o descripciones, intentando convencer mediante la utilización de fundamentos y aportando evidencias.

En los siguientes tres gráficos se muestran los resultados de lo observado para los tres momentos.

En la primera actividad cooperativa (s2-p3 en el gráfico 35), teníamos la mitad de los grupos sin manifestar acciones de explicación y/o argumentación, y para la otra mitad registramos que sólo uno de los integrantes de estos grupos intentaron explicar con sus palabras uno o más de los conceptos, o procedimientos implicados en la solución del problemas, existiendo otros caso, en los que algunos de los estudiantes intentaron exponer argumentos para justificar el uso de una idea para resolver el problema.

Para la segunda actividad (s3-p2) en 8 grupos se apreció que a lo menos dos de sus integrantes intentaron explicar a sus compañeros un procedimiento involucrado en la solución del problema, y algunos casos en que se atrevieron a exponer fundamentos para respaldar una idea de cómo resolver el problema que se estaba tratando de resolver.

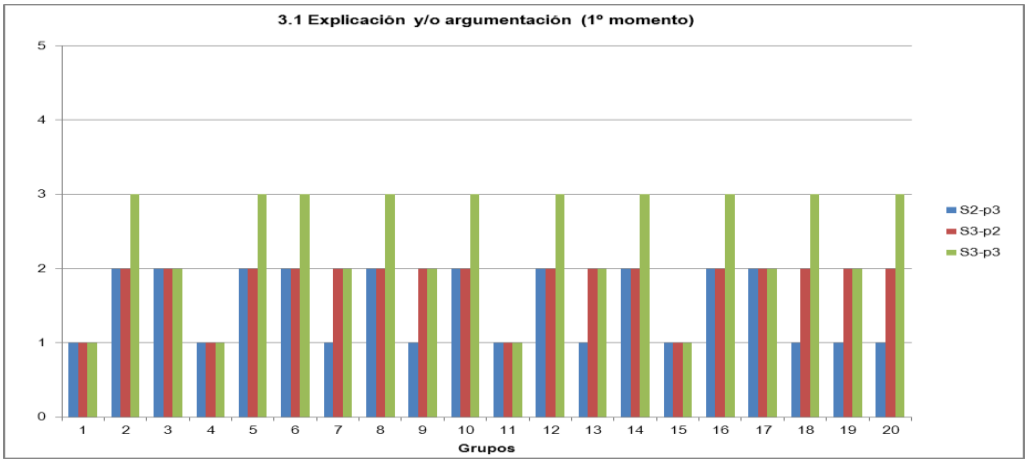

Gráfico 35: Indicador 3.1; 1er Momento

En la tercera actividad para este momento (s3-p3) en 14 de los grupos se observaron acciones como las descritas en el párrafo anterior, sin embargo en 3 grupos no apreciamos ninguna de estas acciones u otras similares.

En el segundo momento (gráfico 36 en la siguiente página) para las dos primeras actividades (s5-p1 y s5-p3), pudimos evidenciar que entre 13 y 12 grupos al menos dos de sus integrantes, intentaron explicar a sus compañeros algún procedimiento implicado en la solución, en otros casos presentaban explicaciones respecto de la idea general que se estaba usando para resolver el problema, o elaboraban argumentos para justificar la estrategia a utilizar para la resolución de un cierto problema.

Como se aprecia, se experimentó una leve disminución en la cantidad de grupos en los que observamos acciones de este tipo, respecto de lo alcanzado en la última actividad del momento anterior, y manteniéndose la cantidad y los grupos que mostraron nulas acciones de este tipo.

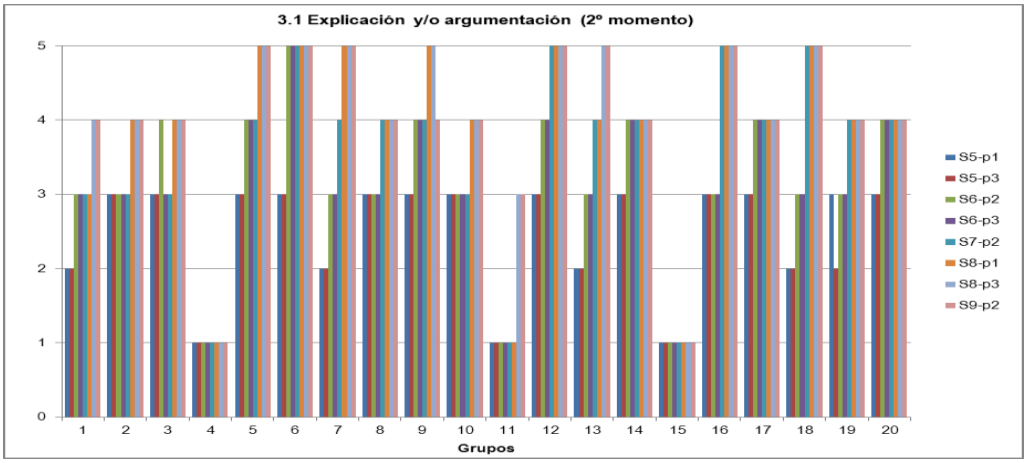

Gráfico 36: Indicador 3.1; 2º Momento

A partir de la tercera actividad de este momento (s6-p2) pudimos registrar grupos en los cuales todos sus integrantes se implicaban en acciones de explicación y argumentación de procedimientos e ideas referidas a cómo resolver los problemas.

Apreciamos también que en algunos grupos además de implicarse en este tipo de acciones todos sus integrantes, las acciones de explicitación y argumentación se dieron de manera reiterativa durante el desarrollo de la tarea cooperativa.

Esto se hizo más evidente, a partir de la quinta actividad de este momento (s7-p2) en que se observaron entre 13 y 17 grupos con estas características de comportamiento durante el desarrollo de la actividad.

Para el tercer momento (gráfico 37 en la página siguiente) los resultados discriminan entre los grupos que alcanzan buenos niveles de explicitación y argumentación, y los dos grupos para los cuales no registramos evidencias de este tipo de acciones durante el desarrollo de las tareas cooperativas de resolución de problemas.

Esta diferenciación se hace más evidente a partir de la sexta actividad de este momento (s14-p2) en que del orden de los 18 grupos se ubican en las dos más altas categoría para las acciones de este tipo, aunque se mantienen los mismos dos grupos sin mostrar manifestaciones de acciones de explicitación y/o argumentación.

Para nosotros este es un resultado importante del trabajo cooperativo, y de cómo este ambiente va generando las oportunidades para que los estudiantes vayan realizando (en este caso), acciones de explicación y/o argumentación referidas a los elementos conceptuales, procedimentales, y estratégicos que son propios de la tarea intelectual de resolver problemas de procesamiento de datos.

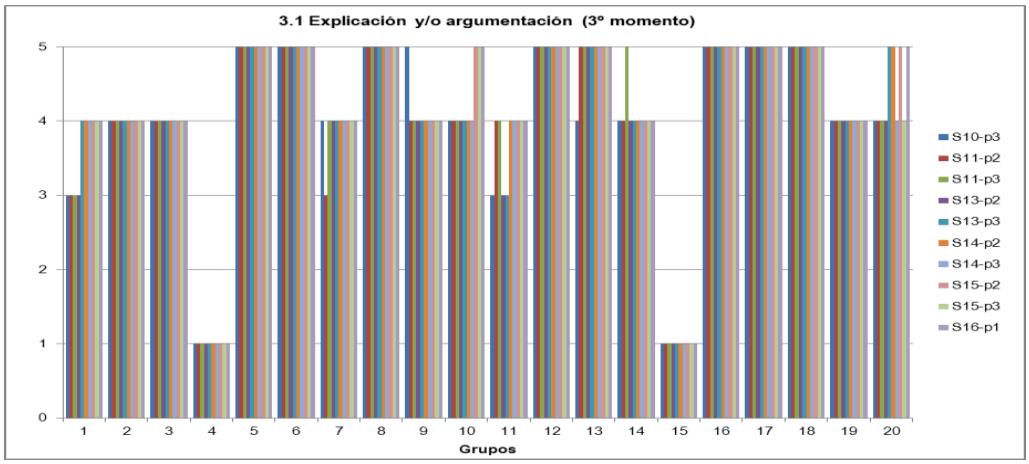

Gráfico 37: Indicador 3.1; 3er Momento

Con este tipo de acciones se da paso a la discusión y argumentación respecto de estos elementos, y de la forma en que están siendo usados en la resolución de los problemas, cuestión que hemos señalado que es fundamental para el propio aprendizaje, tanto de los conocimientos como de las capacidades implicadas en este tipo de tareas.

Así, este aspecto relevante de la construcción de significados y del aprendizaje, está siendo mediado por las actividades cooperativas, puesto que mediante la materialización de este tipo de acciones, se refleja la implicación activa de los estudiantes en la tarea intelectual de resolución de los problemas.

El segundo indicador del ámbito de construcción de significados es *Reformulación y/o síntesis* (3.2), que se refiere a las acciones de reelaboración y/o planteamiento de síntesis e ideas, contenidos o propuestas planteadas durante la interacción que se da durante la tarea cooperativa de resolución de los problemas.

En los gráficos 38, 39, y 40 se presentan los resultados de las observaciones de este indicador para los tres momentos.

Como se aprecia, para la primera actividad (s2-p3, en el gráfico 38 en la página siguiente), en 12 de los grupos no registramos acciones de reformulación y síntesis.

La situación cambió significativamente en la segunda actividad de este primer momento (s3-p2), ya evidenciamos 8 grupos en los que al menos dos de sus integrantes, en algunos casos reformularon uno o más conceptos durante el desarrollo de la tarea de resolución del problemas, en otros casos, reformularon uno o más procedimientos implicados en la solución al problema, y en otros intentaron sintetizar las propuestas de solución que se plantearon.

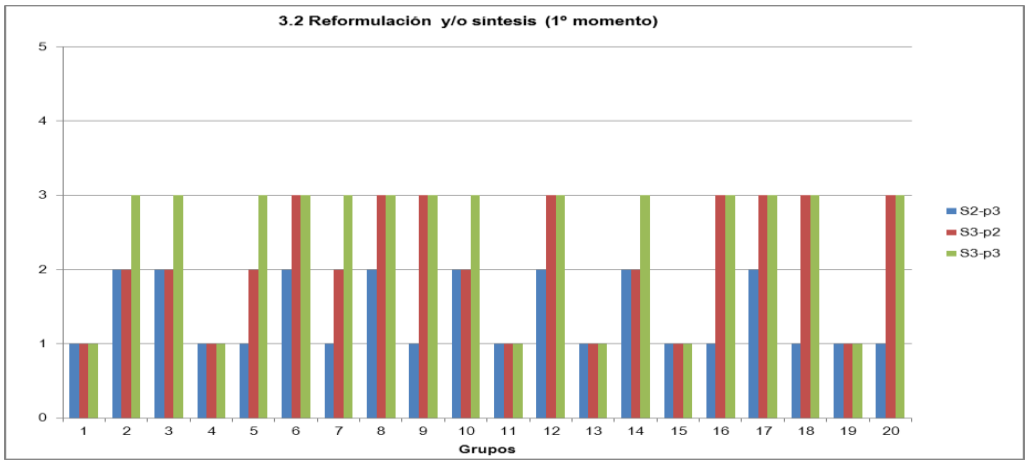

Gráfico 38: Indicador 3.2; 1er Momento

En 6 grupos, este tipo de acciones se llevó a cabo sólo por uno de sus integrantes, y apreciamos igual cantidad de grupos que continuaron sin dar muestras de este tipo de acciones durante el trabajo cooperativo.

Para la tercera actividad (s3-p3) pudimos constatar que los 6 grupos en los cuales solo uno de sus integrantes se había implicado en acciones de reformulación, ahora se implicaron dos de ellos, en tanto seguimos sin registrar acciones de este tipo en 6 grupos.

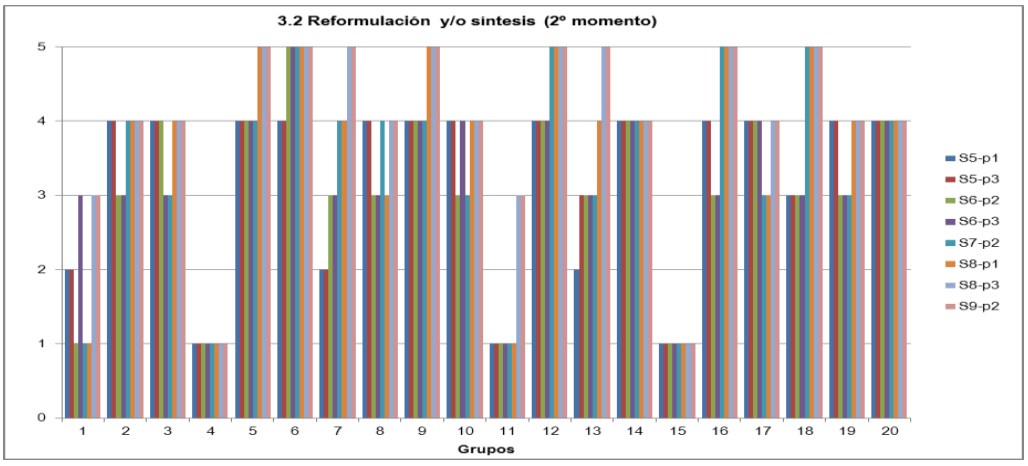

Gráfico 39: Indicador 3.2; 2º Momento

Para las actividades del segundo momento (gráfico 39), para las dos primeras actividades (s5-p1 y s5-p3), se observaron 13 grupos en los que todos sus integrantes se implicaron en acciones de reformulación y síntesis, expresadas en reformulaciones de conceptos y procedimientos, y de síntesis de ideas y propuestas de solución que se presentaban durante el proceso de resolución de los problemas. En tanto para 3 de los grupos seguimos sin observar acciones de este tipo.

En la tercera y cuarta actividades de este momento (s6-p2 y s6-p3), se aprecia una disminución en el número de grupos donde todos sus integrantes se implican en acciones de este tipo, en tanto observamos que en uno de los grupos no sólo se involucraron todos los estudiantes en acciones de este tipo, sino que además lo hicieron de manera reiterativa durante la tarea de resolución de los problemas, que se materializó en repetidas acciones de síntesis de procedimientos, de propuestas de solución, así como de síntesis de ideas respecto de las estrategias de solución que podían ser implementadas para los problemas, teniendo en cuenta además, que conforme al avance de los contenidos, en estas sesiones pedimos a los grupos resolver problemas de mayor complejidad que en los casos anteriores.

A partir de la quinta actividad, apreciamos un incremento de los grupos donde todos sus integrantes se implican en acciones de este tipo y de manera reiterativa, durante el desarrollo del proceso de resolución de los problemas. En tanto, en dos de los grupos, sus integrantes se mantienen sin mostrar acciones de reformulación y/o síntesis.

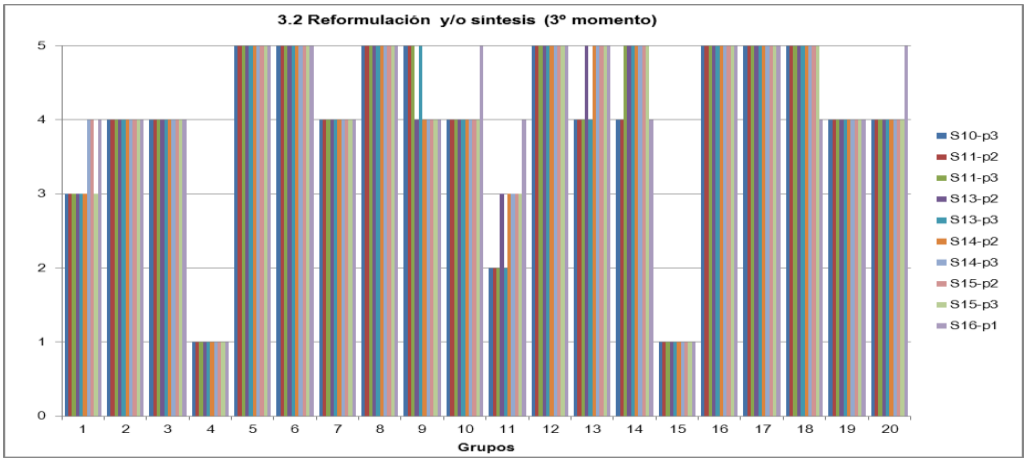

Gráfico 40: Indicador 3.2; 3er Momento

Respecto del tercer momento de actividades (en el gráfico anterior), se mantienen los dos grupos que no dan manifestaciones de acciones de este tipo, en tanto en el resto de los 18 grupos este tipo de acciones se mantienen en un buen nivel.

Este resultado nos parece interesante ya que este indicador da cuenta de una de las etapas importante de la construcción del conocimiento y de su aplicación en la propia tarea de resolución de los problemas de programación.

Las acciones individuales de reformulación y síntesis, dan cuenta del acto de reelaboración de los conocimientos que cada estudiante está llevando a cabo, y de la forma en que estos están siendo usados y relacionados.

Pero lo importante es que al estar mediado por la propia actividad cooperativa, este acto es compartido con otros, generando las condiciones para que mediante la discusión y argumentación con los otros, el resultado de la reformulación y síntesis, pueda ser enriquecido colectivamente, dando con ello lugar, a un aprendizaje de mejor calidad.

Para nosotros, este es un aspecto esencial de la construcción del conocimiento estratégico que, como hemos argumentado en varios apartados de este trabajo, es propio y absolutamente necesario para la resolución de problemas de programación.

En los gráficos 41, 42, y 43 se presentan los resultados de los niveles alcanzados por el indicador *Preguntas de contenido y/u opinión* (3.3), el que está referido a la manifestaciones de acciones de planteamiento de consultas dirigidas a la aclaración de dudas y/o interpretaciones de parecer, hacia otros miembros del grupo, sobre el problema que se resuelve, o sobre uno o más aspectos de su resolución, para los tres momentos de actividades cooperativas.

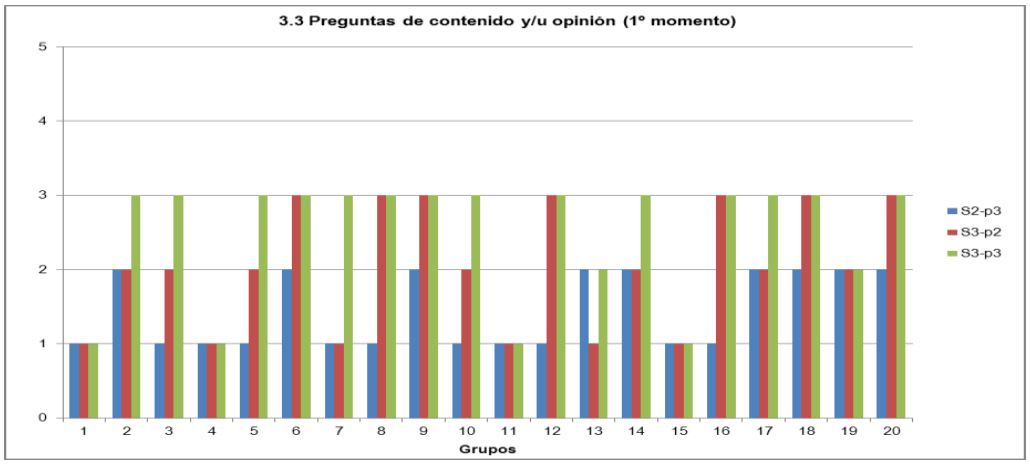

Gráfico 41: Indicador 3.3; 1er Momento

El comportamiento de este indicador para las tres actividades del primer momento (s2-p3, s3-p2, y s3-p3, en el gráfico anterior), ha sido muy similar al resto de los indicadores del ámbito de construcción de significados.

En la actividad s2-p3, en 11 grupos no se observó alguna manifestación de este indicador tales como opiniones aclaratorias o de interpretación respecto de la resolución de los problemas.

En los nueve grupos restantes se apreció que al menos uno de sus integrantes, dirigió a uno de sus compañeros de grupo, una pregunta ya sea referida a un concepto, procedimiento o idea respecto del problema en que estaban trabajando.

Para la segunda actividad (s3-p2), en 14 de los grupos se evidenció que entre uno y dos de sus integrantes realizaron preguntas como las descritas en el párrafo anterior. En los 6 grupos restantes no observamos este tipo de manifestaciones.

Para la tercera actividad de este primer momento, en 14 grupos apreciamos que a lo menos dos de sus integrantes se implicaban en preguntas de este tipo, en tanto, en 4 grupos no registramos la existencia de preguntas para aclarar conceptos, procedimientos o ideas.

Para el segundo momento (gráfico 42), en la actividad s5-p1, en 13 grupos apreciamos que a lo menos dos de sus integrantes realizaron preguntas con el objeto entender mejor un concepto, procedimiento o idea que se estaba usando para resolver el problema, entanto seguimos sin constatar la existencia de preguntas de este tipo en cuatro grupos.

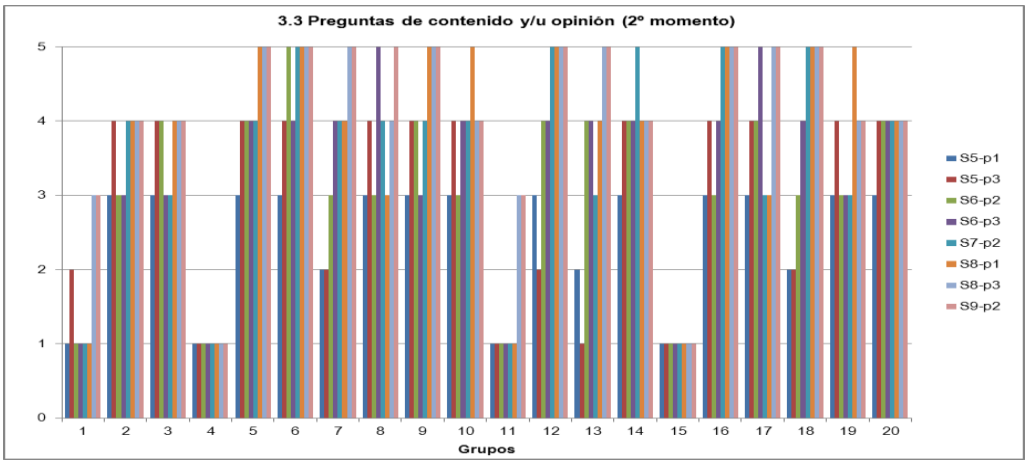

Gráfico 42: Indicador 3.3; 2º Momento

En la actividad (s5-p3) constatamos que en 12 grupos, este tipo de preguntas fueron realizadas por todos los integrantes de cada uno de estos grupos.

A partir de la tercera actividad de este momento (s6-p2) y hasta la octava y última, se apreció que en varios de los grupos, no sólo se implican todos sus integrantes en este tipo acciones, sino que además, comienzan a hacerse de manera reiterativa, en especial las referidas a la contrastación de opiniones respecto de cómo resolver los problemas. Al final de este momento aún se mantienen dos grupos sin mostrar evidencias de este tipo de acciones.

Para el tercer momento (gráfico 43 en la siguiente página), se observa que para las tres primeras actividades (s10-p3, s11-p2, y s11-p3), se mantienen los mismos dos grupos en los que no se manifiestan acciones asociadas a este indicador, y en 12 grupos observamos que durante el trabajo de resolución de los problemas, todos los integrantes hacían preguntas de contenidos, procedimientos o respecto de las ideas generales para resolver los problemas, y en varios de estos grupos, las

manifestaciones de opiniones respecto de la interpretación de estos elementos, las vimos de forma reiterativa durante las sesiones de trabajo.

Para el resto de las actividades el comportamiento para este indicador se mantiene casi de manera regular, llegando a la última actividad donde se constató que en 18 grupos se alcanzó un nivel muy alto de acciones de este tipo, en tanto se mantuvieron los dos mismos grupos para los que no registramos acciones de este tipo.

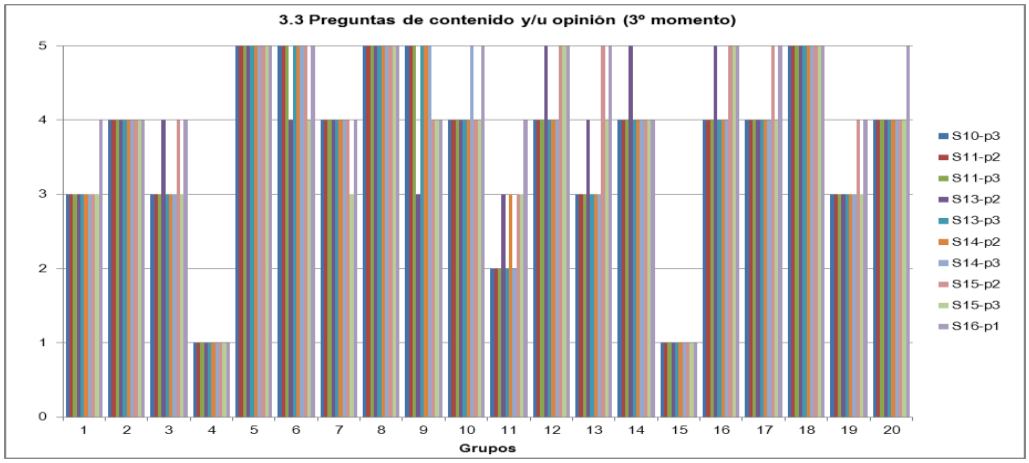

Gráfico 43: Indicador 3.3; 3er Momento

La existencia de preguntas respecto de los contenidos que se usan durante la resolución de los problemas, y el planteamiento de opiniones respecto de esto mismo, durante las interacciones del trabajo grupal, son para nosotros elementos que favorecen el aprendizaje individual y el desarrollo de las capacidades que se requieren para resolver estos problemas.

Bajo las condiciones del trabajo en equipo, la formulación de preguntas de contenidos y/u opinión, por parte de un estudiantes, generan instancias para la elaboración de las respuestas por parte de los otros, y para discutir argumentar la misma y/o sobre las opiniones planteadas, lo que sin duda favorece el aprendizaje tanto individual como colectivo.

Bajo esta perspectiva, consideramos que es beneficioso que la mayoría de los grupos hayan alcanzado buenos niveles de logro para este indicador.

En los gráficos 44, 45, y 46 se presentan los niveles de logro para el indicador, *Justificaciones* (3.4), el que está referido a las acciones de incorporación de evidencias o fundamentos que justifiquen una idea o contenido en base a la experiencia, y que ayuden a la solución de los problemas.

Para la primera actividad del primer momento (s2-p3, gráfico 44), se apreció que en 17 grupos las acciones de justificación fueron nulas, y en solo siete grupos

apreciamos que al menos uno de sus integrantes aportó con evidencias y/o fundamentos para justificar el uso ya sea de conceptos, procedimientos o ideas, como parte de la solución del problema.

Para la segunda actividad (s3-p2) se aprecia una mejora ya que sólo 7 grupos no manifiestan acciones de este tipo, sin embargo en la tercera actividad de este momento, se experimenta un empeoramiento ya que para 12 grupos no registramos evidencias de acciones de justificación, pero en 8 grupos se constató que al menos dos de sus integrantes realizaron acciones como las descritas en el párrafo anterior, a las que además se agregaron la expresión de argumentaciones basadas en la propia experiencia para defender las justificaciones que se daban.

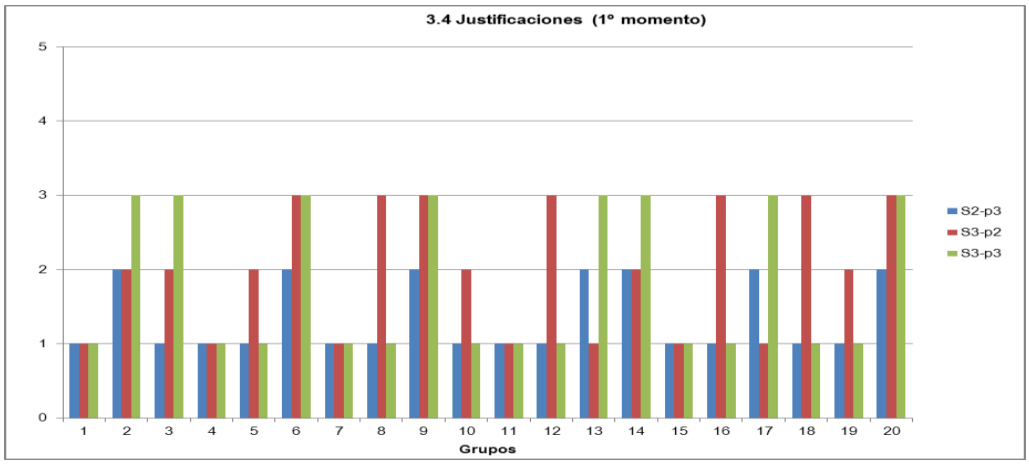

Gráfico 44: Indicador 3.4; 1er Momento

Para la primera y segunda actividades del segundo momento (s5-p1, s5-p3, en el Gráfico 45 en la página siguiente), la situación mejora significativamente ya que en 13 grupos se observó que la totalidad de sus integrantes se implicaban en acciones de justificación como las descritas, sin embargo se experimenta un retroceso en la siguiente actividad (s6-p2), aunque para esta observamos un grupo donde además de implicarse todos los integrantes en estas acciones, lo hicieron de manera reiterativa durante el desarrollo de la tarea cooperativa de resolución de los problemas, incluso aportando elementos desde su propia experiencia.

A partir de las siguientes actividades de aprecia una mejora en el indicador en el sentido de que más grupos alcanzan un muy buen nivel de acciones para este indicador, es decir, se incrementa el número de grupos donde todos sus integrantes realizan acciones como las que hemos señalado, e incluso en algunos grupos este tipo de acciones se observa de manera reiterativa.

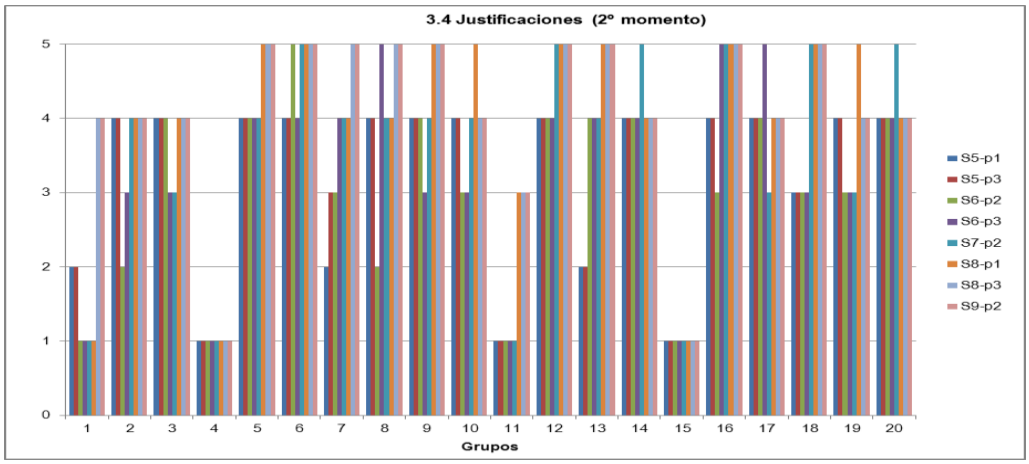

Gráfico 45: Indicador 3.4; 2º Momento

Para la séptima y octava actividades de este momento (s8-p3 y s9-p2) aún se mantienen dos grupos sin manifestaciones de acciones de este tipo, y registramos 17 grupos con muy buenas acciones como las que hemos señalado.

Para las tres primeras actividades del tercer momento (gráfico 46), se vuelven a tener entre uno y tres grupos en los cuales sólo uno de sus integrantes aportan evidencias y/o fundamentos para justificar el uso de conceptos, procedimientos o ideas respecto de la resolución del problema, y registramos entre 9 y 11 grupos con buenos niveles, aunque se mantenían los mismos dos grupos en que no se apreciaron acciones como las que hemos señalado.

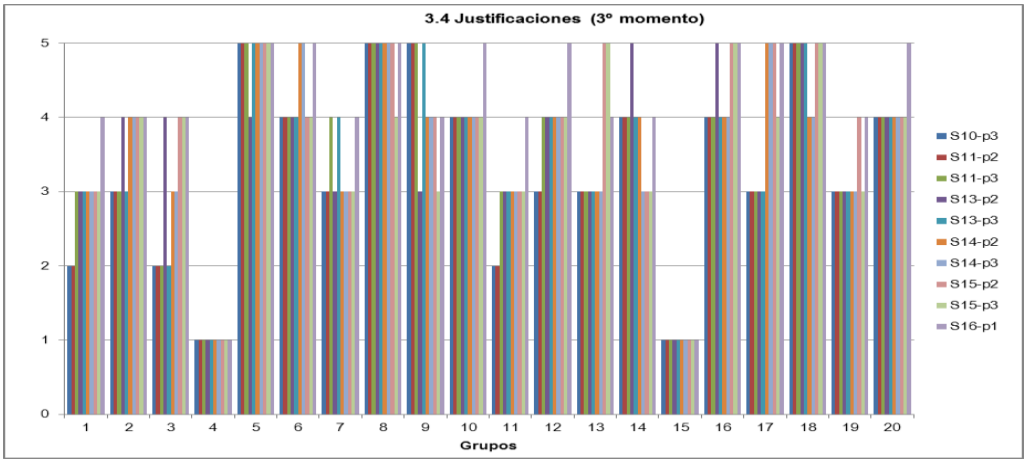

Gráfico 46: Indicador 3.4; 3er Momento

A partir de la sexta actividad de este bloque, se observaron al menos 12 grupos donde se dio un muy buen nivel de acciones de justificaciones, aunque continuábamos teniendo dos grupos en los que no registramos ninguna acción de este tipo.

Al terminar este tercer momento, registramos que en 18 grupos, todos sus integrantes se implicaron en acciones de justificación de uno o más conceptos, procedimientos, o ideas que estaban siendo usadas en la resolución de los problemas, y en muchos casos, con reiteraciones de acciones de este tipo durante el desarrollo de la tarea cooperativa, aunque se mantuvieron los dos grupos que no manifestaron acciones de este tipo durante el desarrollo de la tarea cooperativa.

El plantear justificaciones respecto de porqué se aplican determinados conocimientos y/o estrategias de solución, así como también respecto de porqué se aplican de una cierta manera, es otro de los elementos esenciales del proceso de aprendizaje de esta asignatura. Como hemos argumentado en apartados anteriores, la resolución de un problema de programación, no es la mera aplicación de un conjunto de conocimientos, o la utilización simple de modelos de problemas ya resueltos, sino que demanda una la selección y organización de conocimientos que es particular para el problema que se quiere resolver, contexto en el cual la elaboración y defensa de justificaciones de porqué se están usando determinados conocimientos, y de porqué se organizan de una cierta manera, es una cuestión fundamental del diseño e implementación de las soluciones, lo que en términos del aprendizaje, intentamos que individualmente se potencie mediante el trabajo en equipo.

El planteamiento de justificaciones, obliga también a elaborar y reelaborar elementos, lo que también podría contribuir al aprendizaje personal respecto de la resolución de los problemas, y de paso, ayuda a que los demás integrantes del grupo, mejoren la comprensión de los elementos que intervienen en la solución y de cómo se aplican y utilizan dichos elementos.

En los gráficos 47, 48, y 49, se presenta la evolución para los tres momentos del indicador *Planteamiento de discrepancias* (3.5), el que está referido a la manifestación de acciones de desacuerdo con ideas, contenidos o propuestas para la resolución de los problemas.

Como se observa en el Gráfico 48 de la página siguiente, en la primera actividad (s2-p3) al igual que para otros indicadores, registramos un número importante de grupos (14) en los que no se apreciaron acciones de planteamiento de discrepancia, y en el resto del grupo sólo uno de sus integrantes planteo discrepancias respecto del uso de uno o más contenidos, procedimientos o ideas respecto del problema que se estaba resolviendo.

Para la segunda actividad (s3-p2), en 10 grupos, se verificó este tipo de acciones en uno solo de sus integrantes, en tanto en 8 de ellos, se implicaron a lo menos dos estudiantes en acciones de este tipo durante el trabajo de resolución del problema.

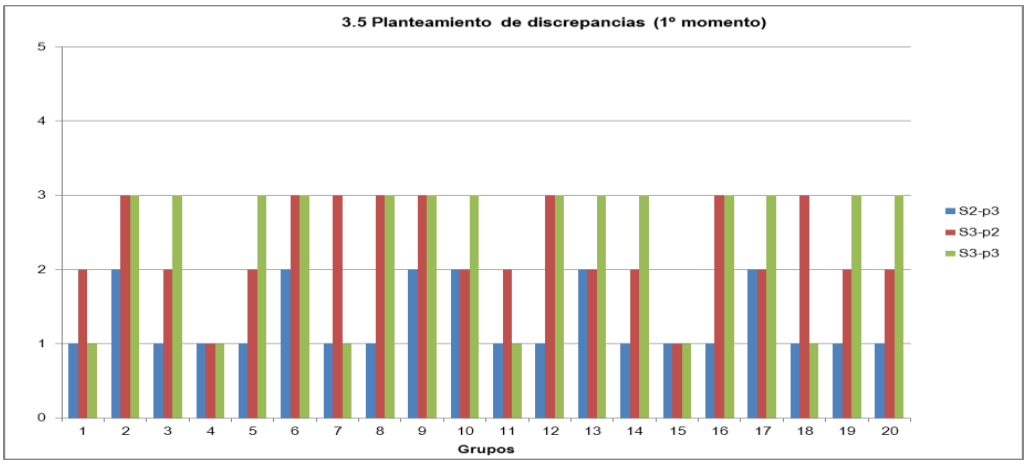

Gráfico 47: Indicador 3.5; 1er Momento

Para el caso de la tercera y última actividad de este momento, se incrementa el número de grupos en los que solo uno de los integrante toma parte de este tipo de acciones (6 grupos) y en 14 de ellos se implicaron a lo menos dos estudiantes de cada grupo en acciones como las descritas.

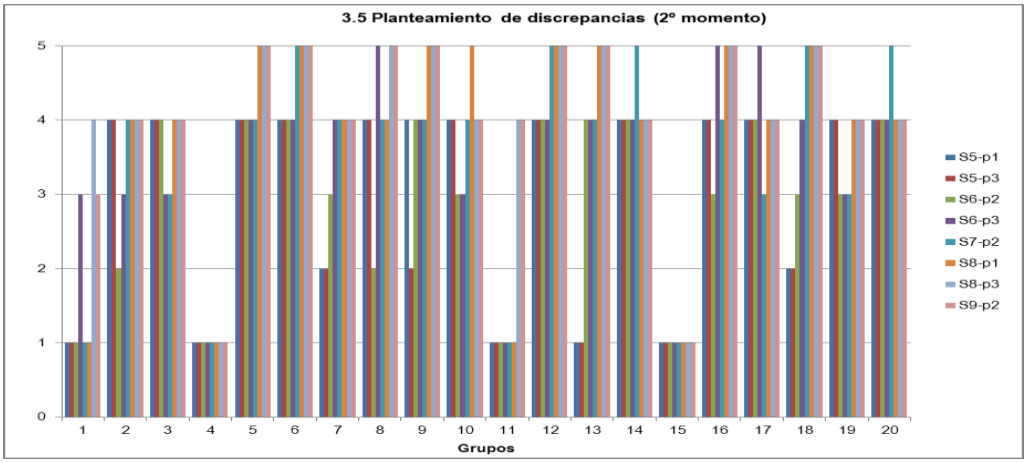

Gráfico 48: Indicador 3.5; 2º Momento

Para las actividades del segundo momento, en las dos primera (s5-p1 y s5-p3, en el gráfico anterior), se constató que en a lo menos 12 grupos todos sus integrantes plantearon discrepancias respecto de si era correcto usar, ya sea unos determinados conceptos, procedimientos o ideas para resolver el problema, en tanto en 5 grupos no se apreciaron acciones de este tipo durante el desarrollo de la tarea grupal.

En la tercera actividad de este momento (s6-p2), disminuye a 9 la cantidad de grupos en que se implican todos los integrantes del grupo en acciones de este tipo.

En las siguientes actividades comienza a incrementarse el número de grupos en que este tipo de acciones las realizan todos los integrantes, como así también los grupos en que estas acciones se dan de manera reiterativa durante el trabajo de resolución

de los problemas, en tanto se mantienen dos grupos donde el planteamiento de discrepancias en sus distintas formas no lo apreciamos.

En las 4 primeras actividades de las 10 del tercer momento (gráfico 49), se constató una distribución regular de los grupos en cada nivel de manifestación de este indicador, manteniéndose los mismos dos grupos que no manifiestan acciones de discrepancia durante el trabajo de resolución de los problemas.

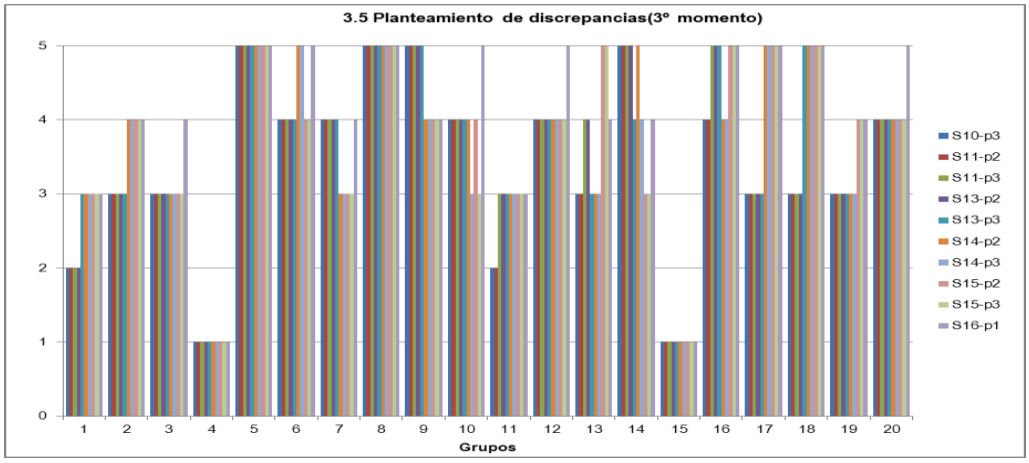

Gráfico 49: Indicador 3.5; 3er Momento

A partir de la quinta actividad de este momento se observa un incremento paulatino de los grupos en que todos sus integrantes plantean discrepancia respecto del uso y/o aplicabilidad de determinados conceptos, procedimientos o ideas para resolver un cierto problema, y grupos en los que este tipo de acciones se da de manera reiterativa.

En la última actividad, registramos 18 grupos en los que se alcanza un alto nivel de planteamiento reiterativo de discrepancia respecto del uso de uno o más conceptos, procedimientos y/o ideas, durante el desarrollo de la tarea de resolución de problema cooperativo, y de mantuvieron los dos grupos para los cuales no evidenciamos acciones de este tipo, ya que trabajaban en la tareas de resolución de problemas de manera individual.

Para nosotros, este es otro tipo de acción fundamental para el aprendizaje individual de los estudiantes, ya que es muestra de las instancias y el nivel de discusión que se da al interior de los grupos, de cómo unos tienen posturas distintas de los otros, respecto de los elementos implicados en la resolución de los problemas y de la forma de resolverlos, pero también, este tipo de acciones da origen al establecimiento de consensos (negociaciones) respecto de cómo llevar adelante la propia tarea de resolución y de los elementos implicados en ella, cuestiones que creemos que favorecen el aprendizaje de los estudiantes.

El planteamiento de discrepancias puede dar paso a acciones de reelaboración y reformulación de contenidos, ideas y estrategias, tanto a nivel individual como colectivo, lo que creemos que influye positivamente en el aprendizaje de nuestros estudiantes.

En los gráficos 50, 51, y 52 se muestran los resultados para el indicador *Aclaración y/o complementación contenidos* (3.6), el que está referido a las manifestaciones de acciones dirigidas a clarificar y/o completar una idea en relación a un contenido o propuesta de resolución de un problema, para los tres momentos de actividades cooperativas.

Para la primera actividad del primer momento (s2-p3, en el gráfico 50), apreciamos 14 grupos en los que no se manifestaron acciones para este indicador, y 6 en los que observamos que al menos uno de sus integrantes realizó aclaraciones y/o complementaciones de contenidos al interior de sus grupos.

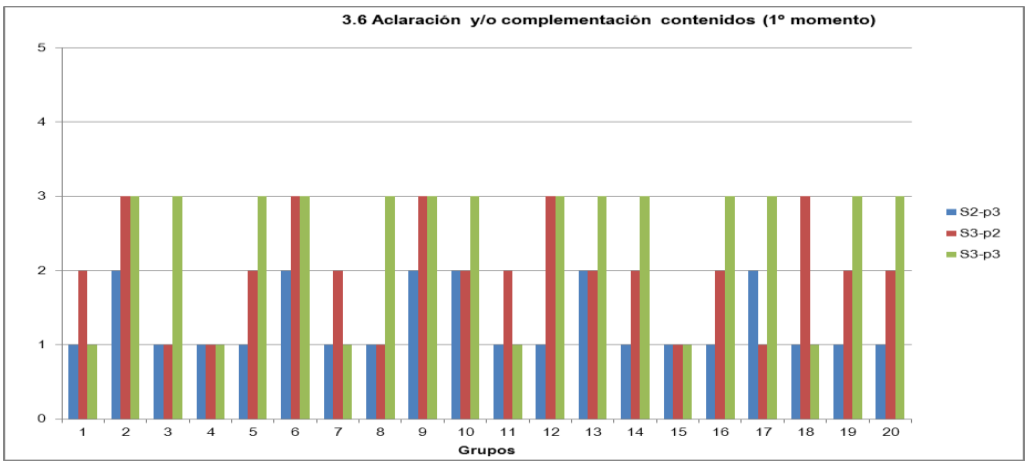

Gráfico 50: Indicador 3.6; 1er Momento

Para la segunda actividad de este momento (s3-p2), 5 grupos se mantenían sin mostrar evidencias de acciones para este indicador, en 10 de ellos se apreció que al menos un integrante realizó acciones como las descritas, y en el resto de los grupos observamos que al menos dos de sus integrantes se implicaron en acciones de este tipo.

La situación mejora para la tercera actividad del mismo momento (s3-p3), donde registramos 14 grupos en la condición anterior, y 6 en los que no tenemos evidencias de manifestaciones de este tipo o similares.

Para la primera actividad del segundo momento (s5-p1, en el gráfico 51 en la siguiente página), en 13 de los grupos se observó que todos los integrantes de estos grupos se involucraron en acciones dirigidas a aclarar uno o más contenidos y en algunos casos, en acciones tendientes a complementar ya sea contenidos o ideas o

propuestas para resolver los problemas en que trabajaban, y en 4 grupos no observamos ningún tipo de estas acciones.

Para la segunda y tercera actividades de este mismo momento (s5-p3 y s6-p2), se apreció una leve baja en los indicadores, que como hemos señalado, creemos que se debe fundamentalmente a que en esas actividades se plantearon problemas de mayor complejidad.

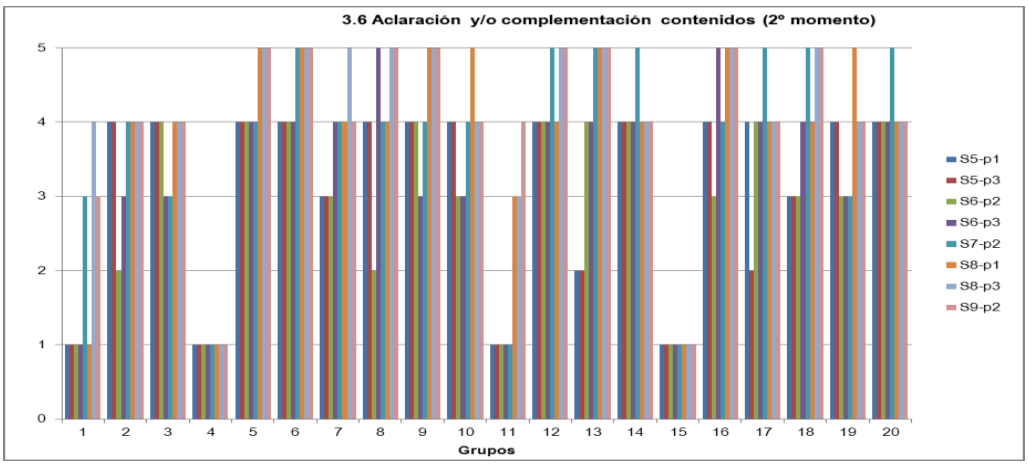

Gráfico 51: Indicador 3.6; 2º Momento

A partir de la cuarta actividad (s6-p3) se observa un incremento de los niveles de logro, y empiezan a observarse grupos en que este tipo de acciones se dieron de manera reiterada al interior de los grupos, y aún se mantenían algunos grupos en los cuales no observamos acciones como las señaladas.

Para las actividades del tercer momento, entre la primera y la quinta (s10-p3 hasta s13-p3 en el gráfico 52 en la página siguiente), se encuentran grupos en los distintos niveles de categorías para este indicador. Cabe hacer notar que para estas actividades también se trabajó con problemas de un mayor grado de complejidad, especialmente por los conocimientos implicados en la solución de los mismos.

A partir de la sexta actividad y hasta el final, se apreciaron mejoras, que determinan que en 18 de los grupos se constató que todos sus integrantes se implicaron en acciones como las señaladas, y en algunos de ellos este tipo de acciones se produjeron de manera reiterativa, aunque se mantuvieron dos grupos en los que no se manifestaron acciones de este tipo.

Las acciones de aclaración y/o complementación de contenidos es otro de los indicadores que nos parecen relevantes como acciones que deben darse durante el trabajo cooperativo de resolución de estos problemas, ya que como hemos señalado,

en la resolución de estos problemas, además del manejo de los contenidos, se requiere de la capacidad de usarlos correctamente dentro del contexto de la

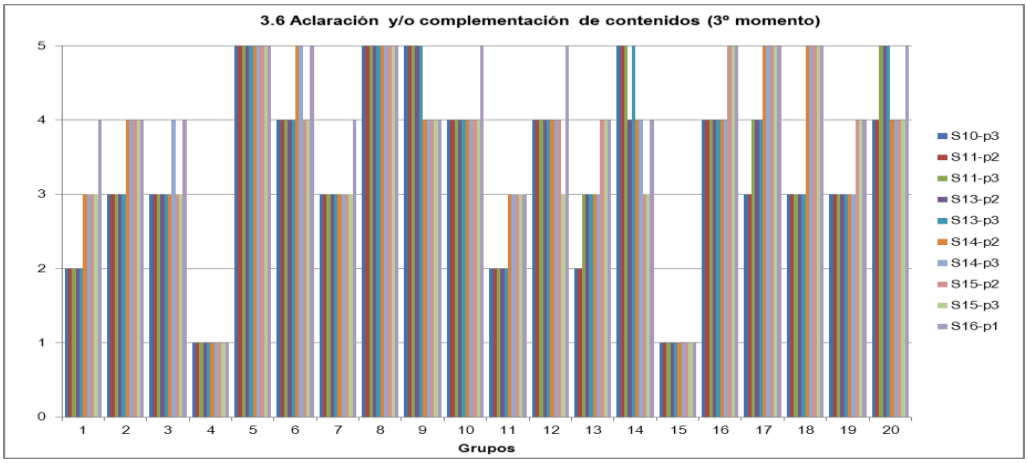

Gráfico 52: Indicador 3.6; 3er Momento

resolución de un determinado problema.

Por otra parte, conforme a lo observado, las acciones de aclaración y/o complementación son gatilladas por el planteamiento de discrepancia al interior de los grupos (indicador 3.5), lo que refleja que los integrantes de los grupos se hacen cargo de las diferencias de opinión, formulando aclaraciones y complementaciones de contenidos e ideas referidas a la resolución de un determinado problema.

Nuevamente estas son acciones que tanto individual como colectivamente, demandan la reelaboración de conocimientos, ideas y estrategias, lo que esperamos contribuya a alcanzar mejores aprendizajes.

En los gráficos 53, 54, y 55 se muestra la evolución del indicador *Explicitación de la tarea* (3.7), que se refiere a la manifestaciones de acciones dirigidas a explicar una tarea con el objeto de alcanzar un mejor comprensión colectiva de la misma, para los tres momentos de actividades cooperativas.

Para la primera actividad (s2-p3) del primer momento (gráfico 53 en la siguiente página), se evidenció que en 15 grupos no existieron acciones de explicitación de la tarea a desarrollar, y en sólo 5 grupos se observó que uno de sus integrantes intentó explicar ya sea a otro de los integrantes, o al grupo en general, los detalles de la tarea, con la intención de hacerla más comprensible.

Para la segunda actividad (s3-p2) 10 de los grupos mantenían la condición de nula explicitación, en tanto se constató que ahora en 5 de ellos ya se involucraron hasta dos integrantes de cada grupo en este tipo de explicaciones.

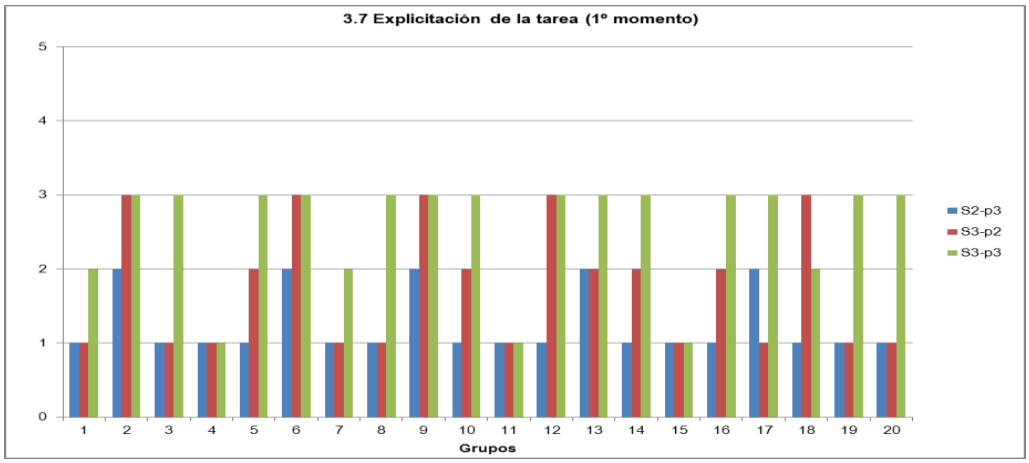

Gráfico 53: Indicador 3.7; 1er Momento

Para la tercera y última actividad de este primer momento (s3-p3), apreciamos que ya en 14 de los grupos, a lo menos dos estudiantes realizaran acciones de este tipo al interior de los grupos.

Para las dos primeras actividades del segundo momento (s5-p1 y s5-p3 en el gráfico 54) se observó que en dos grupos sus integrantes aún se mantenían si realizar acciones de este tipo, en 13 de ellos se implicaba en acciones de explicación de la tarea solo un alumno en cada grupo, y en 5 grupos, existieron 2 estudiantes que se involucraron en acciones de este tipo.

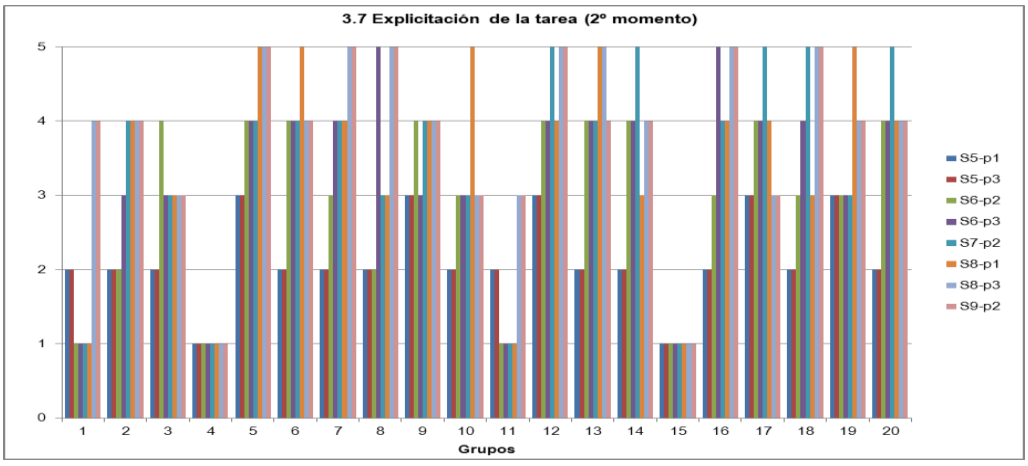

Gráfico 54: Indicador 3.7; 2º Momento

Para la tercera actividad de este segundo momento (s6-p2), se apreció que en 9 grupos ya se implicaban todos los estudiantes del grupo en acciones de explicación de la tarea, incluso en algunos casos presentando ejemplos se problemas conocido para hacer más clara y cercana la explicación de la tarea, en tanto aún manteníamos 4 grupos en los que este tipo de acciones no las registramos.

A partir de la cuarta actividad (s6-p3) se comienza a incrementar el número de grupos en los que se observaron que este tipo de acciones se dio en varios momentos del trabajo de resolución del problema, instancias en las cuales todos los integrantes trataban de explicitar los pasos de la tarea al resto, y se mantenían dos grupos en los que no se constataron acciones de explicitación en ningún momento del trabajo colaborativo.

Para el tercer momento, entre la primera actividad y la quinta (s10-p3 hasta s13-p3, en el gráfico 55), para entre 8 y 10 grupos, pudimos seguir observando que todos sus integrantes se implicaban en acciones como las señaladas, y con reiteraciones de

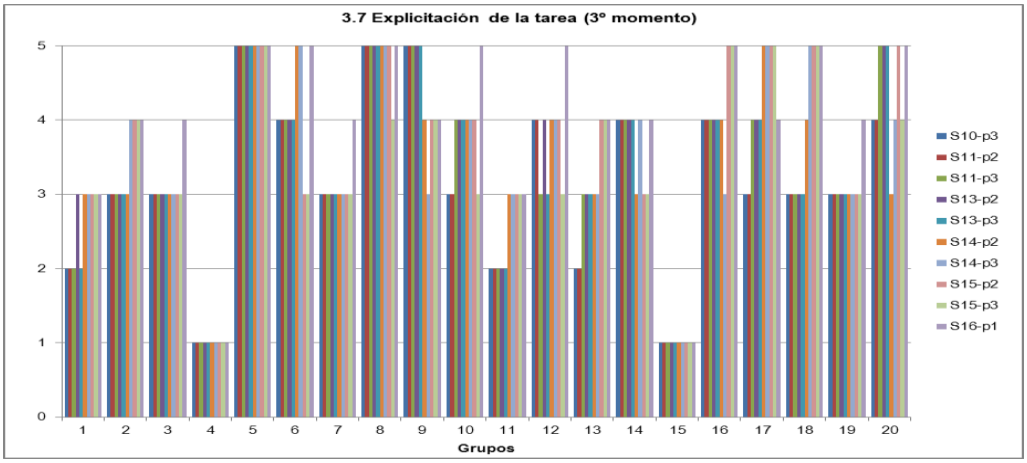

Gráfico 55: Indicador 3.7; 3er Momento

las misma, además se mantuvieron los dos grupos que no mostraron acciones en que alguno de los integrantes intentara explicar detalles de la tarea o parte de ella.

Desde la sexta actividad hasta la última (s14-p2 hasta s16-p1) se registra un incremento de los grupos en que todos los integrantes tomaban parte de acciones de explicación de la tarea o partes de ella, llegando a la última actividad con 16 grupos en esta condición, y se mantuvieron los dos mismos grupos en que sus integrantes no realizan este tipo de acciones.

Las acciones de explicitación de la tarea, representan para nosotros, una muestra de la calidad del trabajo grupal durante la resolución de los problemas. En la medida que estas acciones se dan en el trabajo grupal, se están socializando las estrategias y el proceso de resolución, generando instancias para discutirlas y consensuarlas, lo que creemos que favorece el hecho de que todos los integrantes del grupo consigan un mejor aprendizaje y comprensión de lo que se hace.

En los gráficos 56, 57, y 58 se observa la evolución del indicador *Calidad de la tarea de resolución de los problemas* (3.8), que está referido a la estimación de que tan bien han sido resueltos los problemas. Cada gráfico muestra las actividades cooperativas de un momento particular.

Como se muestra en el gráfico 56, para la primera actividad del primer momento (s2-p3) en 17 de los grupos la calidad de las soluciones fue absolutamente deficiente, lo que significa que para estos grupos nada de lo que desarrollaron como solución al problema propuesto, pudo ser considerado como parte de la solución, o que al menos contribuyera a la solución.

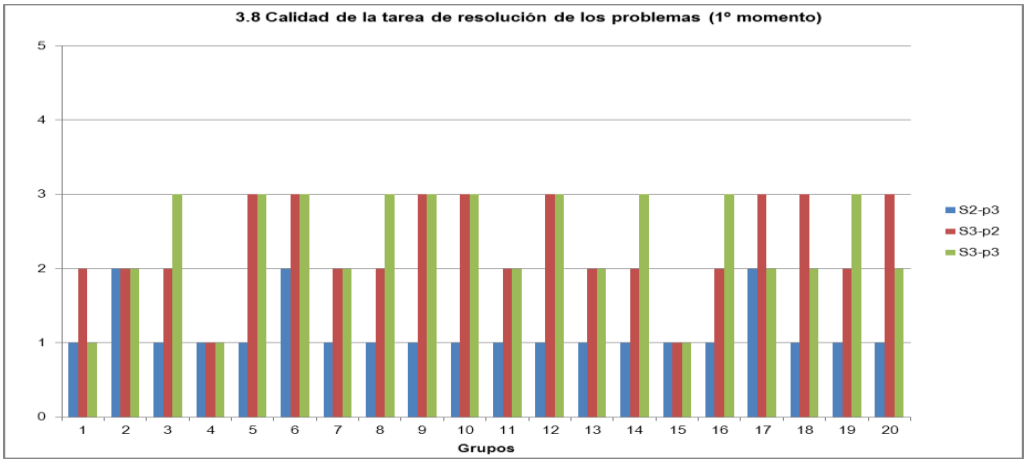

Gráfico 56: Indicador 3.8; 1er Momento

Solo en tres de los grupos pudimos observar que si bien las soluciones se presentaban de manera ordenada, se apreciaban errores tales como el uso incorrecto de conceptos y/o procedimientos, y/o con evidencias de que la metodología no fue correctamente aplicada, y/o que el lenguaje algorítmico no estaba bien utilizado, entre otros errores.

Para la segunda actividad de este primer momento (s3-p2), en las soluciones de 10 grupos pudimos constatar la existencia de uno o más errores como los señalados en el párrafo anterior, además si bien no constituían soluciones completas al problema planteado, contaban con varios de los elementos esenciales y fundamentales de lo que debería ser la solución.

Para la tercera y última actividad de este primer momento (s3-p3), en 10 de los grupos pudimos apreciar soluciones de calidad media con solo unos pocos errores como el uso incorrecto de una estructura algorítmica, o errores en el uso del lenguaje algorítmico, y en 7 grupos, soluciones que tenían errores más graves como por ejemplo el mal uso de un concepto, y 3 grupos cuyas soluciones eran definitivamente malas.

Estos resultados deficientes en la calidad de las soluciones para el primer momento de actividades, se reflejó también en los resultados de la primera evaluación, en la cual el grupo experimental obtuvo resultados por debajo de los logrados por el grupo control.

Para el segundo momento de actividades (gráfico 57) en la primera actividad (s5-p1), en 13 de los grupos se apreciaron soluciones que tenían uno o varios errores menores, generalmente de formas declarativas o de sintaxis, en tanto las soluciones entregadas por 4 grupos estaban definitivamente malas.

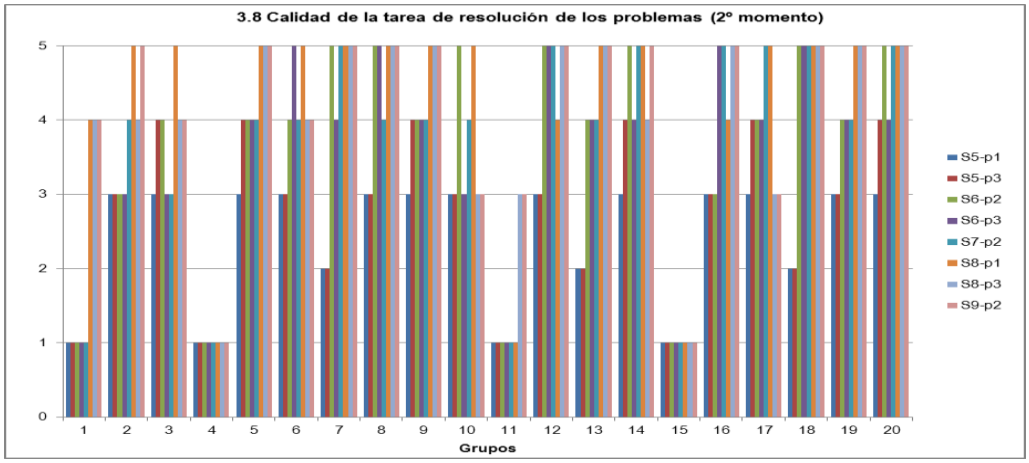

Gráfico 57: Indicador 3.8; 2º Momento

Para la segunda actividad de este momento (s5-p3), 6 de los grupos alcanzaron soluciones completas y ejecutables para el problema planteado, y en 6 grupos, soluciones de muy buena calidad, en el sentido de que no solo eran soluciones en las que se hacía un uso correcto de la metodología, las estructuras algorítmicas, de las sentencias del lenguaje, entre otros aspectos, sino que además, se verificaba que eran soluciones eficaces para el problema propuesto. Las soluciones para 4 de los grupos eran definitivamente malas.

A partir de la tercera actividad de este momento (s6-p2 en adelante), se constataron grupos que consiguen desarrollar soluciones de muy alta calidad, esto es, soluciones que no solo reflejan una correcta aplicación de los conocimientos, los fundamentos y las estrategias de solución, sino que además son soluciones optimizadas, que poseen un muy buen diseño.

Para la tercera actividad, 7 grupos alcanzan niveles de soluciones que reflejan un uso apropiado de los conceptos, las estructuras de programación y las estrategias de solución, y la misma cantidad de grupos alcanza un nivel de soluciones que consideramos de muy alta calidad, ya plantean soluciones inteligentes y novedosas para los problemas, donde además se refleja la intención por optimizar las mismas. En esta actividad tenemos cuatro grupos con un nivel absolutamente deficiente en sus soluciones. Son soluciones que tienen errores importantes y que además no pueden ser ejecutadas en el computador.

Estas condiciones mejoran más hacia las últimas actividades de este momento, de tal forma que en la octava actividad (s9-p2), 12 grupos desarrollan soluciones de muy

buena calidad, y se mantenían 2 grupos con soluciones que no contenían nada rescatable en ellas.

Esto que consideramos un buen nivel de logro en la calidad de las soluciones alcanzadas por los grupos cooperativos, se vio manifestada en los resultados de la segunda evaluación acumulativa, donde claramente, el grupo experimental obtuvo mejores resultados que el grupo control.

Esta tendencia a la mejora en la calidad de las soluciones se observó también en el tercer momento de actividades (gráfico 58). Como se aprecia, para la mayoría de las 10 actividades, se tienen 18 grupos que consiguen desarrollar soluciones que categorizamos entre buenas y optimas, lo cual se ve nuevamente reafirmado, por los buenos resultados en la tercera evaluación acumulativa para el grupo experimental.

Esto resultados son relevantes para nosotros, ya que con ellos intentamos cualificar

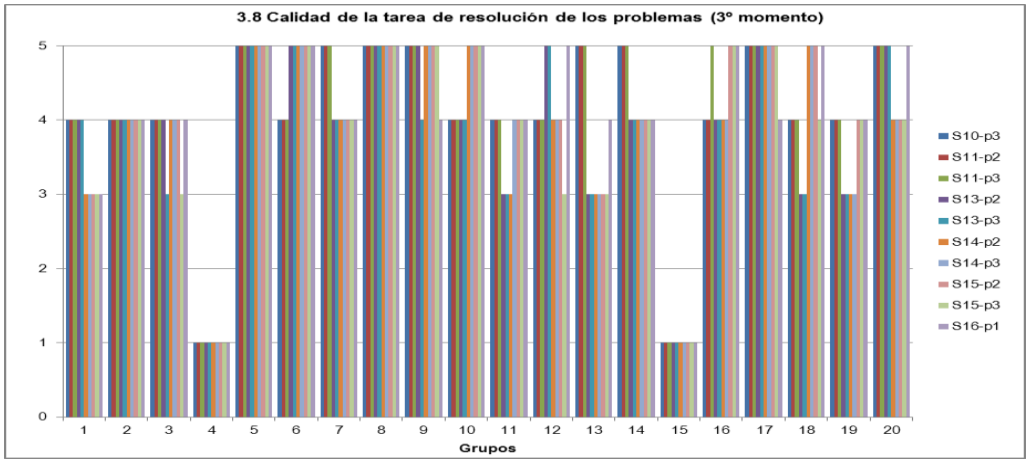

Gráfico 58: Indicador 3.8; 3er Momento

el producto del aprendizaje, es decir la calidad del proceso de manejar los conocimientos y de aplicarlos, así como también, el nivel de aprendizaje y desarrollo de las capacidades para resolver los problemas de procesamiento de datos, que es el propósito ultimo de esta forma de intervención.

Dado que este indicador nos da evidencias respecto de los resultados de la tarea específica de resolución de los problemas, nos provee de los antecedentes respecto de la calidad de esta tarea, así como, de los elementos conceptuales, procedimentales y estratégicos que los estudiantes no están consiguiendo manejar adecuadamente, lo que nos ha permitido realizar los ajustes que permitan mejorar estos aspecto.

Teniendo en cuenta que este indicador nos enI27trega información de la calidad de las tareas de resolución en el ámbito colectivo (de cada grupo), al relacionar estas las valoraciones con las de los otros indicadores, y con los resultados de las pruebas

individuales, nos permite cualificar los buenos efectos que han tenido las actividades cooperativas sobre el aprendizaje individual de los estudiantes

En síntesis, conforme al modelo de intervención que hemos puesto en práctica, creemos que existe concordancia entre los niveles alcanzados por los grupos para el conjunto de estos indicadores, y los resultados alcanzados en el rendimiento de las pruebas individuales para el grupo experimental.

A la luz de los resultados de la evolución de los indicadores, podemos concluir que la incorporación de un número importante de actividades cooperativas en el plan de clases de la asignatura, contribuyó a que la generalidad de los estudiantes, aprendiera y desarrollara habilidades para el trabajo en equipos.

Sin embargo consideramos que la evolución de indicadores como el 1.4 (Preguntas de organización o método de trabajo), 1.5 (Aclaración y/o complementación de aspectos de organización del trabajo), 3.1 (Explicación y argumentación), 3.2 (Reformulación y/o síntesis), 3.3 (Preguntas de contenido y/u opinión), 3.4 (Justificaciones), 3.5 (Planteamiento de discrepancias), y 3.6 (Aclaración y/o complementación de contenidos), han favorecido de manera importante el aprendizaje de la programación de computadores en nuestros estudiantes.

Consideramos también que este tipo de prácticas, les han permitido la participación de ambientes de aprendizaje más motivadores y enriquecedores, donde además, se han sentido apoyados por sus pares durante el proceso de aprendizaje, lo que ha contribuido a generar una sensación de aprendizaje más confortable, en comparación a lo que ocurre cuando esta asignatura se conduce sólo mediante clases de tipo expositivas, lo que ha quedado reflejado en los resultados de la encuesta de satisfacción que se aplicó a los estudiantes del grupo experimental.

6.6. Resultados referidos a la encuesta de satisfacción

La encuesta de satisfacción consta de 24 ítems que fueron presentados a los estudiantes de los grupos experimentales para ser calificados en base a una escala Likert que considera 5 valorizaciones posibles (Totalmente en desacuerdo, En desacuerdo, No tiene opinión, De acuerdo, Totalmente de acuerdo)

La fiabilidad de los ítems de la encuesta para la combinación de los resultados de las dos intervenciones se estimó mediante el cálculo de Chi-cuadrado de Cochran arrojando un alfa de 0,9692.

En la siguiente tabla se presentan los estadísticos descriptivos para la combinación de los resultados de la encuesta.

Ítems	Media	Desv. típ.
Siente que las actividades de trabajo cooperativo han contribuido a que Usted se comprometa de manera más activa en su proceso de aprendizaje.	3.95	0.990
En el ámbito de su grupo de trabajo cooperativo, considera que este tipo de actividades han contribuido al logro de un mayor compromiso colectivo, en el aprendizaje.	4.20	0.833
Considera Usted que como consecuencia de la interacción entre los integrantes del grupo, se ha favorecido el aprendizaje individual.	3.92	1.201
Usted cree que con sus aportes en el trabajo de grupo, contribuyó al aprendizaje de sus compañeros de grupo.	4.08	0.971
Siente que esta forma de trabajo grupal, le ayudó a disminuir los niveles de ansiedad ante las posibilidades de fracaso académico.	3.97	1.125
Considera que esta forma de conducción de la asignatura, contribuyó a que Usted si sintiera más seguro en su desempeño.	3.98	0.904
Cree que esta forma de conducción de la asignatura le ayudó a disminuir su sensación de aislamiento frente al trabajo académico.	3.97	0.966
Cree que la asignación de roles y tareas al interior del grupo, favorece el compromiso de los integrantes del grupo.	3.92	0.988
Cree que el trabajo grupal fomentó en Usted un mayor grado de responsabilidad hacia el logro de los objetivos de aprendizaje.	3.84	1.052
Siente que esta forma de aprendizaje le ha alentado a desarrollar un trabajo más independiente.	3.98	0.922
Siente que el aprendizaje cooperativo le ha servido para desarrollar su capacidad para razonar de manera crítica.	4.26	0.947
Siente que las actividades de trabajo cooperativo le han ayudado a mejorar su habilidad para elaborar argumentos con claridad.	4.33	0.926
Considera que las actividades de trabajo cooperativo le han permitido mejorar su capacidad de expresión oral.	4.00	0.894
Siente que las actividades de trabajo cooperativo, permiten crear ambientes de aprendizaje más agradables.	4.33	0.870
Siente que las actividades de trabajo cooperativo le han permitido tener una actitud más positiva hacia el aprendizaje de los contenidos.	3.93	0.892
Siente que los ambientes de trabajo cooperativo se han acomodado mejor a sus formas de aprendizaje.	4.11	0.877
Cree que las actividades de trabajo cooperativo han contribuido a mejorar sus logros en el rendimiento académico.	3.92	1.038
Cree que los aprendizajes logrados mediante el desarrollo de las actividades de trabajo cooperativo, han sido gravitantes en su éxito académico.	4.18	1.025
Cree que las actividades de trabajo cooperativo contribuyen de manera importante al desarrollo de las capacidades de liderazgo.	4.10	1.012
Cree que las actividades de trabajo cooperativo contribuyen de manera importante al desarrollo de las capacidades para el trabajo en equipo.	4.08	0.781

Tabla 92: Resultados de la encuesta de satisfacción

En síntesis, los resultados de la encuesta para el conjunto de estudiantes que participaron de la intervención muestran que:

El 57.38% de ellos están de acuerdo en que las actividades cooperativas hicieron que se comprometieran de forma más activa en el proceso de aprendizaje. Un 26,23% están totalmente de acuerdo en esto mismo, en tanto el 4.92% de los estudiantes están totalmente en desacuerdo con esto, y un 6.56% de ellos no tienen una opinión al respecto.

Un 37.70% de los estudiantes que participaron de la intervención están totalmente de acuerdo en que las actividades cooperativas han tenido beneficios sobre el aprendizaje del grupo, mientras un 50.38% de ellos sólo están de acuerdo en ello.

En total desacuerdo con esto mismo se encuentra el 1.64% de los estudiantes y un 3.28% están sólo en desacuerdo.

Un 6.56% de los estudiantes está totalmente en desacuerdo de cómo consecuencia de la interacción entre los integrantes de un grupo cooperativo haya favorecido el aprendizaje individual, sin embargo un 36.07% de ellos está totalmente de acuerdo en que si se favoreció el aprendizaje individual. Sólo un 1.64% no tiene opinión a este respecto.

Un 47.54% de los estudiantes de los grupos experimentales están de acuerdo en que sus aportes en el trabajo de grupo contribuyó al aprendizaje de sus compañeros, en tanto, un 36.07% están totalmente de acuerdo con esto. Sólo un 3.28% de ellos están totalmente en desacuerdo en que sus aportes hayan servido para el aprendizaje del resto de sus compañeros de grupo.

Solo un 6.56% de los estudiantes están totalmente en desacuerdo de que el trabajo grupal cooperativo disminuyó sus niveles de ansiedad ante el fracaso académico, por el contrario un 37.70% de ellos están totalmente de acuerdo en que si lo hizo. Un 14.75% de ellos no tiene opinión al respecto.

Un 47.54% de los estudiantes está de acuerdo en que la forma de conducir la asignatura les hizo sentirse más seguros en sus desempeños, a esto se suma un 29.51% de ellos que están totalmente de acuerdo en esto. Sin embargo un 1.64% de los estudiantes están totalmente en desacuerdo con esta opinión.

Un 4.92% de los estudiantes están totalmente en desacuerdo en que la forma de conducir la asignatura les haya contribuido a disminuir sus sensaciones de aislamiento frente al trabajo académico, sin embargo un 52.46% están de acuerdo en que si les ayudó, y un 27.87% de los estudiantes tienen una opinión totalmente de acuerdo respecto de esto.

Un 54.10% de los estudiantes de los grupos experimentales están de acuerdo en que la asignación de roles y tareas al interior de los grupos favorece el compromiso de sus integrantes, y un 26.23% tienen una opinión totalmente en acuerdo con respecto a esto. Sólo un 3.28% están totalmente en desacuerdo con esto.

Un 14.75% de los estudiantes no tienen opinión definida respecto de si el trabajo grupal les fomentó un mayor grado de responsabilidad hacia el logro de los objetivos de aprendizaje en tanto un 24.59% están totalmente en acuerdo de que si lo fomentó y un 50.82% están de acuerdo respecto de esto mismo.

Un 52.46% de los estudiantes están de acuerdo en que el aprendizaje cooperativo les alentó a desarrollar un trabajo más independiente, y el mismo porcentaje de

estudiantes están totalmente de acuerdo con esto. Sólo un 3.28% de ellos están totalmente en desacuerdo con esto.

Un 4.92% de los estudiantes que participaron de las experiencias están totalmente en desacuerdo respecto de que el trabajo cooperativo les favorezca la adquisición de formas de aprendizaje.

Un 49.18% de los estudiantes están totalmente de acuerdo en que el aprendizaje cooperativo les ha servido para desarrollar su capacidad de pensamiento crítico, y un 36.07% de ellos está de acuerdo con esta misma percepción, en tanto un 3.28% tienen una opinión totalmente en desacuerdo respecto de esto mismo y un 9.84% de ellos no tienen una opinión definida respecto de los mismo.

El 3.28% de los estudiantes están totalmente en desacuerdo respecto de que las actividades de trabajo cooperativo les hayan permitido mejorar sus habilidades para elaborar argumentos con claridad, por el contrario un 88.53% de ellos tienen una opinión positiva respecto de esto (52.46% totalmente en acuerdo, y 36.07% en acuerdo. Un 14.75% de ellos no tienen una opinión definida a este respecto.

Un 52.46% está de acuerdo en que las actividades de trabajo cooperativo les permitió mejorar sus capacidades de expresión oral, y un 27.87% de ellos están totalmente de acuerdo con esta opinión. Sin embargo, un 1.64% está en desacuerdo respecto de haber logrado mejorar esta capacidad con estas actividades.

Solo un 1.64% de los estudiantes están totalmente en desacuerdo en que las actividades de trabajo cooperativo permitan crear ambientes de aprendizaje más agradables, por el contrario, un 50.82% de ellos están totalmente de acuerdo en que si lo permiten, y un 37.70 % están de acuerdo en que si lo permiten.

Un 52.46% de los estudiantes están de acuerdo en que las actividades de trabajo cooperativo les ha permitido tener una actitud más positiva hacia el aprendizaje de los contenidos, en cambio un 1.64% de ellos está en desacuerdo con esto, y un 18.03% no tienen una opinión definida respecto de esto mismo.

El 54.10% de los estudiantes están de acuerdo en que los ambientes cooperativos se han acomodado mejor a sus formas de aprendizaje, y sólo un 3.28% están en total desacuerdo con esta opinión. Un 32.79% de ellos están totalmente de acuerdo en que si se ha acomodado mejor a sus formas de aprendizaje y un 8.20% de ellos no tiene opinión al respecto.

Un 4.92% de los estudiantes están totalmente en desacuerdo en que las actividades cooperativas les hayan contribuido a mejorar sus logros en el rendimiento académico, en contraposición con un 27.87% que están totalmente de acuerdo en que si les ha

ayudado a mejorar dichos logros. Además, un 52.46% de ellos están de acuerdo en que ha contribuido a mejorar sus logros.

En relación con la misma línea anterior, un 45.90% están totalmente de acuerdo en que estas actividades han sido gravitantes en su éxito académico, junto a un 37.70% que están también de acuerdo con esta opinión.

Sólo un 3.28% de los estudiantes están en total desacuerdo con que las actividades cooperativas contribuyan de manera importante al desarrollo de sus capacidades de liderazgo, contrariamente a esto, un 39.34% están de acuerdo con que si contribuye y un 40.98% están totalmente de acuerdo con ello. Por su parte, un 11.48% de ellos no manifiestan una opinión definida a este respecto.

Por último, sólo un 1.64% de los estudiantes que participaron de estas experiencias están en total desacuerdo con que las actividades cooperativas les contribuyan de manera importante al desarrollo de las capacidades para el trabajo en equipo, en contraposición con un 62.30% que están de acuerdo en que si les contribuye, y un 26.23% están totalmente de acuerdo con esto. En tanto, un 6.56% de ellos no tienen una opinión definida al respecto

En resumen, porcentajes importante de los estudiantes que participaron de las experiencias de actividades cooperativos opinan que estas han contribuido a crear mejores condiciones para sus aprendizaje en la asignatura, que se han sentido más cómodos con esta forma de trabajo, que les han ayudado a desarrollar formas de trabajo más independientes, y que son de utilidad para desarrollar la capacidad de trabajo en equipo, cuestión que además consideran importante para su desempeño profesional.

CAPÍTULO 7

CONCLUSIONES

7. CONCLUSIONES y PROSPECTIVA

Con este capítulo cerramos lo relacionado con esta investigación, presentando los hallazgos encontrados, los saberes y experiencias vividas en su desarrollo, así como grado en que se cumplen, respecto del planteamiento del problema, los objetivos y las hipótesis, y no menos importante consideramos las experiencias del investigador durante todo el proceso.

Consideramos que las conclusiones refieren un aporte a la educación y las competencias en su proceso de desarrollo en los estudiantes en particular, por más que siempre quedarán interrogantes o pendientes de realizar o a la espera de una respuesta, pues tal y como se ha tratado de dejar constancia, siempre es posible hacer nuevos aportes al conocimiento, más en este ámbito en el que intervienen tantos factores y variables que su acotamiento completo y total es tarea inagotable, frente a las limitaciones del ser humano en capacidad, recursos y tiempo.

Por ello nos resulta estimulante hacer una llamada a otros, interesados en seguir investigando sobre esta materia que, en el fondo, puede ser compartida desde los más diversos ámbitos de la enseñanza y la variedad de las disciplinas, pues se trata ni más ni menos que del aprendizaje de las competencias por parte de los estudiantes y que si en este caso de refieren a la computación se pueden transferir a otras materias u objetos de estudio.

Si el aprendizaje es una tarea esforzada que requiere el concurso de la intencionalidad y compromisos personales, cuyo éxito depende en parte de la gestión cognitivo-emocional de la persona, no es de menor interés estudiar y valorar la

oportunidad de un aprendizaje cooperativo en el que la sintonía cognitivo emocional de varios personas en interacción puede surtir u efecto multiplicador en esas dos dimensiones que tanto tiene que ver con el éxito personal, profesional y social.

La propuesta que hemos realizado, en el contexto de la formación universitaria pretende hallar espacios para el aprendizaje de los estudiantes desde una visión más participativa, en cuya práctica educativa se proyecte al alumnado en el campo de acción compartida, en la que cada uno recibe su aportación con una plusvalía que en otro caso no dispondría de ocasión.

Así pues, la identificación del problema de este estudio se centra en el aprendizaje, las competencias, que adquiere el estudiantado dentro de una estructura de participación en la dinámica de un grupo cooperativo en el que el resultado final, no solo grupal, sobre todo individual tiene impactos sumativos.

En este trabajo no solo ha resultado de interés el hallazgo que nos permite valorar el rendimiento del aprendizaje cooperativo versus el aprendizaje individualizado, sino que es de tanto o más interés la indagación sobre aquellas condiciones que se habrán de dar en las dinámicas internas de tal actividad que atienden a los más diversos aspectos de éstas, caso de lo que hemos denominado interdependencias intragupo, relaciones psicosociales, construcción de significados y calidad de los resultados y su impacto

No menos importante en este proceso tiene que ver con la motivación y satisfacción de las personas en los procesos y tareas que desarrolla, definiendo de un mejor modo la relación cognición-emoción que se traduce en mayores dosis de compromiso, estimulando la creatividad que facilita transitar de lo conocido hacia lo desconocido en la búsqueda de un sistema relacional que facilite el acceso a nuevos conocimientos que atiendan a la interrogación que sugieren los problemas actuales, pero también futuros en una especie de anticipación-previsión sin descartar opciones especulativas que tanto han aportado al acervo cultural y científico de la humanidad.

7.1. Conclusiones

A la hora de elaborar las conclusiones de esta investigación, referenciamos en primer término, los objetivos que se perseguían y las hipótesis de trabajo, ambos elementos enunciados en el capítulo 5. En segundo término, referenciamos los resultados, su análisis y discusión que presentamos en el capítulo 6.

Tomando en cuenta estos elementos, las conclusiones las abordamos como una verificación y contrastación de ellos.

Para hacer esta contrastación, creemos importante destacar los resultados positivos que aportó la intervención, en los ámbitos de logro de competencias, tasas de aprobación de estudiantes, evolución de los indicadores del trabajo cooperativo, y satisfacción de los estudiantes.

El logro de competencias en los tres aspectos que definimos, donde el grupo experimental evidenció niveles bastante superior que el grupo control, no sólo da cuenta de las mejoras en los aprendizajes de los estudiantes del primer grupo, sino que además, da cuenta de alguna manera, de que estos estudiantes, consiguieron un mejor dominio de los conocimientos y las capacidades que se pretenden en este tipo de asignaturas, siendo esto último un resultado relevante, si consideramos que la asignatura de Fundamentos de programación, es la primera que trata con los conocimientos del núcleo de programación, en los planes de formación profesional en CcC, y en nuestro caso, en la Carrera de Ingeniería en Informática.

Por otra parte, esto es también importante, ya que como lo hemos señalado en apartados anteriores, dentro de los curriculums de estas carreras existen varias asignaturas que se relacionan directamente con la resolución de problemas y la programación, además que se reconoce que estas son capacidades que cualquier profesional de este tipo debe poseer.

Las mejoras en las tasas de aprobación de estudiantes, es una aspiración global que se debe perseguir en cualquier intervención de este tipo, ya que esto no sólo mejora los niveles de retención de los estudiantes dentro de la carrera, sino que nos parece que influye positivamente en la percepción y motivación de ellos.

Los resultados referidos a la evolución de los indicadores del trabajo cooperativo, nos parecen relevantes, toda vez que dan cuenta del aspecto central en que se sustentó esta intervención. Consideramos que las mejoras que se fueron dando en la calidad del trabajo cooperativo, no solo son un indicativo de las mejoras en los resultados individuales de los estudiantes del grupo experimental, sino que da cuenta también, de que esta constituye una práctica que es efectiva para mejorar el aprendizaje y el desarrollo de capacidades que son complejas, y que en términos más generales, podría constituirse en una práctica pedagógica, cuyo uso puede ser potencialmente favorable, para el desarrollo de la mayoría de las asignaturas de estos planes de formación profesional.

Por último, los resultados de la encuesta de satisfacción de los estudiantes los consideramos también importantes, ya que dan cuenta de que las actividades cooperativas, consiguieron que los estudiantes del grupo experimental se hayan

sentido más apoyados en sus procesos de aprendizaje, en términos de que estas contribuyeron a que alcanzaran mejores aprendizaje, y mejores condiciones para el desarrollo y el perfeccionamiento de las capacidades que exige la programación de computadores, lo que nos hace suponer también, que ellos se sintieron más motivados para asumir en propiedad las tareas de aprendizaje.

Además de enunciar las conclusiones, exponemos las implicancias que consideramos que se pueden obtener de la intervención, tanto en la referido al uso de metodologías que promuevan aprendizajes de tipo sociocognitivo, cuando se trata con el aprendizaje y el desarrollo de capacidades que son complejas para la mayoría de los estudiantes, y por otra, dando pie para que éstas sean puestas en práctica por otros docentes.

Teniendo en consideración los objetivos que nos habíamos propuesto en esta investigación, y los resultados de la intervención que describimos en el capítulo anterior, estamos en condiciones de afirmar que:

1.- Habiendo realizado la definición del conjunto de competencias que deben alcanzar los estudiantes que cursan la asignatura de Fundamentos de Programación, se verifica que existen diferencias significativas en los niveles de logro de estas competencias entre los grupos experimentales y los grupos control, quedando de manifiesto que el grupo experimental mostró niveles de logro superiores en la evaluación Post-test para todas las competencias evaluadas.

   Los mayores logros para el grupo experimental, también se evidenciaron en las evaluaciones anteriores (primera, segunda y tercera), con excepción de la primera evaluación en lo referido a la evaluación de las competencias de la segunda categoría (2.1, 2.2, 2.3 y 2.4), en las cuales el grupo control mostró niveles de logro levemente superiores que el grupo experimental. Estos resultados son consecuentes con las investigaciones respecto de los logros de las técnicas aprendizaje cooperativo, en el sentido de que al comienzo del uso de estas técnicas, los estudiantes pueden ver disminuidos sus logros en el aprendizaje de los contenidos, debido a que deben aprender a trabajar cooperativamente, requiriendo además de tiempos para adaptarse a su grupo, y conseguir que el trabajo sea efectivo.

2.- Los resultados de las evaluaciones indican también que el grupo experimental muestra resultados que se encuentran más agrupados en torno a las media de su grupo que el grupo control, lo que se manifiesta en los valores de las desviaciones típicas, las que son menores para todas las competencias en este grupo.

3.- Se observa una mejor cohesión y coherencia en el logro de las competencias para el grupo experimental, ya que al determinar los valores de coeficientes de correlación de Pearson para las competencias 2.4 y 3.9 con el resto de las competencias, estos se muestran superiores que para el caso del grupo control, lo que da cuenta de una mejor relación de dependencia entre estas competencias con las demás para el grupo experimental, lo que daría cuenta del logro de un aprendizaje de mejor calidad.

En términos más globales, el grupo experimental alcanzó un nivel de aprobación claramente mayor que el grupo control, que da cuenta de un 75,59% para el primero, contra un 39,84% para el último.

4.- Es posible afirmar que las hipótesis de trabajo han sido verificadas, ya que existen diferencias significativas a favor del grupo experimental respecto del rendimiento del grupo control.

5.- Por otra parte, al valorar positivamente los estudiantes del grupo experimental la intervención, se puede considerar que hay un mayor grado de satisfacción que no sólo les ha favorecido en el proceso sino que también les ha facilitado obtener un mejor resultado de aprendizaje en la signatura.

6.- Además, en el estudio realizado, a partir de la evaluación continua desarrollada se ha podido constatar que esta metodología de trabajo permite a los estudiantes el logro de un mayor grado de consistencia de las competencias al desarrollar de un modo integrado los distintos componentes que las integran, dando así cuenta no solo de un rendimiento superior en los aspectos conceptuales o de comprensión, sino también de habilidades, destrezas, procedimientos y demás estrategias del desempeño.

7.- Por otra parte, desde una perspectiva del docente, se ha posibilitado profundizar en el conocimiento de las dinámicas internas de los grupos en el proceso de aprendizaje, pudiendo determinar algunas de las variables asociadas al éxito, especialmente las asociadas a interdependencia intragrupo como a dimensión psicosocial así como a la construcción de significados.

En definitiva, se constata que el grupo experimental obtiene resultados de logros finales de las competencias que son significativamente superiores a los obtenidos por el grupo control.

Estas diferencias se verifican en aquellas competencias que involucran el manejo de conocimientos declarativos (competencias de la primera categoría), siendo más marcadas en aquellas competencias que involucran el conocimiento procedimental y estratégico (competencias de la segunda y tercera categoría)

Se verifica que la inclusión de actividades de aprendizaje cooperativo como parte del plan de clases de la asignatura, ha contribuido de manera significativa al logro de las competencias en el grupo experimental.

Se demuestra que en la medida en que los estudiantes son implicados de manera más activa en sus procesos de aprendizaje y de construcción de conocimientos, en especial del conocimiento estratégico requerido para la resolución de los problemas de programación, se logran mejores aprendizajes en la asignatura.

Esta forma de conducir el proceso de enseñanza y aprendizaje de la asignatura ha permitido mejorar los niveles de logro de los estudiantes tanto en el aprendizaje de los aspectos metodológicos de la disciplina de la programación, como también en la propia práctica del diseño e implementación de soluciones a los problemas de procesamiento de datos.

Mediante esta forma de intervención se han mejorado de manera significativa los índices de aprobación de la asignatura.

Un porcentaje importante de los estudiantes del grupo experimental se han mostrado más satisfechos y más seguros de sus desempeños en esta asignatura.

Como valor agregado, se ha dado un primer paso hacia el desarrollo de las capacidades para el trabajo en equipo de los estudiantes del grupo experimental, cuestión que es señalada como una capacidad importante que deben poseer los futuros profesionales de CC.

7.2. Implicaciones y prospectiva

Una consecuencia que consideramos relevante de esta investigación, es que sus resultados han mostrado que la utilización de metodologías fundamentadas en los modelos sociocognitivos resultan ser alternativas interesantes de explotar, para intentar mejoras en los procesos de enseñanza y aprendizaje, que por una parte tratan con contenidos relativamente complejos, pero que por sobre todo tratan con el aprendizaje y desarrollo de estrategias cognitivas y de pensamiento dirigidas a la utilización y aplicación de los conocimientos.

Como señalamos en apartados anteriores, si bien la sicología cognitiva ha tenido avances importantes en esta perspectiva, identificando los procesos cognitivos que intervienen en tareas complejas como es la resolución de problemas en general, aún no ha sido posible establecer en términos más prácticos, como estos procesos cognitivos intervienen e interactúan en la mente de los resolutores. Consideramos

que este aspecto es de suma importancia para llegar a establecer la forma en que estos procesos cognitivos pueden ser enseñados y aprendidos con la propia práctica en las tareas de resolución de los problemas.

Aunque esta investigación ha tratado con un área específica de la resolución de problemas, se ha orientado desde la propia práctica en las tareas de resolución de problemas de programación, a crear condiciones para que los estudiantes desarrollen y practiquen el uso de estas estrategias, mediante un proceso que siendo personal, ha estado mediado por los grupos cooperativos y por la socialización de los elementos que intervienen en las tareas intelectuales de resolución de estos problemas.

Otra de las implicaciones importantes que consideramos que tiene esta investigación, es que sus resultados están dando pie para promover la incorporación y el uso de la metodología cooperativa en otras asignaturas del plan de formación profesional, ya que muchas de ellas tratan con el aprendizaje y el desarrollo de las capacidades para resolver problemas, y por ello presentan tasas de reprobación importantes.

En esta perspectiva, los docentes de estas asignaturas podrán encontrar en esta investigación una fuente de experiencia que les permita repensar la dinámica de los procesos de enseñanza y aprendizaje, en especial de aquellas asignaturas en las cuales el aprendizaje y desarrollo de estrategias de pensamiento son determinantes para el logro de sus objetivos.

En consecuencia y como proyección hacia el futuro, con relación a esta metodología de trabajo, consideramos necesario avanzar otras indagaciones, estudios e investigaciones que atiendan fundamentalmente a los aspectos que emergen de nuestro trabajo como referencias de potencial influencia en el éxito de los estudiantes, entendido éste como la construcción de auténticas competencias, y que, por otra parte, ya nos proponen distintas teorías de la psicología del aprendizaje. Nos referimos a:

1. Necesidad de profundizar en la conceptualización del aprendizaje cooperativo, incluyendo las “comunidades de aprendizaje” en sentido amplio y específico como un espacio de un aprendizaje más socializado que permita un mayor grado de desarrollo personal y social

2. Indagar sobre los sistemas relacionales docente-discente desde una perspectiva que se enfoque al desarrollo de la potencialidad de los estudiantes, valorando las estrategias que mayor grado de autonomía y desarrollo promuevan en el estudiante, así resolución de problemas, desarrollo de

proyectos, investigación, etc. pueden ser ejes canalizadores de este tipo de propuestas.

3. Mejorar en el conocimiento de las dinámicas socio-cognitivo-emocionales que se dan en los grupos cooperativos y cuáles de éstas resultan más determinantes en el éxito o fracaso, tanto del propio aprendizaje como de la calidad de éste.

4. Asimismo resulta de interés promover análisis que permitan profundizar en el conocimiento de las competencias, cómo éstas se aprenden, desarrollan y mejoran desde el estudio de las relaciones y sus jerarquías entre los elementos que las definen, valorando la oportunidad de cada uno de éstos en naturaleza, momentos de desarrollo, etc.

En definitiva, consideramos que se abren líneas de investigación que si bien no podrían considerarse nuevas en sentido general, si adquieren una nueva relevancia a partir de nuestro trabajo toda vez que en la modestia del mismo se han evidenciado posibilidades que o bien no estaban suficientemente identificadas o no se habían valorado con el suficiente énfasis para su consideración de forma más generalizada y, sobre todo, como referentes de la práctica docente universitaria,

BIBLIOGRAFÍA

Abelson, H., & Sussman, G. J. (1996). *Structure and interpretation of computer programs* (2da. ed.). Cambridge MA: Editorial MIT Press.

ACM and IEEE Computer Society (2001). *Computing curricula 2001 computer science, December, ACM/IEEE-CS*, recuperado el 10 de Enero del 2008 desde, http://www.computer.org/education/cc2001/final/index.htm.

ACM, IEEE Computer Society, and Association for Information Systems (2005). *Computing curricula 2005: The overview report, September, ACM/IEEE-CS/AIS*, Recuperado el 10 de Enero del 2004 desde, http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf.

ACM/IEEE-CS (1991). *The Joint Curriculum Task Force (1991)*. Computing Curricula 1991. New York: ACM Press.

ACM/IEEE-CS, *The Joint Task Force on Computing Curricula (2001)*. Computing Curricula 2001. Recuperado el 20 de marzo del 2004 desde, http://acm.org/education/curric_vols/cc2001.pdf.

ACM68 (1968). ACM Curriculum Comite on Computer Science Curriculum'68: Recommendations for the undegraduate program in computer science. *Communications of the ACM*, 11(3): 151-197.

ACM (2001). Computing Curricula 2001, Computer Science: Final Report. *Communications of the ACM*.

Adams, D. and Hamm, M. (1996). *Cooperative Learning, Critical Thinking and Collaboration Across The Curriculum*. II Edition, Springfield, Illinois, Charles Thomas Editores

Adams, W. (1980). *The experience of teaching and learning: A phenomenology of education*. Seattle, MA: Psychological Press.

Aho, A., Hopcroft, J., and Ullman, J. (1974). *The design and analysis of computer algorithms*. MA: Addison-Wesley

Ali, S. (2005). Effective Teaching Pedagogies for Undergraduate Computer Science. *Mathematics and Computer Education*, 39(3) 243-270

Alonso, P., García, F., y Mollá R., (2005). *Mejoras en el aprendizaje de la informática en otras escuelas universitarias*. Recuperado el 3 de Enero del 2005 desde www.sig.upv.es/websig/docencia/does/jenui01.pdf

Alvarez, I. y Guasch, T. (2006). Diseño de estrategias interactivas para la construcción de conocimiento profesional en entornos virtuales de enseñanza y aprendizaje. *RED. Revista de Educación a Distancia*, 14. Recuperado el 7 de Agosto del 2011, desde http://www.um.es/ead/red/14/.

Anderson, A. (ed.) (1964). *Minds and machines*. MA:Prentice-Hall

Anderson, J. (1976). *Language, memory and thought*. Hillsdale, NJ: Erlbaum Associates

Anderson, J. (1980). *Cognitive psychology and its implication*. NJ. Freeman

Anderson, J. (1983). *The architecture of cognition*. Cambridge, MA: Hardvard University Press

Anderson, J. (1993). *Rules of the mind*. Hillsdale, NJ: Erlbaum Associates

Anderson, J. (2000). *Cognitive psychology and its implications* (5th ed.). New York: Worth Publishing.

Anderson, L., Krathwohl, D., Airasian, P., Cruikshank, K.., Mayer, R., Pintrich, P. et al. (Eds.). (2001). *A taxonomy for learning, teaching and assessing. A revision of Bloom's taxonomy of educational objectives*. New York: Addison Wesley Longman.

Arvaja, M., Salovaara, H., Häkkinen, P. y Järvelä, S. (2007). Combining individual and group-level perspectives for studying collaborative knowledge construction in context. *Learning and Instruction*, 17, 448-459.

Astrachan, O., Berry, G., Cox, L., & Mitchener, G. (1998). Design patterns: An essential component of CS curricula. *In Proceedings of the 29th SIGCSE Technical Symposium on Computer Science Education* 30(1) 153–160. New York: ACM Press.

Baddeley, A. (1997). *Human memory: theory and practice* (revised ed.). Hove, UK: Psychology Press.

Bailie, F. (1991). Improving the modularization ability of novice programmers. *Papers of the 22nd SIGCSE Technical Symposium on Computer Science Education*, 23, 277–282.

Barberà, E. y Badia, A. (2004). *Educar con aulas virtuales. Orientaciones para la innovación en el proceso de enseñanza y aprendizaje*. Madrid: Antonio Machado Editores.

Barchini, G., Sosa, M., y Herrera, S. (1998). La informática como disciplina científica: Ensayo de mapeo disciplinar. Recuperado el 5 de Abril del 2010, desde Universidad Nacional de Santiago del Estero, http://laboratorios.fi.uba.ar/lie/Revista/Articulos/010102/A1may2004.pdf.

Barkley, E., Cross, K. y Howell, C. (2007). *Técnicas de aprendizaje colaborativo*. Madrid: Morata Editores

Barros, B. y Verdejo, M. F. (1999). *An approach to analyse collaboration when shared structured workspaces are use for carrying out group learning processes*. In S.P. Lajoie and M. Vivet (editores), Artificial Intelligence in Education.

Becker, J., Insley, R., & Endres, M. (1997). Communication skills of technical professionals: A report for schools of business administration. *SIGCPR Computer Personnel*, 18(2), 3–19.

Ben-Ari, M. (1998). Constructivism in computer science education. *Proceedings of the Twenty-ninth SIGCSE Technical Symposium on Computer Science Education* 20(1) 257–261). Atlanta, GA.

Ben-Ari, M. (2004). Situated learning in computer science education. *Computer Science Education*, 14(2), 85–100. Machanick

Berger, P., & Luckmann, T. (1966). *The social construction of reality: A treatise in the sociology of knowledge*. New York: Doubleday.

Berglund, A., Daniels, M. and Pears, A. (2006). Qualitative Research Projects in Computing Education Research, *8 th Australasian Computing Education Conference* (ACE 2006) 52, 13-37.

Biederman, I., & Shiffrar, M. M. (1987). Sexing day-old chicks: A case study and expert systems analysis of a difficult perceptual-learning task. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 13(4), 640–645.

Biggs, J. (1991). Approaches to learning in secondary and tertiary students in Hong Kong: Some comparative studies. *Educational Research Journal*, 6, 27-39

Biggs, J. B., & Collis, K. F. (1982). *Evaluating the quality of learning: The SOLO taxonomy (Structure of the Observed Learning Outcome)*. New York: Academic Press.

Bird, R. (1976). *Programs and Machines*, NewYork: Wiley

Bishop-Clark, C. (1995). Cognitive style, personality, and computer programming. *Computers in Human Behavior*, 11, 241–260.

Bloom, B. S. (Ed.) (1956). *Taxonomy of educational objectives: Book 1 cognitive domain*. London: Longman.

Boada, I., Soler, J., Prados, F., y Poch J., (2006). *Entorno virtual de ayuda a la docencia de un curso de programación básica*. Recuperado el 1 de Febrero del 2006 desde http://acme.udg.es/cidui1.pdf

Bonwell, C., & Eison, J. (1991). Active learning: Creating excitement in the classroom*. ASHE-ERIC Higher Education Report No. 1,* Washington, DC: The George Washington University, School of Education and Human Development.

Bossert, S. (1989). The instructional management role of the principal.*Educational Administration Quarterly*, 18(3), 34-63

Bravo, C., Redondo, M. y Ortega, M., (2004*).* Aprendizaje en grupo de la programación mediante técnicas de colaboración distribuida. Recuperado el 2 de Enero del 2005 desde http://griho.udl.es/i2004/BajarPonencia/46.pdf

Brickner, M.A., Harkins, S. y Ostrom, T. (1986). Personal involvement: Thought provoking implications for social loafing. *Journal of Personality and Social Psichology*, 51, 763-769.

Brna P. y Burton M. (1997). Roles, Goals and Effective Collaboration. *Proceedings of the IV Collaborative Learning Workshop in the 8th World Conference on Artificial Intelligence in Education*. Kobe, Japón.

Brookes, W., & Indulska, J. (1996). Teaching internet literacy to a large and diverse audience. *Proceedings of the Second Australasian Conference on Computer Science Education*. The University of Melbourne.

Brooks, R. E. (1983). Towards a theory of the comprehension of computer programs. *International Journal of Man-Machine Studies*, 18, 543–549.

Brooks, R. E. (1990). Categories of programming knowledge and their application. *International Journal of Man-Machine Studies*, 33, 241–246.

Brown, G. y Atkins, M. (1998). *Effective teaching in higher education*. London: Methue & Ltd.

Brown, G., Bull, J. y Pendlebury, M. (1997). *Assessing student learning in higher education*. Abingdon, Routledge.

Brown, R. (2003). Universities and colleges with laptop and notebook initiatives. Recuperado el 6 de Mayo 2011, desde http://www.acck.edu/~arayb/NoteBookList.html

Brusilovsky, P., Kouchnirenko, A., Miller, P., and Tomek, I., (1994). Teaching programming to novices: a review of approaches and tools. *Educational Multimedia and Hypermedia, Proceeding of ED-MEDIA 94, World Conference on Educational Multimedia and Hypermedia, Vancuver, British Columbia, Canada*, 103-110

Bucci, P., Long, T.J., & Weide, B.W. (2001). Do we really teach abstraction?. *In Proceedings of the 31th SIGCSE Technical Symposium on CS Education,* 26–30. New York: ACM Press.

Buckenmyer, J. (2000, Nov-Dec). Using teams for class activities: Making course/classroom teams work. *Journal of Education for Business, 76* (2), 98-107.

Buffington, J. (1998). Self-directed teams in the introductory information systems course: Lessons learned. *Proceedings to the 13th International Academy for Information Management Annual Conference.* Helsinki, Finland. (ERIC Document Reproduction Service No. ED 431420)

Bunge, M. (2014). *La ciencia, su método y su filosofía*. Ed. Sudamericana, Buenos Aires.

Bunning, C. (2001). *Turning experience into learning*. In L. P. Steffe & J. Gale (Eds.), Action learning at work. Aldershot: Gower.

Burkhardt, J., Détienne, F., & Wiedenbeck, S. (1997). *Mental representations constructed by experts and novices in object-oriented program comprehension*. In S. Howard, J. Hammond, & G. Lindgaard (Eds.), *Human-computer interaction: INTERACT'97,* 339–346. London: Chapman & Hall.

Burrks, A. W., Goldstine, H. H. y von Neumann, J. (1974). Preliminary discussion of the logical design of an electronic computing instrument. En *Jon von Neumann: Collected Works*, 5, Macmillan, N. J., 34-79

Burton, P. and Bruhn, R. (2003). Teaching Programming in the OOP Era, *SIGCSE Bulletin, ACM Publishers*, 35(2), 111-114.

Byrney, P. & Lyons, G. (2001). The effect of student attributes on success in programming, *Proceedings of the 6th annual conference on Innovations and technology in computer science education*, 49-52.

Cabero, J. (2003). *Principios pedagógicos, psicológicos y sociológicos del trabajo colaborativo: su proyección en la telenseñanza*. En F. Martínez (Ed.), Redes de comunicación en la enseñanza. Las nuevas perspectivas del trabajo cooperativo.131-156. Barcelona: Paidós.

Cabero, J. y Llorente, M. (2007). Propuestas de colaboración en educación a distancia y tecnologías para el aprendizaje. EDUTEC, Revista electrónica de tecnología educativa, número 23. Recuperado el 1 de

diciembre de 2011, desde http://edutec.rediris.es/Revelec2/revelec23/jcabero/jcabero.html.

Campbell, D. T. y Stanley, J. C. (1997). *Experimental and Quasi-experimental Designs for Research*. Rand McNally College Publishing Company, EEUU.

Caneo, O. (2008 Octubre). *Aspectos claves del proceso de pensamiento en programación de computadores*. Ponencia presentada ante II Congreso Nacional de enseñanza en Ingeniería, Universidad de Antofagasta, Chile

Caneo, O. (2009 Octubre). *Las complejidades del aprendizaje de la programación de computadores.* Ponencia presentada ante III Congreso Nacional de enseñanza en Ingeniería, Universidad del Bio Bio, Chile

Caneo, O. (2010 Noviembre). *Favoreciendo el aprendizaje de la programación de computadores.* Ponencia presentada ante Primer Encuentro Latinoamericano de Aprendizaje Significativo (I ELAS), Escuela Naval Arturo Prat, Valparaíso, Chile

Cañas, J. J., Bajo, T., & Gonzalvo, P. (1994). Mental models and computer programming. *International Journal of Human-Computer Studies*, 40, 795–811.

Cappel, J. (2001). Entry-level IS job skills: A survey of employers. *Journal of Computer Information Systems*, *42*(2), 76-82.

Casanova, M. (2008). *Aprendizaje cooperativo en un contexto virtual universitario de comunicación asincrónica: un estudio sobre el proceso de interacción entre iguales a través del análisis del discurso*. Tesis Doctoral. Programa de Doctorado Psicología de la comunicación: interacción social y desarrollo humano. Departamento de Psicología Básica, Evolutiva y de la Educación. Universidad Autónoma de Barcelona. Biblioteca de Comunicaciones. Código 1501152964.

Caspersen, M.E., & Bennedsen, J. (2007). Instructional design of a programming course – a learning theoretic approach. *In Proceedings of the 1st International Computing Education Research (ICER) Workshop.*111–122. New York: ACM Press.

CC2001, (C2001), IEEE-CS y ACM Computing Curricula. 2001:Computer Science. Final Report, December 15, 2001. Recuperado el 3 de Marzo del 2007, desde http://www.acm.org/education/curric_vols/cc2001.pdf.

CC2005, (C2005), IEEE-CS y ACM Computing Curricula. 2005: The Overview ReprtComputer Science. Final Report, September 30, 2005. Recuperado el 10 de Abril del 2007, desde http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf

Clanchy, M. T. (1993). *From memory to written record*. (2nd ed.). Oxford: Blackwell.

Clancy, M. J., & Linn, M. C. (1999). Patterns and pedagogy. *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education*, 37–42.

Coll, C. y Colomina, R. (1990). Interacción entre alumnos y aprendizaje escolar. En C. Coll, J. Palacios, y A. Marchesi (Comps.), *Desarrollo psicológico y educación II. Psicología de la Educación*. 435-453. Madrid: Alianza.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing and mathematics. In Laren B. Resnick (Ed*.), Knowing, learning, and instruction,* 452–494. Hillsdale, NJ: Erlbaum.

Collins, W. J. (2005). *Data structures and the Java collections framework* (2nd ed.). Boston, MA: McGraw Hill.

Connolly, C. (2007). *Addressing programming anxiety and underperformance among first year computing students through pedagogical innovation: An in-depth analysis*. Limerick, University of Limerick,.

Cooper, M. (1995). Cooperative learning. An approach for large enrollment courses. *Journal of Chemical Education*, 72(2), 162-164.

Cordes, D., & Parrish, A. (1996, June). Active learning in technical courses. *Proceedings of the 17th Annual National Educational Computing Conference.* Minneapolis, MI. (ERIC Document Reproduction Service No. ED 398884)

Corritore, C. L., & Wiedenbeck, S. (1991). What do novices learn during program comprehension?. *International Journal of Human-Computer Interaction*, 3, 199–222.

CSC (2008). Computer Science Curriculum 2008: An Interim Revision of CS 2001. ACM and IEEE Computer Society. Recuperado el 10 de Enero del 2010 desde http://www.acm.org/education/curricula/ComputerScience2008.pdf

Cruz J. y Carrillo J., *La actividad matemática en el aula*, .J. Giménez, L. Santos, y J. da Ponte, coordinadores, 103-115, Ed. Biblioteca de uno, Barcelona, 2004.

Cuseo, J. (1996). *Cooperative Learning: A Pedagogy for Addressing Contemporary Challanges & Critical Issues in Higher Education*. Marymount College. New Forums Press

Dann, W., Cooper, S., & Pausch, R. (2006). *Learning to program with Alice.* New York. Prentice Hall.

Dansereau, D. (1986). Dyadic cooperative learning and performance strategies. *Annual Meeting of the America Educational Research Association*, San Francisco, CA.

Davies, S. P. (1993). Models and theories of programming strategy. *International Journal of Man-Machine Studies*, 39, 237–267.

De Benito, B., y Pérez, A. (2003). *La evaluación de los aprendizajes en entornos de aprendizaje cooperativo.* 209-226 en F. Martínez (Ed.), Redes de comunicación en la enseñanza, Barcelona, Paidós.

de Raadt, M. (2007). Incorporating strategies explicitly into curricula, Working Paper. Recuperado el 10 de Mayo 2010, desde www.sci.usq.edu.au/research/workingpape s/sc-mc-0705.ps

de Raadt, M., Toleman, M., & Watson, R. (2004). Training strategic problem solvers. *ACM Press*

de Raadt, M., Watson, R., & Toleman, M. (2002). Language trends in introductory programming courses. *Proceedings of the Informing Science and IT Education Conference*, 329–337.

de Raadt, M., Watson, R., & Toleman, M. (2004). Introductory programming: What's happening today and will there be any students to teach tomorrow? *Australian Computer Science Communications*, 26(5), 277–284.

Deavor, J. (1994): Role-paying in the quantitative analysis lab. *Journal of Chemical Education*, 71(11), 980-982.

Decker, R., & Hirshfield, S. (1995). The object concept. Boston, MA: PWS.

Deek, F. P., Kimmel, H., & McHugh, J. A. (1998). Pedagogical changes in the delivery of the first-course in computer science: Problem solving, then programming. *Journal of Engineering Education*, 87, 313-320

Denning, P. J. (1999). Educating a new engineer. *Communications of the ACM* 35(12), 82–97

Détienne, F. (1990). Expert programming knowledge: A schema based approach. In J.M. Hoc, T.R.G. Green, R. SamurÇay, & D.J. Gillmore (Eds.), *Psychology of programming.* 205–222. London: Academic Press.

Deutsch, M. (1949). A theory of cooperation and competition. *Human Relations*, 2, 129-502.

Dijkstra, E. W. (1968). Go To Statement Considered Harmful. *Communications of the ACM*, 11, 147-148.

Dijkstra, E. W. (1972). Notes on structured programming. En *Dahl, Dijkstra y Hoare*, 1-82

Dijkstra, E. W. (1976). *A Discipline of Programming*. Prentice-Hall, Englewood Cliffs, N. J.

Dijkstra, E. W. (1989). On the cruelty of really teaching computer science. *Communications of the ACM*, 32, 1398–1404.

Dillenbourg, P. (1999). *Collaborative learning. Cognitive and computacional approaches*. Amsterdam: Elsevier.

Dillenbourg, P., Baker, M., Blake, A. y O'Malley, C. (1995). The evolution of research on collaborative learning. In Spada, H. and Reimann, P. (editores), *Learning in Humans and Machines*.

Director, S. W., Khosla, P. K., Rohrer, R. A., & Rutenbar, R. A. (1995). Reengineering the curriculum: Design and analysis of a new undergraduate electrical and computer engineering degree at Carnegie Mellon University. *Proceedings of the IEEE*, 83(9), 1246– 1269.

Dreyfus, H., & Dreyfus, S. (1986). *Mind over machine: The power of human intuition and expertise in the era of the computer*. New York: Free Press.

du Boulay, B. (1989). *Some difficulties of learning to program*. In E. Soloway & J.C. Spohrer (Eds.), 283–299. Hillsdale, NJ: Lawrence Erlbaum.

du Boulay, B., O'Shea, T., & Monk, J. (1989). *The black box inside the glass box: presenting computing concepts to novices*. In E. Soloway & J.C. Spohrer (Eds.), Studying the novice programmer, 431–446. Hillsdale, NJ: Lawrence Erlbaum.

Dutt, J. (1994). A cooperative learning approach to teaching an introductory programming course. *Proceedings of the ninth International Academy for Information Management, Las Vegas, NV*, 225-232.

EC77, (1977). Education Comité of the IEEE Computer Society. A curriculum in computer science and engineering. *Publication EHO119-8, Computer Society of the IEEE*, January 1977

Echeita, G. (1995). *El aprendizaje cooperativo. Un análisis psicosocial de sus ventajas respecto a otras estructuras de aprendizaje*. En P. Fernández y A. Melero (Comps). La interacción social en contextos educativos. Madrid: Siglo XXI.

Edwards, S. H. (2004). Using software testing to move students from trial-and-error to reflectionin-action. *Proceedings of the Thirty-fifth SIGCSE Technical Symposium on Computer Science Education*. 26–30. Norfolk, VG.

Ellis, C., Gibbs, S. and Rein, G. (1991). Groupware, Some Issues and Experiences. *Communications of the ACM*, 34(1), 38-58.

El-Sheikh, E. (2006). An Adaptive Learning Environment for Improving Learning Experiences in Introductory Computer Programming Courses. *International Journal of Learning*, 12(9), 83-90

Ender, P., y Lewis, S. (1986). Problem solving strategies and group processes in small groups learning computer programming. *American Educational Research Journal*, 23(2), 243-261.

Evans, M.D. (1996). A new emphasis and pedagogy for CS1 course. *SIGCSE Bulletin*, 28(3), 12–16

Fekete, A., Kay, J., Kingston, J., & Wimalaratne, K. (2000). Supporting reflection in introductory computer science. *Proceedings of the Thirty-first SIGCSE Technical Symposium on Computer Science Education* 144–148). Austin, TX. DOI.

Felder, R., & Brent, R. (1994). Cooperative learning in technical courses: Procedures, pitfalls, and payoffs. *National Science Foundation.* (ERIC Document Reproduction Service Nº. ED 377038)

Felder, R., & Brent, R. (1996). Navigating the bumpy road to student-centered instruction. *College Teaching, 44*, 43-47.

Fernández, A., Trella, M., Galindo, J. y Aranda, M., (2006). Nuevas Tecnologías y Metodologías para la Enseñanza de una Introducción a la Programación de Ordenadores en Ingenierías. Recuperado el 1 de Febrero del 2006 desde www.lcc.uma.es/~afdez/Papers/Papers/chileESC2001.pdf

Fincher, S. (1999). What are we doing when we teach programming? *29th ASEE/IEEE Frontiers in Education Conference*, 12, 4-5.

Fix, V., Wiedenbeck, S., & Scholtz, J. (1993). Mental representations of programs by novices and experts. *Proceedings of the Conference on Human Factors in Computing Systems*, 74–79.

Foyle, H. (1995). *Interactive learning in the higher education classroom: Cooperative, collaborative, and active learningstrategies*. Washington, DC: National Education Association.

Fussell, S., Kraut, R., Lerch, F., Scherlis, W., McNally, M. y Cadiz, J. (1998). Coordination, Overload and Team Performance: Effects of Team Communication Strategies. *Proceedings CSCW'98*, Seattle, Washington, USA.

Garrison, D. y Anderson, T. (2005). *El e-learning en el siglo XXI: investigación y práctica* (trad. de A. Fuentes). Barcelona: Octaedro.

Gehringer, E. F. (2001). Electronic peer review and peer grading in computer-science courses. *Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education*. 139-143.

Gergen, K. J. (1995). Social construction and the educational process. In L. P. Steffe & J. Gale (Eds.), *Constructivism in education*. 17–39. Hillsdale, NJ: Lawrence Erlbaum Associates.

Gibbs, D. C. (2002). An interactive introductory programming environment using a scripting langu*age*. Paper presented at the Ed-Media 2002, Denver, Colorado.

Gibbs, G. (1999). *Using assessment strategically to change the way students learn*. In S. Brown & A. Glasner (Eds.), Assessment matters in higher education. 41–53. Buckingham: Society for Research into Higher Education and Open University Press.

Gilmore, D.J. (1990). Expert programming knowledge: A strategic approach. In J.M. Hoc, T.R.G. Green, R. Samurc¸ay, & D.J. Gillmore (Eds.), *Psychology of programming,* 223–234. London: Academic Press.

Gödel, K. (1934). *On Undecidable Propositions of Formal Mathematical Systems*. Lecture notes taken by Kleene and Rosser at the Institute for Advanced Study, reprinted in Davis, M. (ed.) 1965, New York: Raven.

Goei, S. L., & Pieters, J.M. (1991). Mental modelling as a framework for programming computer numerical controlled machinery. Paper presented at the annual meeting of the American Educational Research Association, Chicago, IL.

Goldberg, A., & Robson, D. (1983). *Smalltalk-80: The language and its implementation*. MA: Addison-Wesley.

Goldfinch, J. (1994). Further developments in peer assessment of group projects. *Assessment & Evaluation in Higher Education*, 19(1), 29-35.

Goldfinch, J. y Raeside R. (1990). Development of a peer assessment technique for obtaining individual marks on a group project. *Assessment & Evaluation in Higher Education*, 15(3), 210-231.

Goldstine, H. H. (1972). *The Computer: from Pascal to von Neumman*. Princeton University Press, Princeton, N. J.

Gomes A. J. & Mendes, A J., (1999). Una animación para el aprendizaje de conceptos básicos de programación. *Revista de enseñanza y Tecnología*, 13, 22-32

Gonzalez, G. (2004). Constructivism in an introduction to programming course. *The Journal of Computing Sciences in Colleges*, 19(4).

Götschi, T., Sanders, I., & Galpin, V. (2003). Mental models of recursion. *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, Reno, NV, 346–350.

Granger, M., & Lippert, S. (1999) Peer learning across the undergraduate information systems curriculum. *The Journal of Computers in Mathematics and Science Teaching, 18* (3), 267-285.

Green, K. (2001, October). The 2001 campus computing survey. *Proceedings of the EDUCAUSE 2001 Annual Conference*. Indianapolis, IN.

Green, T.R.G. and Brna R. (1990). Programming languages as information structures. In J.M. Hoc, T.R.G. Green, R. Samurçay, & D.J. Gillmore (Eds.), *Psychology of programming*. 117–137). London: Academic Press.

Grennon, J. and Brooks, M. (1999). *In Search of Understanding: The Case for Constructivist Classrooms*. Alexandria, VA: ASCD

Guàrdia, L., Sangrà, A. y Maina, M. (2007). *Case-based learning in VTLE. An effective strategy for improving learning design*. En U.Bernath y A. Sangrà (Eds.), Research on competence development in online distance education and e-learning. Oldenburg: BIS-Verlag der Carl von Ossietzky Universität.

Guerrero, L., Alarcón, R., Franco, F., Ibérico, V. y Collazos C. (1999). Una Propuesta para la Evaluación de Procesos de Colaboración en Ambientes de Aprendizaje Colaborativo. *Memorias del Taller Internacional de Software Educativo, TISE'99*. Santiago, Chile.

Guindon, R. (1990). Knowledge exploited by experts during software systems design. *International Journal of Man-Machine Studies*, 33, 182–279.

Gunawardena, C., Lowe, C. y Anderson, T. (1997). Analysis of a global online debate and the development of an interaction analysis model for examining social construction of knowledge in computer conferencing. *Journal of Educational Computing Research*, 17(4), 395-431.

Guralnik, D.B. (1984). *Webster's new world dictionary of the American language* (2nd collage ed.) New York: Simon & Schuster

Hall, K. (1995). Co-assessment: participation of students with staff in the assessment process. A report of work in progress, en *2nd European Electronic Conference on Assessment and Evaluation, 6-10 Febrero, European Association for Research into Learning and Instruction, SIG Assessment & Evaluation.*

Han, S. y Hill, J. (2007). Collaborate to learn, learn to collaborate: examining the roles of context, community and cognition in asynchronous discussion. *Educational Computing Research*, 36(1), 89-123.

Harvey, B., & Wright, M. (1999). *Simply Scheme: Introducing computer science* (2nd ed.). Cambridge, MA: MIT Press.

Hazzan, O. (2006). Abstraction in computer science & software engineering: A pedagogical perspective. System Design Frontier. Recuperado el 22, Enero 2010, desde

http://edu.technion.ac.Faculty/OritH/HomePage/FrontierColumns/OritHazzan_SystemDesigFrontier_Column5.pdf

Heckhausen, H. (1975) *Algunos acercamientos a la interdisciplina: Disciplina e Interdisciplinariedad*, en Apostel et al: Interdisciplinariedad. Problemas de la enseñanza e investigación en las universidades. ANUIES, México

Herken, R. (1988). *The universal Turing machine*. Oxford: Oxford University Press.

Hindle, B. (1993). The Project: putting student-controlled, small-group work and transferrable skills at the core of a Geography course, *Journal of Geography in Higher Education*, 17(1), 11-20.

Hoare, C. A. R. (1972). Proof of correctness of data representations. *Acta Informática* 1, 271-281.

Hoc, J. M. (1989). Do we really have conditional statements in our brains? In E. Soloway & J. C. Spohrer (Eds.), *Studying the novice programmer.* 179–190. Hillsdale, NJ: Lawrence Erlbaum.

Hoc, J. M., & Nguyen-Xuan, A. (1990). Language semantics, mental models and analogy. In J.M. Hoc, T.R.G. Green, R. Samurc¸ay, & D.J. Gillmore (Eds.), *Psychology of programming*, 139–156. London:

Honey, P. & Mumford, A. (1986). *The Manual of Learning Styles. Maidenhead, Berkshire*. P. Honey, Ardingly House. Academic Press.

Holubec, E. y Roy, P. (1984). Circles of learning. cooperation in the classroom. Alexandria. VA. *Association for Supervision and Curriculum Development.*

Hopcroft, J. (1987). Computer Science: The Emergence of a discipline. 1986 Turing Award Lecture. *ACM Communications* 30, 198-202

Horton, I. (2002). *Beginning Java 2*. Birmingham: Wrox.

Hübscher-Younger, T., & Narayanan, N. H. (2003). Constructive and collaborative learning of algorithms. *Proceedings of the SIGCSE Technical Symposium on Computer Science Education*, 6–10.

Hufford, T. (1991). Increasing academic performance in an introductory biology course, *Bioscience*, 41(2), 107-108.

Hundhausen, C. D. (2002). Integrating algorithm visualization technology into an undergraduate algorithms course: Ethnographic studies of a social constructivist approach. *Computers & Education*, 39(3), 237–260.

Husserl, E. (1971). Phenomenology. Edmund Husserl's Article for the Encyclopaedia Brittanica (1927): revised translation by Richard E. Palmer. *Journal of British Society for Phenomenology*, 2, 77–90.

Inaba, A. y Okamoto, T. (1997). The Intelligent Discussion Coordinating System for Effective Collaborative Learning. *Proceedings of the IV Collaborative Learning Workshop in the 8th World Conference on Artificial Intelligence in Education*. Kobe, Japón.

IEEE77 (1977). Education Committee of the IEEE Computer Society. A curriculum in computer science and engineering. *Publication EHO119-8, Computer Society of the IEEE*, January 1977.

Jenkins, T. (2002 September). On the cruelty of really teaching programming. Paper presented at the 2nd LTSN-ICS one day conference on the teaching of programming, University of Wolverhampton, UK.

Jeong, A. (2006). The effects of conversational language on group interaction and group performance in computer-supported collaborative argumentation. *Instructional Science*, 34, 367-397.

Jiménez, G., y Llitjós, A. (2005). Synergeia: Adaptación del BSCW al mundo educativo. Quarkk, Recuperado el 1 de Diciembre de 2011 desde http://www.fq.profes.net/ apieaula2.asp?id_contenido=46049.

Jiménez, G.; Llobera, R., y Llitjós, A. (2006). La atención a la diversidad en las prácticas de laboratorio de química: los niveles de abertura. *Enseñanza de las Ciencias*, 24(1), 59-70.

Johnson, D., Johnson, R., y Smith, K. (1988). Active Learning: Cooperation in the College Classroom, Interaction Book Company. Recuperado el 10 de Enero del 2009 desde http://www.ce.umn.edu/~smith/docs/CL%20College-804.doc

Johnson, D. y Johnson, R. (1975). *Learning Together and Alone. Cooperation, competition and individualization*. Prentice Hall Inc. Englewood Cliffs, New Jersey.

Johnson, D. y Johnson, R. (1979). Conflict in the classroom: controversy and learning. *Review of Educational Research*, 49(1), 51-70.

Johnson, D. y Johnson, R. (1999). Aprender juntos y solos. Aprendizaje cooperativo, competitivo e individualista (trad. de M. Wald). Buenos Aires: Aique. [V.O.: Learning together and alone: cooperative, competitive and individualistic learning. Needham Heights: Allyn & Bacon, 1999].

Johnson, D., Johnson R. y Holubec E. (1999). *El aprendizaje cooperativo en el aula*. Barcelona: Paidós.

Johnson, D., y Johnson, R. (1992). Positive interdependence: key to effective cooperation, en R. Hertz-Lazarowitz y N. Miller (Eds.), Interaction in cooperative groups. *The theoretical anatomy of group learning*, Cambridge, Cambridge University Press, 174-199.

Johnson, D.W. y Johnson, R. (1987). *Cooperation and competition*. Edina, Minnesota: Interaction Book Company.

Johnson, D.W., Maryuama, G., Johnson, R., Nelson, O. y Skon, L. (1981). Effects of cooperative, competitive and individualistic goal structures on achievement. A meta-analysis. *Psychological Bulletin*, 89, 47-62.

Johnson, W. L. (1986). *Intention based diagnosis of novice programming errors*. Los Altos, CA: Morgan Kauffman Publishers.

Johnson, W. L., & Soloway, E. (1984). PROUST: Knowledge-based program understanding. *Proceedings of the 7th International Conference on Software Engineering*, 369–380.

Johnson L., y Miles, L. (2004). Assessing contributions to group assignments. *Assessment & Evaluation in Higher Education*, 29(6), 751-767.

Kagan, S. (1990). The Structural Approach to cooperative learning. *Educational Leadership*. 47(4), 12-15.

Kahney, H. (1989). What do novice programmers know about recursion?. In E. Soloway & J. C. Spohrer (Eds.). *Studying the novice programmer* 209–228. Hillsdale, NJ: Lawrence Erlbaum.

Kamin, S. M., Mickunas, M. D., & Reingold, E. M. (2002). *An introduction to computer science using Java*. Boston, MA: McGraw Hill.

Karau, S.J. y Willian, K.D. (1993). Social loafing. A meta-analytic review and theoretical integration, *Journal of Personality and Social Psychology*, 65, 681-706.

Kaufman, D., Felder, R., y Fuller, H. (2000). Accounting for individual efforts in cooperative learning teams. *Journal of Engineering Education*, 89(2), 133-140.

Kay, J., Barg, M., Fekete, A., Greening, T., Hollands, O., Kingston, J., & Crawford, K. (2000). Problem-based learning for foundation computer science courses. *Computer Science Education*, 10, 109–128.

Kerr, N. (1983). Motivation losses in small groups: A social dilemma analysis. *Journal of Personality and Social Psychology*, 45(4), 819-828.

Kessler, C. M., & Anderson, J. R. (1989). Learning flow of control: Recursive and iterative procedures. In E. Soloway & J. C. Spohrer (Eds.), *Studying the novice programmer.* 229–260. Hillsdale, NJ: Lawrence Erlbaum.

Kölling, M., & Barnes, D. J. (2004). Enhancing apprentice-based learning of Java. *Proceedings of the Thirty-fifth SIGCSE Technical Symposium on Computer Science Education* 286–290. Norfolk, VG.

Kramer, J. (2007). Is abstraction the key to computing?. *Communications of the ACM*, 50(4), 37–42.

Krause, J., and Popovich, N. (1996). A group interaction peer/self assessment process in a pharmacy practice course. *American Journal of Pharmaceutical Education*, 60, 136-145.

Kuhn, T. S. (Ed.) (1996). *The structure of scientific revolutions* (3rd ed.). Chicago, IL: University of Chicago Press.

Kuittinen, M., & Sajaniemi, J. (2003). First results of an experiment on using roles of variables in teaching. Papers from the Joint Conference at Keele University (EASE & PPIG 2003), 347–212 M. de Raadt

Kwan, K., and Leung, R. (1996). Tutor versus peer group assessment of student performance in a simultation training exercise. *Assessment & Evaluation in Higher Education*, 21(3), 205-214.

Lave, J., & Wenger, E. (1991). Legitimate peripheral participation in communities of practice. Situated learning: Legitimate peripheral participation. 89–117. Cambridge: Cambridge University Press.

Lawrence, K. (2001). Some refinements on peer assessment of group projects. *Assesment & Evaluation in Higher Education*, 26(1), 5-18.

Lejk, M., and Wyvill, M. (2001). Peer assessment of contributions to a Group Project: a comparison of holistic and categorybased approaches. *Assessment & Evaluation in Higher Education*, 26(1), 61-72.

León del Barco, B. y otros (2005). *Técnicas de aprendizaje cooperativo en contextos educativos*. Badajoz: Abecedario.

Linn, M.C., & Dalbey, J. (1989). Cognitive consequences of programming instruction. In E. Soloway & J.C. Spohrer (Eds.), *Studying the novice programmer.* 57–81. Hillsdale, NJ: Lawrence Erlbaum.

Lippert, S., & Granger, M. (1997). Peer learning in an introductory programming course. *Proceedings of the 12th International Academy for Information Management Annual Conference*. Atlanta, GA. (ERIC Document Reproduction Service No. 422924)

Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M. et al. (2004). A multinational study of reading and tracing skills in novice programmers. *ACM SIGCSE Bulletin*, 36(4), 119–150.

Lister, R., Simon, B., Thompson, E., Whalley, J. L., & Prasad, C. (2006). Not seeing the forest for the trees: novice programmers and the SOLO taxonomy. *ACM SIGCSE Bulletin*, 38(3), 118–122.

Liu, E.; Lin, S.; Chiu, C., y Yuan, S. (2001). Web-based peer review: the learner as both adapter and reviewer, *IEEE Transactions on Education*, 44(3), 246-251.

Lobato, C. (1998). *El trabajo en grupo: aprendizaje cooperativo en Secundaria.* Bilbao: Universidad del País Vasco (Euskal Herriko Unibertsitatea).

Lui, A. K., Kwan, R., Poon, M., & Cheung, Y. H. (2004). Saving weak programming students: Applying constructivism in a first programming course. *SIGCSE Bulletin*, 36(2), 72-76.

Machanick, P. (1998). The abstraction-first approach to data abstraction and algorithms. *Computers & Education*, 31(2), 135–150.

Machanick, P. (2005). Peer assessment for action learning of data structures and algorithms. *Proceedings of the Seventh Australasian Computer Education Conference (ACE2005), January – February*, Newcastle, Australia, 73–82.

Marcelo, C. y Perera, V. (2007). Comunicación y aprendizaje electrónico: la interacción didáctica en los nuevos espacios virtuales de aprendizaje. *Revista de Educación*, 343, 381-429.

Mayer R. (2003). *Mathematical Cognition*. New York. Ed. Information Age Publishing.

Mayer, R. E. (1987). Cognitive aspects of learning and using a programming language. In J. M. Carrols (Ed.). interfacing throught: *Cognitive aspect of human-computer interaction*. 51-79. Cambridge, MA: MIT Press.

Mayer, R. E. (1989). The psychology of how novices learn computer programming. In E. Soloway & J.C. Spohrer (Eds.), *Studying the novice programmer.* 129–159. Hillsdale, NJ: Lawrence Erlbaum.

McCauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L., and Carol Zander, C., (2008). Debugging: a review of the literature from an educational perspective. *Computer Science Education*. 18(2), 67–92. Routledge.

McConnell, D. (2000). *Implementing Computer Supported Cooperative Learning.* Londres, Kogan Page.

McConnell, J. (1996). Active learning and it use in computer science. *Association for Computing Machinery SIGCSE Bulletin*, 28, 52-54.

McCracken, M., Wilusz, T., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D. et al. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *ACM SIGCSE Bulletin*, 33(4), 125–180.

McDowell, L. (1995). The impact of innovative assessment on student learning. *Innovations and training international*, 32(1), 302-313.

McKenna, A. F., Nocedal, J., Freeman, R., & Carr, S. H. (2005). Introducing a constructivist approach to applying programming skills in engineering analysis. *Paper presented at the Frontiers in Education Conference, Indianapolis, Indiana*.

Mello, J. (1993). Improving individual member accountability in small group work settings. *Journal of Management Education*, 17(2), 253-259.

Mendelsohn, P., Green, T.R.G., & Brna, P. (1990). Programming languages in education: The search for an easy start. In J.M. Hoc, T.R.G. Green, R. Samurçay, & D.J. Gillmore (Eds.), *Psychology of programming* 175–199. London: Academic Press.

Metropolis, N., Howlett, J. y Rota, G. C. (Eds.). (1980). *A History of Computing in the Twentieth Century*. New York, Academia Press

Miliszewska & Tan (2007). Befriending Computer Programming: A Proposed Approach to Teaching Introductory Programming. Issues in Informing *Science and Information Technology*, 4, 278-289.

Miller, J., Groccia, J., & Wilkes, J. (1996). Providing structure: The critical element. In T. Sutherland and C. Bonwell (Eds.), *Using Active Learning in College Classes: A Range of Options for Faculty,* 17-30. San Francisco, CA: Jossey-Bass.

Millis, B., & Cottell, Jr., P. (1998). *Cooperative learning for higher education faculty*. Phoenix, AZ: American Council on Education Oryx Press.

Milne, I. and Rowe, G. (2002). Difficulties in Learning and Teaching Programming: Views of Students and Tutors. *Education and Information Technologies*, 7(1), 55-66

Mitchell, W. (2000). A Paradigm shift to OOP has occurred implementation to follow. *Proceedings of the fourteenth annual consortium on Small Colleges Southeastern conference*, Published in the *Consortium for Computing Sciences in Colleges archive*, 94-105

Monereo C., Castelló, M., Clariana, M., Palma, M., Pérez, M. (1998). *Estrategias de enseñanza y aprendizaje*. Formación del profesorado y aplicación en el aula. Barcelona, Grao.

Morrison, P. y Morrison, E. (Eds.) (1961). *Charles Babbage and his Calculating Engines*. N. J., Dover.

Mukherjee, A. (2000, Spring). Effective user of in-class mini case analysis for discovery learning in an undergraduate MIS course. *Journal of Computer Information Systems, 40* (3), 15-23.

Muller, O. (2007). Pattern-oriented-instruction and its influence on problem decomposition and solution construction. In *Proceedings of the 12th annual SIGCSE Conference on Innovation and Technology* in

Computer Science Education (ITiCSE) 102–06. New York: ACM Press.

Mullera, O. and Haberman, B. (2008). Supporting abstraction processes in problem solving through pattern-oriented instruction. *Computer Science Education*. 18(3), 187–212. Routledge.

Naidoo, R., and Ranjeeth, S., (2007). A Classification of Students' Cognitive Abilities with Errors in Computer Programming. *The international Journal of Learning.* 14(7), 23-38.

Neufeld, D., & Haggerty, N. (2001, Fall). Collaborative team learning in information systems: A pedagogy for developing team skills and high performance. *Journal of Computer Information Systems,* 42(1), 37-43. Northern Michigan University. (n.d.). TLC Initiative. Recuperado el 20 Julio de 2010, desde http://www.nmu.edu/laptops.htm

Oliver, D., Dobele, T., Greber, M., & Roberts, T. (2004). This curse has a Bloom rating of 3.9. *Proceedings of the Sixth Australasian Computing Education Conference (ACE2004)*, 30, 227–231.

Ormerod, T. (1990). Human cognition and programming. In J.M. Hoc, T.R.G. Green, R. Samurc¸ay, & D.J. Gillmore (Eds.), *Psychology of programming*. 63–82. London: Academic Press.

Papert, S. (1991). *Situating constructionism. Constructionism*. Norwood, NJ: Ablex.

Papert, S. (1993). *Mindstorms* (2nd ed.). New York: Basic Books.

Pear, J., & Crone-Todd, D. (2002). A social constructivist approach to computer-mediated instruction. *Computers & Education*, 38(3), 221–231.

Pennington, N., & Grabowski, B. (1990). The tasks of programming. In J.M. Hoc, T.R.G. Green, R. Samurçay, & D.J. Gillmore (Eds.), *Psychology of programming*. 45–62. London: Academic Press.

Perkins, D. N., & Martin, F. (1986). Fragile knowledge and neglected strategies in novice programmers. In E. Soloway & S. Iyengar (Eds.), *Empirical studies of programmers, First Workshop*. 213–229. Norwood, NJ: Ablex.

Perkins, D. N., Hancock, C., Hobbs, R., Martin, F., & Simmons, R. (1989). Conditions of learning in novice programmers. In E. Soloway & J.C. Spohrer (Eds.), *Studying the novice programmer*. 261–279. Hillsdale, NJ: Lawrence Erlbaum.

Perkins, D. N., Schwartz, S., & Simmons, R. (1988). *Instructional strategies for the problems of novice programmers*. In R. E. Mayer (Ed.). Teaching and learning computer programming: Multiple research perspective. 154-178. Hillsdate, NJ: Lawrence Erlbaum Associates.

Petre, M. (1995). Why looking isn't always seeing: Readership skills and graphical programming. *Communications of the ACM*, 38(6), 33–44.

Piaget, J. (1953). *The origin of intelligence in the child*. London: Routledge.

Piaget, J. (1971). *Psychology and epistemology*. New York: Grossman.

Poindexter, S., & Allen, D. (2001). Customizing the classroom learning environment – A phased experiment. *Issues in Information Systems, 2*, 364-370

Porter, R., & Calder, P. (2003). A pattern-based problem-solving process for novice programmers. *Fifth Australasian Computing Education Conference (ACE2003)*, 20, 231–238.

Porter, R., & Calder, P. (2004). Patterns in learning to program: an experiment?. *Proceedings of the Sixth Conference on Australasian Computing Education*, 30, 241–246.

Proulx, V. K. (2000). Programming patterns and design patterns in the introductory computer science course. *Proceedings of the thirty-first SIGCSE technical symposium on computer science education*. 80–84. New York: ACM Press.

Purao, S. (1997, December). Hyper-link teaching to foster active learning. *Proceedings of the 12th International Academy for Information Management Annual Conference*. Atlanta, GA.

Raadt, M., (2007). A Review of Australasian Investigations into Problem Solving and the Novice Programmer, *Computer Science Education*. 17(3), 20–113.

Rada, R., Ramsey, P., & Michailidis, A. (1993). Educational perspectives in collaborative hypermedia. *Proceedings of the 1993 ACM Conference on Computer Science, Indianapolis*, 304–309.

Ramadhan, H. A. (2000). Programming by discovery. *Journal of Computer Assisted Learning*, 16, 83-93.

Ramsden, P. (1988). *Studying learning: Improving teaching*. In P. Ramsden (Ed.), Improving learning: New perspectives. 13–31. London: Kogan Page.

Reber, A. S. (1993). *Implicit learning and tacit knowledge*. New York: Oxford University Press.

Riego, J. (1990). Teaching computer programming. *Journal of Object-Oriented Programming,* 4, 12-32.

Rist, R. S. (1991). Knowledge creation and retrieval in program design: A comparison of novice and intermediate student programmers. *Human–Computer Interaction*, 6, 1–46.

Rist, R. S. (1995). Program estructure and design. *Cognitive Science*, 19, 507-526

Rist, R.S. (1996). Teaching Eiffel as a first language. *Journal of Object-Oriented Programming*, 9, 30–41.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and Teaching Programming: A Review and Discussion. *Computer Science Education*, 13(2).137-172

Rogalsky, J., & Samurçay, R. (1990). Acquisition of programming knowledge and skill. In J. M. Hot, T.R.G. Green, R. Samurçay, & D. J. Gillmore (Eds.), *Psychology of programming*. 157-174. London: Academic Press.

Rountree, N., Rountree, J., & Robins, A. (2002). Identifying the danger zones: Predictors of success and failure in a CS1 course. *Inroads the SIGCSE Bulletin*, 34, 121–124.

Rowlands, S., & Carson, R. (2001). The contradictions in the constructivist discourse. *Philosophy of Mathematics Education Journal*, Recuperado el 14 de Mayo del 2010 desde, http://www.ex.ac.uk/PErnest/pome14/rowlands.pdf.

Sajaniemi, J. (2002). An empirical analysis of roles of variables in novice-level procedural programs. *Proceedings of IEEE 2002 Symposia on Human Centric Computing Languages and Environments (HCC'02)*, 37–39.

Sajaniemi, J., & Kuittinen, M. (2005). An experiment on using roles of variables in teaching introductory programming. *Computer Science Education*, 15(1), 59–82.

Sajaniemi, J., & Kuittinen, M. (2005). An experiment on using roles of variables in teaching introductory programming. *Computer Science Education*, 15(1), 59–82.

Salomon, G., and Perkins, D. (1987). Transfer of cognitive skills from programming: When on how?. *Journal of Educational Computing Research*. 3(2), 149-169

Samurçay, R. (1989). The concept of variable in programming: Its meaning and use in problem solving by novice programmers. In E. Soloway & J., Spohrer (Eds.), *Studying the novice programmer*. 161–178. Hillsdale, NJ: Lawrence Erlbaum.

Sánchez, F. (2006). Objetivos formativos y estrategias docentes para el primer curso de las ingenierías Informáticas. Recuperado el 1 de Febrero del 2006 desde http://research.ac.upc.edu/hpc/Papers/2004/fermin2004aC.pdf.

Santos, L. (1992). Resolución de problemas: el trabajo de Alan Schoenfeld: una propuesta a considerar en el aprendizaje de las matemáticas. *Revista Matemática Educativa*. 4(2), 16-24

Sawyer, S., Eschenfelder, K. R., Diekema, A., & McClure, C. R. (1998). Corporate IT skill needs: A case study of BigCo. *SIGCPR Computer Personnel*, 19(2), 27–41.

Schön, D. (1995). The new scholarship requires a new epistemology. Change, 27(6), 27 – 34.

Schulte, C., Magenheim, J., Niere, J., & Schäfer, W. (2003). Thinking in objects and their collaboration: Introducing object-oriented technology. *Computer Science Education*, 13(4), 269–288.

Slavin, R.E. (1990). *Cooperative learning. Theory, research and practice*. New Jersey: Prentice Hall, Englewood Cliffs.

Sleeman, D., Putman, R. T., Baxtar, J., & Kuspa, L. (1988). An introductory Pascal class: A case study of students´errors. In R. E. Mayer (Ed.). *Teaching and learning computer programming: Multiple research perspective*. 238-257. Hillsdate, NJ: Lawrence Erlbaum Associates

Sloane, K. D., & Linn, M. C. (1988). Instructional conditions in Pascal programming classes". In R. E. Mayer (Ed.). *Teaching and learning computer programming: Multiple research perspective*. 208-235. Hillsdate, NJ: Lawrence Erlbaum Associates

Soloway, E. (1985). The psychology of programming. *Proceedings of the 1985 ACM Annual Conference on The Range of Computing*. 252.

Soloway, E. (1986). Learning to program learning to construct mechanisms and explanations.*Communications of the ACM*, 29(9), 850–858.

Soloway, E., & Spohrer, J .C. (Eds.). (1989). *Studying the novice programmer*. Hillsdale, NJ: Lawrence Erlbaum.

Soloway, E., & Woolf, B. (1980). Problems, plans and programs. *Proceedings of the Eleventh ACM Technical Symposium on Computer Science Education,* 16–24.

Soloway, E., Bonar, J., & Ehrlich, K. (1989). Cognitive strategies and looping constructs. In E. Soloway & J. C. Spohrer (Eds.), *Studying the novice programmer*. 191–207. Hillsdale, NJ: Lawrence Erlbaum.

Soloway, E., Ehrlich, K., & Bonar, J. (1982). Tapping into tacit programming knowledge. *Proceedings of the First Major Conference on Human Factors in Computer Systems*, 52–57.

Spohrer, J. C., & Soloway, E. (1986). Novice mistakes, are the folk wisdoms correct. *Communications of the ACM*, 29(7), 624–632.

Spohrer, J. C., Soloway, E., & Pope, E. (1985). A goal/plan analysis of buggy Pascal programs. *Human–Computer Interaction*, 1, 163–207.

Spohrer, J. C., Soloway, E., & Pope, E. (1989). A goal/plan analysis of buggy Pascal programs. In E. Soloway & J. C. Spohrer (Eds.), *Studying the novice programmer*. 355–399. Hillsdale, NJ: Lawrence Erlbaum.

Spohrer, J.C., & Soloway, E. (1989). Novice mistakes: Are the folk wisdoms correct?. In E. Soloway & J.C. Spohrer (Eds.), *Studying the novice programmer*. 401–416. Hillsdale, NJ: Lawrence Erlbaum.

Sutherland, T., & Bonwell, C. (Eds.). (1996). *Using active learning in college classes: A range of options for faculty*. San Francisco, CA: Jossey-Bass.

Tang, H-L., Lee, S., & Koh, S. (2000-2001, Winter). Educational gaps as perceived by IS educators: A survey of knowledge and skill requirements. *Journal of Computer Information Systems, 41*(2), 76-84.

Thomas, L., Ratcliffe, M., & Thomasson, B. (2004). Scaffolding with object diagrams in first year programming classes: Some unexpected results. *Proceedings of the Thirty-fifth SIGCSE Technical Symposium on Computer Science Education*. 250–254. Norfolk, VG.

Truong, N., Bancroft, P., & Roe, P. (2005). Learning to program through the Web. *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiSCE 2005)*, 9–13.

Truong, N., Roe, P., & Bancroft, P. (2004). Static analysis of students' Java programs. *Proceedings of the Sixth Australian Computing Education Conference (ACE2004)*, 30, 317–325.

Turing, A. (1936). On computable numbers, with an application to the Entscheidungs problem. *Proceedings of the London Mathematical Society*, 2(42), 230-265.

van Gorp, M.J., & Grissom, S. (2001). An empirical evaluation of using constructive classroom activities to teach introductory programming. *Computer Science Education*, 11, 247–260.

van Merrienboer, J. (1990). Instructional strategies for teaching computer programming: interactions with the cognitive style reflection-

impulsivity. *Journal of Research on Computing in Education*, 1990. 23(1), 25-44

Van Slyke, G, Trimmer, K., & Kittner, M. (1997, December). Integrating teamwork into information systems courses. *Proceedings of the 12th International Academy for Information Management Annual Conference*. Atlanta, GA.

Vilalta, R., & Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18, 77–95.

Visser, W. (1990). More or less following a plan during design: Opportunistic deviations in specification. *International Journal of Man-Machine Studies*, 33, 247–278.

Visser, W., & Hoc, J.M. (1990). Expert software design strategies. In J.M. Hoc, T.R.G. Green, R. Samurc¸ay, & D.J. Gillmore (Eds.), *Psychology of programming*. 235–250. London: Academic Press.

von Glasersfeld, E. (1995). Sensory experience, abstraction, and teaching. In L. P. Steffe & J. Gale (Eds.), *Constructivism in education*. 369–383. Hillsdale, NJ: Lawrence Erlbaum Associates.

von Mayrhauser, A., & Vans, A.M. (1994). *Program understanding – A survey* (Tech. Rep. CS- 94-120). Department of Computer Science, Colorado State University.

Vygotsky, L. S. (1978). *Mind in Society*. Cambridge, MA: Harvard University Press

Wegner, P. (1976). Programming languages-the first 25 years. *IEEE Trans. Computers*, 1207-1225.

Wellington, C. A., Briggs, T. H., & Girard, C. D. (2005). Comparison of student experiences with plan-driven and agile methodologies. *Paper presented at the Frontiers in Education Conference, Indianapolis*, Indiana.

Wexelblat, R. L. (Ed.). (1981). *History of Programming Language*. Academic Press. N. J.

Whalley, J. L., Lister, R., Thompson, E., Clear, T., Robins, P., Kumar, P. K. A. et al. (2006). An Australasian study of reading and comprehension skills in novice programmers, using the Bloom and SOLO taxonomies. *Proceedings of the Eighth Australasian Computing Education Conference (ACE2006)*, 52, 243–252.

White, G. and Sivitanides, M. (2002). A theory of the relationships between cognitive requirements of computer programming languages and programmers' cognitive characteristics, *Journal of Information Systems Education*, 13(1), 59-66.

Wiedenbeck, S., & Ramalingam, V. (1999). Novice comprehension of small programs written in the procedural and object-oriented styles. *International Journal of Human-Computer Studies*, 51, 71–87.

Wiedenbeck, S., Ramalingam, V., Sarasamma, S., & Corritore, C.L. (1999). A comparison of the comprehension of object-oriented and procedural programs by novice programmers. *Interacting with Computers*, 11, 255–282.

Williams, K.D., Harkins, S. y Latané, B. (1981). Identificability as a deterrent to social loafing: Two cheering experiments. *Journal o Personality and Social Psichology*, 40, 303-311.

Williams, L., Wiebe, E., Yang, K., Ferzli, M., & Miller, C. (2002). In support of pair programming in the introductory computer science course. *Computer Science Education*, 12, 197–212.

Wills, C. E., Deremer, D., McCauley, R.A., & Null, L. (1999). Studying the use of peer learning in the introductory computer science curriculum. *Computer Science Education*, 9, 71–88.

Winslow, L. E. (1996). Programming pedagogy–A psychological overview. *SIGCSE Bulletin*, 28, 17–22.

Wolf, C. (2003), Weaver: Towards Learning Style-based e-Learning in Computer Science Education. *In 5th Australasian Computing Educ. Conf*., Adelaide, Australia.

Ye, N., & Salvendy, G. (1996). Expert–novice knowledge of computer programming at different levels of abstraction. *Ergonomics*, 39(3), 461–481.

Yu-Fen, S. and Stephen, A., (1993). Mental models and transfer of learning in computer programming. *Research on Computing in Education*. 23(2), 16-45.

Yu-Fen, S., and Stephen, M. (1994). Mental models and transfer of learning in computer programming. *Academic Search Premier*. 26(2), 73-92.

.

ANEXOS

ANEXOS

Referencia de las 11 asignaturas comparadas para la elaboración de la Tabla 94

Nº	Nombre de la primera asignatura del área de conocimientos	Carrera afectada	Universidad	País
1	Programación	Ing. Inf.	Técnica Federico Santa María	Chile
2	Intro. a la Programación	Ing. de Ejec. en Comp. e Inf.	Santiago de Chile	Chile
3	Intro. a la Programación	Ing. en Comp.	P. Univ. Católica de Santiago	Chile
4	Fund. de Programación	Ing. en Inf,(*)	Univ. de Playa Ancha	Chile
5	Fund. de Programación	Ing. en Inf,(**)	Univ. de Valparaíso	Chile
6	Intro. to Programm	Comp. Ing.	Nanyang Technologic Univ.	Singapur
7	Comp. Science 1: Programming	Comp. Science	Univ. of Colorado at Boulder	USA
8	Intro. to Programm	Bachelor of Information Technology	The University Sydney	Australia
9	Intro. a la Programación	Ing. en Inf,	Complutense de Madrid	España
10	Intro. a la Programación I	Ing. en Inf	Univ. de Sevilla	España
11	Intro. a la Programación	Ing. Téc. en Inf. de Gestión	Univ. de Oviedo	España
(*)Carrera de la cual se ha tomado el grupo experimental. (**)Carrera de la cual se ha tomado un grupo control.				

Tabla 93: Referencia de las 11 asignaturas comparadas para la elaboración de la Tabla 94

Localización académica y semestre de impartición de las asignaturas referidas en la tabla 93

Localización académica y semestre de impartición de las asignaturas referidas en la tabla 2											
Asignatura	Instituciones analizadas										
	1	2	3	4	5	6	7	8	9	10	11
Semestre	2	2	1	2	1	1	1	1	1	1	1
Algoritmos y Estructura de Datos II.		X									
Análisis y Diseño de Algoritmos/Algorithms/Programming Languages					X		X	X		X	
Bases de Datos/Bases de Datos y Sistemas de Información/Modelos de Datos/Database Systems/Database Systems 1	X	x	X	X	X	X		X	X	X	X
C Language Proficiency.						X					
Computer Systems.							X				
Sistemas de Bases de Datos/Information Systems/Database Systems 2.					X			X			
Desarrollo de Aplicaciones en Sistemas Distribuidos e Internet.											X
Desarrollo de Aplicaciones Hipermedia.											X
Discrete Structures.							X				
Estructura de Datos/Estructura y Representación de Datos/Estructuras de Datos y Algoritmos/Estructuras de Datos y de la Información/Estructuras y Representación de Datos/Algoritmos y Estructura de Datos I/Principles of Data Structure/Computer Science 2: Data Structures.	X	X	X	X	X		X	X	X	X	X
Ingeniería del Software II/Metodología de Desarrollo de Software.	X			X						X	
Ingeniería del Software III.										X	
Ingeniería del Software/Ingeniería del Software I/Fundamentos de Ingeniería de Software/Software Engineering.	X	X	X	X	X	X			X	X	X
Inteligencia Artificial/Introduction to Artificial Intelligence			X	X			X				
Internet Software Platforms.							X				
Introducción a la Programación II.										X	
Laboratorio de Programación I.									X		
Laboratorio de Programación II.									X		
Laboratorio de Programación III.									X		
Lenguajes de Programación/Principles of Programmin Lenguajes/Programming Languages an Paradigms	X	X					X	X			
Metodología de la Programación a gran Escala											X
Metodología y Tecnología de la Programación.									X		
Métodos Gráficos Computacionales.		X									
Microprocessor Programming.						X					
Programación Declarativa.										X	
Programación I.				X							
Programación Avanzada/Programación II/Programación Orientada a Objetos/Data Structures and Object-Oriented Programming.			X	X		X			X		
Redes de Computadores/ Computer Networks/Data Communications and the Internet	X	X	X	X	X	X	X	X			
Sistemas Operativos/Teoría de Sistemas Operativos/Operating Systems Internals/Operating Systems/Operating Systems and Machine Principles.	X	X	X	X	X	X	X	X	X	X	X
Software Systems and Models.						X					

Taller de Programación.				X							
Taller de Sistemas Distribuidos.		X									
Tecnología de la Programación.											X
Tecnología Multimedia.				X							
Teoría de la Computabilidad.										X	
Teoría de la Programación.											X
(*)Carrera de la cual se ha tomado el grupo experimental. (**)Carrera de la cual se ha tomado el grupo control											

Tabla 94: Localización académica y semestre de impartición de las asignaturas referidas en la tabla 93

Ámbitos del manejo de los conocimientos en PC y los contenidos implicados

Ámbito del manejo de conocimientos	Contenidos implicados
1.1. Conocer la metodología.	La metodología de resolución de problemas de procesamiento de datos.
1.2. Describir los propósitos de cada etapa.	
1.3. Describir los procedimientos para el desarrollo de cada etapa.	
1.4. Diseñar e implementar soluciones a problemas usando la metodología.	
1.5. Utilizar el vocablo técnico de la metodología.	
2.1. Conocer los algoritmos	Concepto de algoritmo. Lenguajes algorítmicos. Formas de representación algorítmica. El lenguaje algorítmico como forma de representación de procesos.
2.2. Describir el propósito de los algoritmos.	
2.3. Conocer los lenguajes algorítmicos.	
2.4. Conocer el significado de la simbología algorítmica.	
2.5. Describir las ventajas y desventajas de cada lenguaje algorítmico.	
2.6. Utilizar el lenguaje y las formas de representación algorítmica para construir algoritmos de solución.	
2.7. Realizar tareas de seguimiento de procesos a partir de un algoritmo.	
2.8. Evaluar la eficacia y eficiencia de un algoritmo.	
2.9. Utilizar el vocablo técnico de los algoritmos.	
3.1. Describir el paradigma de programación estructurada e imperativa.	Programación estructurada e imperativa.
3.2. Reconocer las características de este tipo de paradigma.	
3.3. Describir el concepto de programación descendente.	
3.4. Utilizar el vocablo técnico de la programación estructurada e imperativa.	
4.1. Conocer un lenguaje de programación.	Lenguaje de programación.
4.2. Describir las características generales del lenguaje de programación.	
4.3. Describir las diferentes estructuras de programación secuencial y selectiva.	
4.4. Definir y declarar variables y constantes para almacenar y manipular datos simples.	
4.5. Conocer y utilizar sentencias de entrada y salida de datos.	
4.6. Escribir programas a partir de algoritmos secuenciales	
4.7. Escribir programas a partir de algoritmos selectivos simples y dobles.	
4.8. Escribir programas completos a partir de un algoritmo de solución.	
4.9. Utilizar de manera eficaz y eficiente los recursos del lenguaje de programación para la escritura de programas.	
4.10. Utilizar el vocablo técnico asociado con el uso de los recursos de un lenguaje de programación	
5.1. Describir las características de un lenguaje compilado.	Concepto de compilación.
5.2. Diferenciar entre programa fuente y programa objeto.	
5.3. Definir el concepto de programa ejecutable.	
5.4. Definir el concepto de lenguaje de máquina.	
5.5. Utilizar el vocablo técnico del proceso de compilación.	
6.1. Describir las características del software para la escritura, edición, grabación, recuperación, compilación y ejecución de un programa.	El software de compilación y ejecución.
6.2. Manejar las opciones del software para realizar la escritura y edición del código fuente.	
6.4. Manejar las opciones del software para realizar la grabación y recuperación de archivos de código fuente.	
6.5. Manejar las opciones del software para identificar, interpretar y corregir errores de sintaxis, semántica y declaraciones en un programa fuente.	
6.6. Manejar las opciones del software para ejecutar y realizar seguimiento y evaluación del programa.	
6.7. Utilizar el vocablo técnico asociado con el manejo del software.	
7.1. Describir los propósitos de las pruebas de programas.	Técnicas de prueba.
7.2. Diseñar casos de pruebas.	
7.3. Realizar tareas de prueba y depuración de programas.	
7.4. Utilizar el vocablo técnico asociado con el proceso de prueba de programas.	
8.1. Describir los tipos de estructuras iterativas.	Estructuras iterativas.
8.2. Diferenciar los tipos de estructuras iterativas y sus potenciales aplicaciones.	
8.3. Utilizar las sentencias del lenguaje para implementar estructuras iterativas.	

8.4 Escribir programas que consideren estructuras iterativas.	
8.5. Elegir la estructura iterativa apropiada para una determinada situación de programación.	
8.6. Utilizar el vocablo técnico asociado con las estructuras iterativas.	
9.1. Describir la clase de estructura de datos cadena de caracteres, sus operaciones básicas y sus funciones asociadas.	Estructuras de datos básicas.
9.2. Describir las clases de estructuras de datos tipo arreglos y sus operaciones básicas.	
9.3. Describir la clase de estructura de datos registro y sus operaciones básicas.	
9.4. Definir y declarar variables para almacenar y manipular estructuras de datos básicas.	
9.5. Escribir programas que requieran del uso de estructuras de datos básicas y sus operaciones.	
9.6. Decidir cuál es la estructura de datos más apropiada para un determinado procesamiento.	
9.7. Utilizar el vocablo técnico asociado con las estructuras de datos básicas.	
10.1. Describir el funcionamiento de los algoritmos de ordenamiento.	Algoritmos de ordenamiento.
10.2. Analizar la eficiencia de cada algoritmo de ordenamiento.	
10.3. Decidir cuál es el algoritmo de ordenamiento más adecuado para un determinado procesamiento.	
10.4. Escribir programas que requieran del uso de algoritmos de ordenamiento.	
10.5. Utilizar el vocablo técnico asociado con los algoritmos de ordenamiento.	
11.1. Describir el funcionamiento de los algoritmos de búsqueda.	Algoritmos de búsqueda.
11.2. Analizar la eficiencia de cada algoritmo de búsqueda.	
11.3. Decidir cuál es el algoritmo de búsqueda más adecuado para un determinado procesamiento.	
11.4. Escribir programas que requieran del uso de algoritmos de búsqueda.	
11.5. Utilizar el vocablo técnico asociado con los algoritmos de búsqueda.	
12.1. Describir el concepto de programación modular.	Programación modular.
12.2. Diferenciar entre las distintas formas de pasar parámetros a funciones.	
12.3. Escribir funciones bajo las distintas formas de pasada de parámetros.	
12.4. Organizar programas modularmente.	
12.5. Escribir programas completos que utilicen funciones.	
12.6. Utilizar el vocablo técnico de la programación modular.	

Tabla 95: Ámbitos del manejo de los conocimientos en PC y los contenidos implicados

Asignaturas que requieren conocimientos previos de FP

ASIGNATURA	DESCRIPCIÓN GENERAL	CONOCMIENTOS PREVIOS
Estructura de datos	Asignatura del tercer semestre referida al estudio de estructuras de datos estáticas y dinámicas. En este ámbito se estudian técnicas de organización de datos y la implementación de operaciones sobre estructuras de datos complejas.	Análisis de problemas, Diseño y representación algorítmica de la solución, Transformación del algoritmo en un programa, y Prueba y verificación de programas. Concepto de variable. Tipos de datos. Estructuras de programación. Funciones. Programación modular.
Ingeniería de software	Asignatura del sexto semestre referida al estudio de los aspectos metodológicos y de las técnicas para el desarrollo de software a gran escala, abarcando el proceso y el producto de software que se desea desarrollar.	
Métodos de desarrollo de software	Asignatura del séptimo semestre que es complementaria a la anterior, que se orienta al desarrollo completo de un producto de software.	
Sistemas operativos.	Asignatura de cuarto semestre, referida al estudio de los sistemas operativos como software que controla y administra los recursos de un sistema computacional. Se tratan las características de los distintos sistemas operativos, sus diseños, y los algoritmos que resuelven los distintos problemas de control y gestión de los recursos del sistema computacional.	Transformación del algoritmo en un programa, y Prueba y verificación de programas.
Redes de Computadores	Asignatura de quinto semestre, referida al estudio de los sistemas de comunicación basados en computadores. Se tratan los aspectos teóricos y prácticos de la comunicación, y las formas de comunicación entre computadores, incluyendo los sistemas operativos de red, su diseño, servicios, e implementación	
Inteligencia artificial	Asignatura del séptimo semestre, referida al estudio de los aspectos teóricos de la inteligencia artificial y a la especificación e implementación de pequeñas aplicaciones desarrolladas en un lenguaje de programación inferencial.	Diseño y representación algorítmica de la solución, Transformación del algoritmo en un programa, y Prueba y verificación de programas
Ingeniería del conocimiento	Asignatura del octavo semestre, en la que se estudian los aspectos de análisis y diseño de sistemas que operan sobre bases de conocimiento, y que es continuidad de la anterior.	

Tabla 96: Asignaturas que requieren conocimientos previos de FP

PRE-TEST

PRUEBA DE DIAGNÓSTICO DE CONOCIMIENTOS BÁSICOS PARA ALUMNOS DE LA ASIGNATURA DE FUNDAMENTOS DE PROGRAMACIÓN

En la siguiente prueba de diagnóstico encontrará una serie de preguntas que tienen que ver con los conocimientos básicos que Ud. debería disponer con el objeto de asegurar su éxito en la asignatura.

Como comprobará las preguntas no son difíciles pero requieren de su esfuerzo para comprender exactamente lo que se le está preguntando, esto con la finalidad de que su respuesta sea lo más precisa posible.

Por esta razón es importante que lea atentamente cada pregunta y reflexione sobre la misma antes de construir una respuesta. Verifique siempre que no se ha equivocado en la respuesta que quería construir.

1. Cálculo

1.1 A las 7 A.M. horas, en un lugar determinado, el termómetro marca +5° C y desde las 7 A.M. hasta las 10 A. M. horas, en el mismo lugar, la temperatura sube a razón de 3° C por hora. Con estos datos, determine la temperatura que dicho termómetro debería marcar a las 8 A. M.; a las 9 A.M. y a las 10 A. M.

	Horas	**TEMPERATURA EN ° C**
Respuestas	8 h. A M	
	9 h. A M	
	10 h. A M	

1.2 Disponemos de la cantidad 545 de la que queremos conocer a cuanto equivale el 28% de tal cantidad

	El 28% de 545 es:
Resultado	

1.3 Sabemos que 33 es el resultado de haber aplicado un porcentaje a la cantidad 213. Se requiere determinar qué porcentaje es 33 con relación a 213

	33 es de 213 el %:
Resultado	

1.4 Hemos realizado la compra de un artículo cuyo precio de venta al público es de $1.240. Necesitamos saber cuanto se deberá pagar por el impuesto de IVA, sabiendo que éste es del 18%.

Resultado	El I V A de $ 1.240 es:

2. Expresiones algebraicas:

2.1 A continuación se le presentan varias expresiones algebraicas que es necesario reducir a la forma más sencilla posible. Se le solicita que reduzca las siguientes expresiones:

2.1.1 $2x + 5 - 3xy - 4 + 3x - xy$

Expresión reducida =	

2.1.2 $-3x^2 + 5xy^3 + 8x - x^2 + 6x^3y + 10 + x^3 - 5y^3x$

Expresión reducida =	

2.2 Se tienen varias expresiones que representan una igualdad y en las que existe una incógnita. Le pedimos que determine el valor de la incógnita para que se verifique la igualdad propuesta en cada una de las siguientes expresiones:

2.2.1 $5x = 8x - 15$

Valor de la incógnita es	x =

2.2.2 $9y - 11 = -10 + 12y$

Valor de la incógnita es	y =

2.2.3 $16 + 7t - 5 + t = 11t - 3 - t$

Valor de la incógnita es	t =

2.3 Sabemos que la suma de dos números es 106 y el mayor de ellos excede al menor en 8. Determine el valor de ambos números

Resultados	Nº mayor =	
	Nº menor =	

2.4 Disponemos de tres canastos que contienen en total 575 frutas. El primer canasto tiene 10 frutas más que el segundo y 15 más que el tercero. Encuentre la cantidad de fruta en cada uno de los canastos.

Nº de frutas en cada canasto	1º canasto =	
	2º canasto =	
	3º canasto =	

2.5 Se nos han dado tres sistemas en los que en cada uno de ellos hay dos valores desconocidos, *Se le solicita que determine* el valor de de cada uno de esos valores en cada uno de los sistema para que se cumpla las igualdades en cada caso:

2.5.1 Calcule los valores ***x*** e ***y*** en el siguiente sistema:

$$5x + 6y = 20$$

$$8x - 6y = -46$$

Resultados	**x =**	
	y =	

2.5.2 Determine el valor de m y n que cumplan:

$$\frac{3m}{2} + n = 11$$

$$m + \frac{n}{2} = 7$$

Resultados	**m =**	
	n =	

2.5.3 Calcule el valor de r e s que cumplen con:

$$(r - s) - (6r + 8s) = -(10r + 5s + 3)$$

$$(r + s) - (9s - 11r) = 2s - 2r$$

Resultados	**r =**	
	s =	

2.6. Un sistema de ecuaciones de primer grado como el del ejercicio anterior puede ser resuelto mediante métodos conocidos como *Eliminación por igualación*, *Eliminación por sustitución* o mediante *Reducción*. Tome como referencia uno de estos métodos de solución (a su elección) y describa con el mayor detalle posible la secuencia de pasos y las operaciones necesarias que se deberán ejecutar para resolver un sistema cualquiera de ecuaciones de primer grado.

2.7. A continuación se encontrará con dos funciones para cada una de las cuales se ha asignado un valor a la variable independiente. Calcule en cada caso el valor de la función, según el valor que se indica para la variable en cada caso.

2.7.1. $f(x) = 2x - 6 + 8x^3$ para $x = 2$

Resultado	**f(x) =**	

2.7.2. $r(z) = (z-5)(z^2 + z + 14z) + 2(2z - z^3 + \frac{z^2}{10})$ para $z = 3$

Resultado	**y(z) =**	

2.8. Nos han sido dados los siguientes valores para $a = 2$; $b = \frac{1}{3}$; $x = -2$; $y = -1$; $m = 3$; y $n = \frac{1}{2}$, con lo cual deberá calcular ahora el valor de cada una de las siguientes expresiones:

Resultado	

2.8.1. $\frac{x^4}{8} - \frac{x^2 y}{2} + \frac{3xy^2}{2} - y^2$

Resultado	

2.8.2. $(a-x)^2 + (x-y)^2 + (x^2 - y^2)(m + x - n)$

Resultado	

2.8.3. $x^2(x - y + m) - (x - y)(x^2 + y^2 - n) + (x + y)^2(m^2 - 2n)$

Resultado	

2.9. Más abajo encontrará unas expresiones algebraicas que corresponden a ecuaciones de 2° grado, para las que es necesario determinar los valores de *x*, que cumplen con la igualdad especificada en cada caso:

2.9.1. $3x^2 - 5x + 2 = 0$

Resultado de x_1	
Resultado de x_1	

2.9.2. $3(3x-2)=(x+4)(4-x)$

Resultado de x_1	
Resultado de x_2	

2.9.3. $(x+2)^3-(x-1)^3=x(3x+4)+8$

Resultado de x_1	
Resultado de x_2	

2.9.4. $\frac{5}{x}-\frac{1}{x+2}=1$

Resultado de x_1	
Resultado de x_2	

2.9.5. $\frac{4x^2}{x-1}-\frac{1-3x}{4}=\frac{20x}{3}$

Resultado de x_1	
Resultado de x_2	

2.10. El problema que tiene que resolver ahora consiste en lo siguiente: Considerando como forma general de una ecuación de segundo grado la expresión $Ax^2+Bx+C=0$, describa la secuencia de acciones y operaciones que se deben realizar para encontrar los valores de x que satisfacen la forma general.

3. Álgebra de Boole

3.1. Las tareas que debe resolver ahora tiene que ver con estructuras lógicas de acuerdo con los principios que rigen el álgebra de Boole. Así pues, considerando $p=Verdadero$; $q=Falso$ y $r=Verdadero$ determine el resultado de las siguientes expresiones lógicas. Considere que $\cap$ simboliza el conector lógico y o and, en tanto $\cup$ simboliza el conector lógico o u or

3.1.1. $p \cup q$

Resultado	

3.1.2. $r \cap p$

Resultado	

3.1.3. $(p \cup r) \cap q$

Resultado	

3.1.4. $(q \cup p) \cap (r \cap p)$

Resultado	

3.1.5. $p \cap (p \cup r) \cup (q \cup r)$

Resultado	

4. Resolución de problemas

En este apartado se le van a proponer algunos problemas sobre los que tendrá que analizar y describir los procedimientos lógicos que deben seguirse al objeto de hallar la solución que se le solicita a partir de los datos y elementos que se le proponen. Le invitamos a que lea con atención la formulación de cada problema antes de que trate de ofrecer la solución.

4.1. Describa la secuencia de acciones y operaciones para encontrar el área de un rectángulo sabiendo que ésta es equivalente a la mitad del cuadrado de su diagonal

4.2. Describa la secuencia de acciones y operaciones que deberían realizarse para determinar cuando ganó cada día una persona, sabiendo que en tres días ganó $G, y que cada día ganó la mitad de lo que ganó el día anterior.

4.3. Sabiendo que la edad de A es el triple de la edad de B, y la de B es 5 veces la de C. Además se sabe que B tiene 12 años más que C. Describa la secuencia de acciones y operaciones que debieran efectuarse para determinar la edad de cada uno.

4.4. Suponga que se tiene una ecuación de primer grado de la forma $ax + b = c$ Describa la secuencia de acciones y operaciones que deberían efectuarse para encontrar el valor de la incógnita x.

4.5. Supongamos que vamos a una tienda que se encuentra en liquidación. Supongamos también que los precios marcados en cada artículo, no se encuentran con el descuento aplicado, considerando además que los artículos pueden tener distintos porcentajes de descuentos. Describa la secuencia de acciones y operaciones que se debieran realizar para determinar el valor final a apagar por un artículo cualquiera de la tienda.

4.6. Cuando se emite la factura por la compra de un artículo en ésta se debe indicar el valor del mismo, sin el impuesto IVA, lo que constituye su valor neto. Después deberá expresarse el monto del impuesto IVA del artículo, y finalmente se debe expresar el valor total a pagar por el artículo, una vez incluido el impuesto IVA.
Describa la secuencia de acciones y operaciones que son necesarias ejecutar para determinar el valor neto y el impuesto IVA de un artículo cualquiera a partir de su valor final a pagar.

4.7. Las compañías telefónicas, al igual que otras suministradoras de servicio, suelen establecer valores mínimos por el uso del servicio durante un tiempo establecido, incrementándose después en función del tiempo de uso que sobrepase lo fijado.
Así, una cierta compañía telefónica en su servicio de llamadas locales cobra $85 por los primeros 3 minutos de llamada, incluso aunque estos no se completen. Pero si la llamada excede de este tiempo, se cobran $30 por cada minuto adicional.

Teniendo en cuenta esto describa la secuencia de acciones y operaciones que es necesario realizar para calcular el valor total de una llamada cualquiera, de acuerdo a su duración real.

Poderación de los ítems

1.1 = 3 puntos	1.2 = 3 puntos	1.3 = 3 puntos	1.4 = puntos	2.1.1 = 3 puntos
2.1.2 = 3 puntos	2.2.2 = 3 puntos	2.3.2 = 3 puntos	2.3 = 4 puntos	2.4 = 4 puntos
2.5.1 = 3 puntos	2.5.2 = 3 puntos	2.5.3 = 3 puntos	2.6 = 4 puntos	2.7.1 = 3 puntos
2.7.2 = 3 puntos	2.8.1 = 3 puntos	2.8.2 = 3 puntos	2.8.3 = 3 puntos	2.9.1 = 4 puntos
2.9.2 = 4 puntos	2.9.3 = 4 puntos	2.9.4 = 4 puntos	2.9.5 = 4 puntos	2.10 = 5 puntos
3.1.1 = 3 puntos	3.1.2 = 3 puntos	3.1.3 = 4 puntos	3.1.4 = 4 puntos	3.1.5 = 4 puntos
4.1 = 4 puntos	4.2 = 4 puntos	4.3 = 4 puntos	4.4 = 4 puntos	4.5 = 4 puntos
4.6 = 4 puntos	4.7 = 4 puntos			

POST-TEST

1. En un cierto banco se han establecido las siguientes condiciones para el otorgamiento de créditos a sus clientes:

   - Los créditos de otorgarán a plazos de 1, 2, o 3 años
   - No se otorgarán créditos si el valor de la cuota mensual a pagar por el cliente es mayor que el 25% de los ingresos del cliente.

   Para calcular el monto final a pagar por el cliente por el crédito se usará la formula $Cf=Ci*(1+i)^n$ de donde:

   - *Cf* es el monto final que deberá pagarse por el crédito
   - *Ci* es el monto de crédito solicitado
   - *i* es el interés mensual a aplicar al crédito y
   - *n* es el número de años del crédito.

   Escriba un programa en C, tal que ingresado el monto del crédito que solicita un cierto cliente, además del número de años (1, 2 o 3) en que lo cancelará, y los ingresos que percibe mensualmente el cliente, el programa determine si al cliente se le puede otorgar el crédito, de ser así, se deberá mostrar el valor de cada cuota mensual que deberá pagar el cliente, en caso contrario, se deberá indicar con un mensaje que el crédito no puede ser otorgado.

2. Escriba el prototipo y la definición de una función que reciba un arreglo de enteros de tamaño n y un valor entero y devuelva la cantidad de veces que el valor entero aparece dentro del arreglo

3. En una cierta universidad se ha implementado el siguiente sistema de evaluación para los estudiantes:

   - En cada semestre se aplicarán 3 pruebas, y para el cálculo de la nota semestral se considera que primera vale 25%, la segunda 35%, y tercera 40%.
   - Si la nota semestral es igual o superior a 4,5 el estudiante aprueba el curso, con nota final igual a la nota semestral.
   - Si la nota semestral es menor que 4,5 e igual o superior que 3,0 el estudiante tiene derecho a un examen especial, en cuyo caso, la nota final es la suma del 60% de la nota semestral y el 40% de la nota del examen especial.
   - Si al sumar ambas ponderaciones se obtiene una nota final igual o superior a 4,5 el estudiante es aprobado con nota 4,5, en caso contrario el estudiante es reprobado.
   - Si la nota semestral es menor que 3,0 el estudiante reprueba con nota final igual a la nota semestral.

   Escriba el prototipo y la definición de una función que reciba las cuatro notas de un estudiante cualquiera y calcule y muestre, la nota final del estudiante y el mensaje APROBADO o REPROBADO según corresponda.

4. Escriba el prototipo y la definición de una función que reciba dos números enteros positivos n y m que son distintos de cero y devuelva 1 si n es “contingente” de m. n es contingente de m si la suma de los divisores de n es igual a m. Por ejemplo: 6 es contingente de 12, ya que los divisores de 6 son 1, 2, 3 y 6, que sumados

dan 12. La función debe devolver 0 en caso contrario, y debe considerar que los números pueden ser recibidos en cualquier orden.

5. Escriba el prototipo y la definición de una función que reciba un arreglo de enteros de tamaño n y un valor entero y devuelva la cantidad de veces que el valor entero aparece dentro del arreglo.

6. Suponga que se tiene la definición de una estructura y la declaración de variables que se muestran en el recuadro de la derecha. Suponga también que el arreglo de estructura ha sido llenado con los datos correspondientes. Considerando estos supuestos:

```
typedef struct per
      { char rut[12];
        int edad;
        double peso;};

per pp, p[20];
```

   a. Escriba el prototipo y la definición de una función que reciba el arreglo p y devuelva el promedio de los campos edad de los elementos del arreglo.

   b. Escriba el prototipo y la definición de una función que reciba el arreglo p y una cadena de caracteres de tamaño 12, correspondiente a un rut, y devuelva la posición dentro del arreglo en la cual se encuentra el rut recibido por la función, y en caso de no encontrarlo, la función debe devolver -1. La función debe utilizar el algoritmo de búsqueda binaria, y se supone que el arreglo no está ordenado por el campo de búsqueda.

Competencias evaluadas	Ítems en los que se evalúa
2.4	1, 2, 3, 4, 5, 6ª, y 6b
3.1	1, 2, 3, 4, 5, 6ª, y 6b
3.2	1, 2, 3, 4, 5, 6ª, y 6b
3.3	1, 2, 3, 4, 5, 6ª, y 6b
3.4	2, 5, 6ª, y 6b
3.5	6b
3.6	6b
3.7	1, 2, 3, 4, 5, 6ª, y 6b
3.8	2, 3, 4, 5, 6ª, y 6b
3.9	1, 2, 3, 4, 5, 6ª, y 6b

ENCUESTA DE SATISFACCIÓN DE LOS ESTUDIANTES

Frente a cada ítem de consulta, marque con una X en el casillero que mejor represente su percepción respeto de la forma de conducción de la asignatura	Muy de acuerdo	De acuerdo	No lo sé	En desacuerdo	Totalmente en desacuerdo
1. Siente que las actividades de trabajo cooperativo han contribuido a que Usted se comprometa de manera más activa en su proceso de aprendizaje.					
2. En el ámbito de su grupo de trabajo cooperativo, considera que este tipo de actividades han contribuido al logro de un mayor compromiso colectivo, en el aprendizaje.					
3. Considera Usted que como consecuencia de la interacción entre los integrantes del grupo, se ha favorecido el aprendizaje individual.					
4. Usted cree que con sus aportes en el trabajo de grupo, contribuyó al aprendizaje de sus compañeros de grupo.					
5. Siente que esta forma de trabajo grupal, le ayudó a disminuir los niveles de ansiedad ante las posibilidades de fracaso académico.					
6. Considera que esta forma de conducción de la asignatura, contribuyó a que Usted si sintiera mas seguro en su desempeño.					
7. Cree que esta forma de conducción de la asignatura le ayudó a disminuir su sensación de aislamiento frente al trabajo académico.					
8. Cree que la asignación de roles y tareas al interior del grupo, favorece el compromiso de los integrantes del grupo.					
9. Considera que el trabajo de grupo contribuye a mejorar el grado de responsabilidad de sus integrantes.					
10. Cree que el trabajo grupal fomentó en Usted un mayor grado de responsabilidad hacia el logro de los objetivos de aprendizaje.					
11. Siente que esta forma de aprendizaje le ha alentado a desarrollar un trabajo más independiente.					
12. Cree que las experiencias de trabajo cooperativo favorecen la adquisición de formas de aprendizaje autónomo, preparándole para el aprendizaje durante toda la vida.					
13. Siente que el aprendizaje cooperativo le ha servido para desarrollar su capacidad para razonar de manera crítica.					
14. Siente que las actividades de trabajo cooperativo le han ayudado a mejorar su habilidad para redactar argumentos con claridad.					
15. Considera que las actividades de trabajo cooperativo le han permitido mejorar su capacidad de expresión oral.					
16. Siente que las actividades de trabajo cooperativo, permiten crear ambientes de aprendizaje más agradables.					
17. Siente que las actividades de trabajo cooperativo le han permitido tener una actitud más positiva hacia el aprendizaje de los contenidos.					
18. Siente que los ambientes de trabajo cooperativo se han acomodado mejor a sus formas de aprendizaje.					
19. Cree que las actividades de trabajo cooperativo han contribuido a mejorar sus logros en el rendimiento académico.					

20. Cree que los aprendizajes logrados mediante el desarrollo de las actividades de trabajo cooperativo, han sido gravitantes en su éxito académico.					
21. Considera que las capacidades de liderazgo son importantes para su futuro desempeño profesional.					
22. Cree que las actividades de trabajo cooperativo contribuyen de manera importante al desarrollo de las capacidades de liderazgo.					
23. Considera que las capacidades para el trabajo en equipo son importantes para su futuro desempeño profesional.					
24. Cree que las actividades de trabajo cooperativo contribuyen de manera importante al desarrollo de las capacidades para el trabajo en equipo.					