**UNIVERSIDAD DE OVIEDO**

## Trabajo Fin de Máster

# *CrowDSL*: Plataforma para la gestión de incidencias en el contexto de una Smart City

**Máster en Ingeniería Web**


**Directores:**
Dr. Vicente García Díaz
Dr. Cristian González García

**Autor:**
Darío Rodríguez García


**Julio 2019**

# Resumen

La mejora de la calidad de vida de los seres humanos a lo largo de la historia se ha materializado, generalmente, por medio del avance de la tecnología. En las últimas décadas se ha trabajado para que aquellas personas que tienen acceso a estas tecnologías puedan disponer de más y mejor información acerca de su contexto físico y social. En este punto nacen disciplinas como *Internet of Things*, cuyo objetivo es ampliar la forma tradicional de comunicación entre humanos (*human-to-human*) hacia un nuevo paradigma en el cual diferentes objetos inteligentes construidos de diferentes formas y para propósitos distintos se puedan comunicar a fin de intercambiar información y tomar decisiones.

Dentro de las tecnologías que tienen que ver con *Internet of Things* se agrupan un conjunto de conceptos que han ido tomando una mayor relevancia y aceptación con el paso del tiempo. Términos como *Smart Object*, *Smart Home*, *Smart City* o *Smart Earth* son cada vez más tenidos en cuenta por la sociedad, debido a las ventajas que estos pueden aportar a corto, medio y largo plazo.

El presente proyecto se centra en el concepto de *Smart City*. Una *Smart City* tiene el objetivo de mejorar las infraestructuras de una ciudad a fin de optimizar los diferentes servicios con los que cuentan estas, tales como transporte, energía, acceso a información por parte de los ciudadanos, etc.

Para esta investigación se plantea la creación de una plataforma para la gestión de las incidencias que se generan en el día a día dentro del ámbito de una *Smart City*. La plataforma contempla diversos procesos que tienen por objetivo seguir todo el ciclo de vida de una incidencia desde que esta se origina hasta que se da por resuelta. Para el proceso de captura de los datos por parte de los usuarios se propone el uso de Ingeniería Dirigida por Modelos, por medio de la construcción de un Lenguaje de Dominio Específico (DSL) gráfico que posibilita la estructuración y parametrización de la información con la que se opera. Mediante este DSL el usuario podrá modelar las entidades, sectores y elementos que intervienen en una incidencia concreta. Una vez se completa todo el proceso de reporte de las incidencias estas se harán públicas de modo que sirvan de consulta al resto de los habitantes de una ciudad. La plataforma permite adjuntar elementos multimedia a cada uno de los reportes, de manera que se pueda obtener una representación lo más realmente posible acerca de la gravedad o estado de una incidencia. De igual modo, se dispone de un apartado para que el resto de los usuarios de la plataforma puedan mostrar sus opiniones acerca de las alertas que se publican relativas a su zona. Con esta metodología se pretende abstraer al usuario en la labor de reporte de las incidencias por medio del diseño de un DSL además de lograr que la información con la que se opera se encuentre debidamente etiquetada y estructurada a modo de facilitar su consulta de cara a posteriores propósitos.

# Palabras clave

Internet de las cosas; Ciudad inteligente; Ingeniería Dirigida por Modelos; MDE; Lenguaje de Dominio Específico; DSL; Crowdsourcing;

# Abstract

The improvement of the quality of life of humans over the years has materialized, mainly, by means of technology. In last decades, it has been working to reach that those people who have access to new technologies can get more and better information about their physical and social context. At this point, disciplines such as the *Internet of Things* emerge, The main objective of the *Internet of Things* is to extend the traditional communication mode among humans (*human-to-human*) toward a new paradigm wherein different smart objects built on different ways and for different purposes can communicate among them in order to exchange information and take decisions.

Within the technologies related to the Internet of Things there are a set of terms that have been taking more relevance and approval over the years. Concepts such as *Smart Object*, *Smart Home*, *Smart City* or *Smart Earth* are increasingly being taken into account for society due to the advantages they could provide in a short, medium and long term.

This project focuses on *Smart City* context. A *Smart City* has the objective of improving the infraestructures of a city in order to optimize the different services they have, such as transport, energy, information access on the part of citizens, and so on.

In this work, we propose to build a platform for incidents management in a *Smart City* context. Platform includes multiple processes that have the goal of monitoring the entire life cycle of issues since they are originated and reported, until they are solved. For the data collection process, it will be developed a graphical Domain-Specific Language using Model-Driven Engineering. This approach lets users to structure and parameterize the information they handle. Using this DSL user will be able to model all the entities, sectors and elements that are involved in a certain incident. Upon completion of incidents reporting process, issues are published in the same platform in order to be accessed by other citizens. Platform also supports the attachment of information for each reporting. This way, users can obtain a representation of the incident as real as possible in certain aspects such as the dangerous level or the state of the issues.

# Keywords

Internet of things; Smart City; Model-Driven Engineering; MDE; Domain Specific Language; DSL; Crowdsourcing;

# Contenidos

# Índice de figuras

# Índice de tablas

# 1 Introducción

A lo largo de los últimos años *Internet of Things* (IoT) se ha posicionado como una de las tecnologías que gozan de un mayor impacto dentro del mundo de la informática y las telecomunicaciones. Hasta hace bien poco el uso de Internet se concebía únicamente como una red de comunicación entre humanos, de manera que estos pudiesen comunicarse y compartir información entre ellos mismos. Este paradigma ha ido evolucionando de manera progresiva hacia un nuevo enfoque en el que una "nube" de objetos inteligentes interconectados generan entornos de computación de manera ubicua [1]. Atendiendo a este razonamiento, uno de los objetivos de *IoT* es lograr que, diferentes objetos que se encuentran en el mismo entorno, sean capaces de interactuar y cooperar con sus vecinos con la finalidad de lograr obtener unos objetivos comunes [2]. Dentro de las tecnologías que engloba *IoT* conviven gran cantidad de objetos y dispositivos que ponen de manifiesto la heterogeneidad existente entre los diferentes nodos que se tratan de comunicar. *IoT* puede ser aplicado tanto en objetos inanimados, tales como máquinas, neveras, bricks de leche o yogures, o en objetos animados como personas o animales [3]. Para lograr la comunicación y la monitorización de estas entidades se emplean diversos dispositivos, como sensores, actuadores o tarjetas RFID (Radio Frequency Identification), muchos de los cuales disponen de inteligencia integrada y capacidades de comunicación [4].

Con el paso del tiempo y el avance de las tecnologías relacionadas con *IoT* han ido apareciendo nuevos conceptos que han logrado obtener una fuerte repercusión y aceptación por parte de la sociedad actual en los países desarrollados. Términos como *Smart Object*, *Smart Home*, *Smart City* o *Smart Earth* son tenidos en cuenta cada vez más debido a los beneficios que estos aportan y que podrían llegar aportar a la sociedad del futuro. Todos estos conceptos persiguen un objetivo común, la mejora de la calidad de vida de los usuarios de dichas tecnologías.

Muchas de estas herramientas se enfocan hacia la optimización de los servicios de los que hacen uso las personas, tanto en un ámbito doméstico como a nivel de convivencia en las ciudades. Dos ejemplos claros de esto son las *Smart Homes* y las *Smart Cities*. El término hogar inteligente o *Smart Home* se emplea para designar una vivienda equipada con tecnologías que permiten monitorizar la actividad de sus habitantes de manera que se pueda mejorar su independencia y/o su estado de salud [5]. Gran parte de estos conceptos se alinean con otros de carácter demográfico y sociológico en los que se recoge, por ejemplo, una

tendencia hacia una sociedad cada vez más envejecida, resultando por tanto imprescindible acondicionar los espacios habitables de las ciudades a las circunstancias sociales de cada momento. Bajo esta misma premisa parte de igual modo el concepto de *Smart City*. Son múltiples las definiciones de *Smart City* que se han enunciado desde sus orígenes. Estas ciudades se conciben como un espacio orientado hacia la progresiva mejora de la economía, del gobierno, de la movilidad, etc. [6], que se esfuerza por hacerse "más inteligente", enteniendo esta inteligencia como la eficiencia, sostenibilidad y habilitabilidad de la misma [7] y que comparte la cultura y el conocimiento, alentando a sus ciudadanos a la participación activa [8] en la actividad de la ciudad. Las *Smart Cities* cobran una importancia mayor si cabe debido a que se prevé que las grandes ciudades aumenten en un alto porcentaje su volumen de población a lo largo de las próximas décadas. De acuerdo a las previsiones que se recogen, en el año 2050 el número de personas que viven en las ciudades se podría situar en los 5,2 mil millones, frente a los 2,6 mil millones que se registraron en el año 2010 [9].

Con todos estos datos en la mano parece evidente que uno de los pilares fundamentales bajo los que se deben asentar las ciudades del futuro es la cooperación entre ciudadanos a fin de lograr una mejora en la convivencia y la optimización de los servicios de las zonas urbanas. El presente proyecto de investigación se sitúa justo en este punto, se plantea el diseño y construcción de una plataforma para el reporte y la gestión de las incidencias que se originan en el día a día de una ciudad. Sucesos como inundaciones, cortes de comunicación de diverso tipo o accidentes de tráfico se producen de manera recurrente en los espacios urbanos. Para este proyecto se ha diseño un Lenguaje de Dominio Específico (Domain-Specific Language, DSL) que permite a los ciudadanos de *Smart Cities* abstraer la captura de información y de ese modo reportar todos los sucesos anómalos que se originan a su paso. Mediante este DSL se trata de lograr que los usuarios puedan categorizar las incidencias que se reportan así como plasmar los elementos que intervienen en las mismas (medios de transporte, edificios, seres humanos, etc…). El objetivo final es conseguir que la información sea accesible al mayor número de personas y que se pueda ayudar a estas en la toma de deciciones. Todo el proceso de reporte de incidencias se complementa con la posibilidad de incluir elementos multimedia que permitan obtener un mayor *feedback* de la incidencia, como por ejemplo imágenes, videos, enlaces, datos de mediciones de sensores o comentarios en redes sociales.

# 2 Motivación

*IoT* y *Smart City* representan dos conceptos que en sus orígenes fueron concebidos desde plantamientos y perspectivas muy distintas, sin existir relación alguna entre ellos. Con el paso del tiempo se ha venido experimentando una confluencia entre estos dos conceptos hacia la búsqueda de un objetivo común [10]. Esta confluencia ha posibilitado que se hayan acuñado nuevos términos como el de «*Sensing as a service*» [11]. Este concepto se basa a la idea de aprovechar los componentes presentes en los dispositivos móviles, tales como sensores y otros medidores de diferentes variables para la captura y futuro procesamiento de los datos recabados, de manera que estos se puedan exponer a un conjunto de usuarios que los utilicen a modo de consulta. Basándonos en esta premisa, podríamos enumerar un conjunto de objetivos que se establecen como esenciales para llegar a considerar que una ciudad determinada se enmarca dentro del concepto de *Smart City*. En lo relativo al presente proyecto nos centramos en los puntos que se exponen a continuación.

## 2.1 ACERCAMIENTO DE LOS ÓRGANOS DE GOBIERNO A LOS CIUDADANOS

Un aspecto fundamental en toda sociedad avanzada es el acercamiento de las instituciones públicas a los ciudadanos que estas representan. El hecho de que se establezcan cauces de comunicación entre ciudadanos y órganos de gobierno permite acercar las necesidades de las personas a los representantes que han sido designados para un determinado cargo. En muchos territorios, los gobiernos y organismos públicos han ido adoptando el concepto de "smartness" aplicado a la ciudad con el objetivo de orientar sus nuevas políticas, estrategias y programas, hacia el desarrollo sostenible, el sólido crecimiento económico y la mejora de la calidad de vida de los ciudadanos [12]. Volviendo nuevamente al ámbito de alcance de la presente investigación, se tratará de lograr que las instituciones puedan ser informadas de todos aquellos hechos que resulten reseñables en la actividad cotidiana de la ciudad y que puedan ser reportados por los propios usuarios.

## 2.2 OPTIMIZACIÓN EN EL ACCESO A LA INFORMACIÓN POR PARTE DE LOS CIUDADANOS

Acciones habituales como seleccionar la mejor ruta para llegar a un puesto de trabajo o consultar las vías de una ciudad que se encuentran cortadas al tráfico de personas y/o vehículos representan casuísticas que se producen de manera recurrente en el contexto de una urbe. Una buena forma de optimizar el acceso a este tipo de información pasa por proveer a los ciudadanos de herramientas de consulta que monitoricen el estado de las ciudades en tiempo real. De esta forma, se consiguen prevenir y mitigar diversos acontecimientos tales como atascos, riesgos de accidente o situaciones de emergencia que requieren la intervención de la autoridad competente. Otro de los pilares bajo los que se asienta la investigación en curso es el objetivo de lograr que la información publicada por unos ciudadanos sirva de ayuda a otros a modo de prevención y de soporte a la toma de decisiones.

## 2.3 MEJORA DE LOS SERVICIOS Y LA CALIDAD DE VIDA

Muchos proyectos que tienen por objetivo dotar de inteligencia a diferentes ciudades destinan buena parte de sus fondos a la mejora de las condiciones de habitabilidad de las mismas. Políticas como la reducción de la contaminación y del ruido o la mejora del sistema de tratamiento de residuos se encuentran incluidos en muchos programas políticos, debido a que estos representan asuntos de vital importancia para las personas. Con la evolución de IoT se han ido empleando nuevas tecnologías, tanto hardware como software que posibilitan la captura y procesamiento de datos en tiempo real. En muchas ciudades ya disponen de "nubes" de sensores que permiten obtener de manera actualizada la medición de diferentes magnitudes, facilitando llevar un control detallado de posibles valores anómalos que pongan en riesgo la integridad física de los ciudadanos a corto, medio o largo plazo. En este asunto también juega un papel importante el reporte de incidencias. Mediante el uso de información abierta (*open data*) publicada por organismos públicos y/o por determinados usuarios a título personal se puede complementar la información que se aporta a la hora de presentar el contexto en el que se origina o tiene lugar una determinada incidencia, permitiendo plasmar de manera gráfica el estado de gravedad o peligrosidad de la misma.

# 3 Objetivos

De acuerdo a lo expuesto hasta el momento se ha podido determinar que son varios los factores que deben coexistir para que se esté en condiciones de afirmar que una ciudad se enmarca dentro del concepto de *Smart City*. En lo que atañe a este trabajo, se ha fijado como objetivo principal la validación de la siguiente hipótesis:

> Es posible **abstraer el proceso de captura y reporte de incidencias en el ámbito de una Smart City** por medio del **diseño e implementación de un Lenguaje de Dominio Específico** (DSL) gráfico, aplicando Ingeniería Dirigida por Modelos (MDE), que contemple el conjunto de **entidades, escenarios y sectores** que se interrelacionan en el contexto de una ciudad.

El hecho de que se diseñe y desarrolle un DSL gráfico permite incorporar al proyecto gran parte de las ventajas que aporta el uso de Ingeniería Dirigida por Modelos al proceso de construcción de software. Uno de los principios esenciales bajo los que se asienta la investigación es la idea de lograr un alto nivel de abstracción para los usuarios, una característica inherente a este tipo de lenguajes y que resulta especialmente útil para aquellas personas que no disponen de conocimientos de programación y en los cuales su grado de conocimiento se restringe al dominio del problema para el que se emplea tal lenguaje.

Partiendo del objetivo principal que se ha enunciado, se han fijado un conjunto de subobjetivos que afectan de manera transversal a la temática del proyecto:

## 3.1 DEFINICIÓN DE UN METAMODELO DEL DOMINIO DE CONOCIMIENTO

Con el objetivo de definir las entidades que intervienen en el dominio del problema que se estudia para este proyecto, se planteó el diseño de un metamodelo que permita representar, tanto dichas entidades, como las relaciones existentes entre ellas, detallando de igual modo las restricciones para los diferentes elementos en materia de tipado de datos o de relaciones entre clases. En dicho metamodelo se contemplarán tanto las entidades físicas (personas, animales, ubicaciones, etc.) como sectores de la ciudad, mapas para representar los reportes de incidencias, etc.

## 3.2 DEFINICIÓN DE LA SINTAXIS CONCRETA DEL LENGUAJE DE DOMINIO ESPECÍFICO

Una vez diseñado el metamodelo de la aplicación se continuará por definir la sintaxis concreta de los elementos que intervinen en dicho metamodelo. Para la definición de esta sintaxis concreta se partirá de la sintaxis abstracta que permite definir la forma en que se nombran cada uno de los componentes del DSL. De manera adicional se trabajará en una semántica abstracta que posibilitará efectuar validaciones sobre los modelos a fin de asegurar su validez y consistencia finales.

## 3.3 DESARROLLO DE UN PROTOTIPO

Con el objetivo de validar los resultados de la investigación se desarrollará un prototipo basado en el metamodelo diseñado como punto de partida al comienzo de la investigación. El citado prototipo permitirá operar con cada uno de los componentes que se recogen en el DSL construido a partir del citado metamodelo. Dicho prototipo tendrá como pilar fundamental la construcción de un editor web que permitirá a los usuarios operar con el lenguaje. De manera similar se ofrecerá otro apartado para las alertas que vayan siendo reportadas por los usuarios.

## 3.4 EVALUACIÓN DEL PROTOTIPO

Una vez concluidos todos los subobjetivos marcados hasta este punto se planteará una evaluación que permita estimar el impacto y los resultados de la investigación. Para el desarrollo de esta evaluación se empleará el editor web con el objetivo de que diferentes usuarios operen con el mismo para la definición de un caso práctico específico. De este modo se estudiarán tanto tiempos de trabajo como errores y dificultades encontradas por casa usuario.

# 4 Ámbito de aplicación

El ámbito de aplicación del proyecto que se desarrolla se orienta principalmente a su implantación en diferentes ciudades. Se tiene el objetivo de lograr una mejora en la calidad de vida de los ciudadanos por medio del acceso a fuentes de información. La plataforma tiene por objetivo monitorizar en tiempo real todas las incidencias que se vayan originando en diferentes *Smart Cities*, permitiendo a los ciudadanos tener el control de todo lo que sucede a cada momento. Este proyecto trata de complementar otras políticas y prácticas que ya se se están llevando a cabo en algunas ciudades y que apuestan por lograr gran parte de los objetivos que se fijan para la mejora de las condiciones de vida de los ciudadanos en las ciudades.

A priori no se establece ningún requerimiento en cuanto al número de usuarios que deberían hacer uso de la aplicación ni a las dimensiones territoriales de las ciudades, pero parece evidente que cuanta más información se maneje de manera continua, mayor será el servicio y la ayuda que la aplicación pueda aportar a los ciudadanos que hagan uso de la misma.

Dentro de los eventos que tienen lugar de manera habitual en las ciudades, la aplicación se podría utilizar para informar acerca de cualquoera de los siguientes acontecimietos:

- Accidentes, tanto personales como de tráfico.
- Cortes de comunicación en diversas vías.
- Incendios o inundaciones.
- Daños en las vías públicas.
- Reyertas o peleas entre individuos.
- Etc…

# 5 Estado del arte

La creciente implantación en la sociedad de diferentes tecnologías relacionadas con las Tecnologías de la Información y la Comunicación (TIC), así como una potente innovación en la industria han permitido ir transformando muchas áreas urbanas económica, social y espacialmente [13]. Como se mencionó en la introducción a este proyecto, son numerosos los estudios y estadísticas que sostienen que en las próximas décadas se producirá un éxodo de población hacia las ciudades. Algunos de estos estudios afirman que: en el año 2040 el porcentaje de población que habitará en ciudades se situará en el 65%. Ese mismo año se estima que, al menos 30 ciudades acogerán un volumen de población superior a los 10 millones de habitantes y que el total del conjunto de núcleos urbanos consumirá el 85% de la energía que se produzca a nivel mundial, generando, por contra, el 80% del total de residuos que se originen [14]. Con todos estos datos parece evidente que resulta imprescindible continuar con el trabajo que se está llevando a cabo actualmente para tratar de optimizar el funcionamiento de estas urbes. En este intento por mejorar los servicios y en definitiva la calidad de vida de los ciudadanos tienen cabida muchas tecnologías y disciplinas diversas que podrían contribuir a tal fin.

Para este proyeto se plantea el uso de Ingeniería Dirigida por Modelos, por medio de la construcción de un DSL que facilite la generación y el acceso a la información para todos los individuos de una colectividad. El empleo de esta técnica permite incorporar al proyecto amplios beneficios, como un incremento en la productividad, el acercamiento del dominio del problema a los usuarios y la posibilidad de ofrecerles a estos un mayor nivel de abstracción.

## 5.1 INTERNET OF THINGS

Como ya se ha expuesto en este mismo documento, *the Intenet of Things* es una de las disciplinas que gozan de un mayor impacto dentro del mundo de las tecnologías actuales. Esto viene respaldado por algunos estudios, como el del *United States National Intelligence Council* en el que se considera como una de las 6 tecnologías con mayor impacto en los Estados Unidos hasta 2025 [15], llegando a ser considerada como la segunda revolución de la segunda revolución digital en [16].

El avance de esta tecnologías ha repercutido en que se haya transformado el paradima tradicional de comunicación entre humanos a través de Internet (*human-to-human*) hacia un nuevo enfoque en el que diferentes "objetos inteligentes" se conectan a la red a fin de comunicarse, tomar decisiones e intercambiar información [17], es decir, de máquina a máquina (*machine-to-machine*).

Una de los dificultades principales a las que debe hacer frente IoT es la heterogeneidad existente entre los diferentes componentes que conforman la red, ya que cada uno de ellos disponde de diferentes dimensiones y capacidades de procesamiento [18]. La citada tecnología ha fomentado la aparición de nuevos términos y conceptos que se han ido haciendo cada vez más familiares con el paso de los años. Conceptos como *Smart Object*, *Smart Home*, *Smart City* o *Smart Earth* aparecen de manera recurrente cuando se profundiza en este campo. Este proyecto se centra en el concepto de *Smart City* a fin de poder aportar una solución a uno de los retos fundamentales de estas ciudadades inteligentes, el acceso a la información y mejora de las condiciones de vida de los ciudadanos.

## 5.2 SMART CITY

Al igual que sucede con gran parte de los conceptos que se agrupan dentro de las tecnologías IoT, no resulta sencillo dar una definición concreta y concisa de *Smart City*. Son múltiples los autores que han aportado una acepción para este término. En [19] se define como "El empleo de 'tecnologías de computación inteligentes' para hacer más inteligentes, interconectados y eficientes los componentes y servicios más importantes de la infraestructura de una ciudad (administración, educación, salud, …). Por su parte [8] habla de una ciudad que gestiona las condiciones de todas sus infraestructuras principales, tales como carreteras, puentes o edificios y puede optimizar sus recursos, planificar sus gestiones de mantenimiento y monitorizar aspectos de seguridad mientras se maximizan los servicios para los ciudadanos. Desde el punto de vista del tratamiento de datos, en [20] se define como un espacio donde las TIC permiten afianzar la libertad de expresión y el acceso a la información pública y los servicios. Con todas estas definiciones surge cierta controversia a la hora de clasificar las *Smart Cities* debido a la variedad de criterios que se emplean para tal fin. Como resultado de lo anterior se han publicado diversos rankings que tratan de aportar un método de clasificación para estas ciudades [21]. Uno de los más reconocidos es el expuesto en [22], donde se divide el concepto de ciudad inteligente en 6 sectores, como muestra la Figura 1: Modelo de Smart City propuesto por la Univeridad de Viena: *Smart Economy*, *Smart Mobility*, *Smart Environment*, *Smart People*, *Smart Living* y *Smart Governance*. Partiendo de esta división se hace una subdivisión de estos puntos en la que se evalúa cada una de las ciudades de acuerdo a diferentes criterios, obteniéndose como resultado una puntuación global de manera independiente, de manera que en último término se pueda establecer una comparativa entre las mismas.

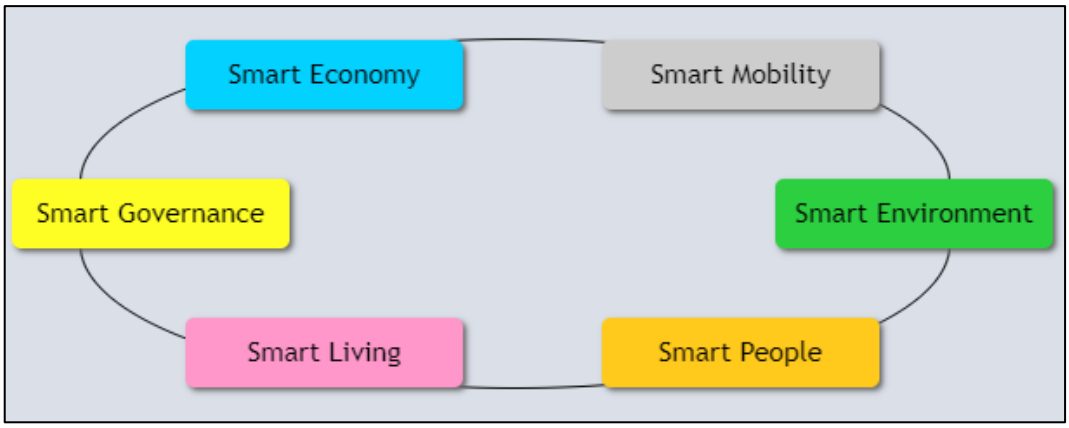


Figura 1: Modelo de Smart City propuesto por la Univeridad de Viena [22]

Este es uno de los enfoques mas aceptados, pero existen más, como el propuesto en [23] donde se estructuran las ciudades entorno a 6 *cores* que se componen de:

1) La sociedad conformada por los ciudadanos, compuesta por la seguridad pública (tratamiento de incendios, inteligencia policial, …), la salud, la educación y la calidad de vida.
2) La regulación fiscal, constituida por la gestión administrativa, el entorno político y la planificación económica.
3) El transporte: Tratamiento de las vías y el tranporte público urbano además del aéreo y marítimo.
4) Las comunicaciones: Redes de comunicación de telefonía, banda ancha, etc, …
5) Agua: Gestión del ciclo del agua, del suministro y de la depuración de la misma.
6) Energía: Producción, gestión del transporte y de las posibles pérdidas generadas.

## 5.3 INGENIERÍA DIRIGIDA POR MODELOS

La Ingeniería Dirigida por Modelos (Kent 2002), en adelante MDE (Model-Driven Engineering), surge con el objetivo de solucionar problemas inherentes al desarrollo de software, tales como el desvío en los costes y la planificación o la baja calidad del software que se obtiene de un desarrollo. MDE se fundamenta en idea de automatizar o semiautomatizar procesos de manera que se logren obtener amplias ventajas, como la mejora del nivel de abstracción experimentado por el usuario. Esto último se logra por medio de la construcción de los modelos. De manera general el concepto de modelo se emplea para designar la forma en que se va a construir algo. En la construcción de software los modelos tratan de extraer las cualidades más esenciales de los objetos por medio de un proceso de abstracción, de manera que se representen los aspectos más importantes del sistema con el que se opera [24].

La aplicación de MDE comienza con la selección del dominio de un problema que se trata de resolver. Si este dominio resulta excesivamente complejo se podría llegar a subdividir en otros dominios más pequeños [21]. Para este

trabajo, el dominio del problema se centra en una ciudad inteligente y se trata de modelar la captura y reporte de incidencias por parte de los ciudadanos de la misma.

El proceso de modelado continuaría con el diseño de un metamodelo, basándose en su meta-metamodelo. Un meta-metamodelo describe al metamodelo, es decir, representa un lenguaje que permite definir dicho metamodelo. Este mismo patrón se sigue con los pares de elementos metamodelo-modelo y modelo-elementos del mundo real (el dominio del problema). Estas relaciones se representan en la Figura 2: Relación entre elementos de MDE expuesta en [25].

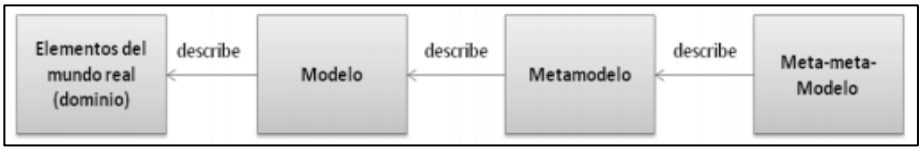


Figura 2: Relación entre elementos de MDE expuesta en [25]

Dos de los meta-metamodelos más extendidos y utilizados son Ecore [26] y MetaObject Facility Specification (MOF) del Object Management Group (OMG) [27]. Los metamodelos se componen de dos tipos de sintaxis: la abstracta y la concreta. Mediante la sintaxis abstracta especificamos la estructura del mismo, es decir, las construcciones, sus propiedades y los conectores que puede tener dicho lenguaje [28]. Por contra, la sintaxis concreta se emplea para definir la notación o representación de los elementos del metamodelo que emplearán los usuarios a la hora de construir los modelos. De igual modo existe el concepto de semántica estática, relacionada con la sintaxis abstracta, que permite efectuar chequeos semánticos en los modelos a fin de asegurar la consistencia de los mismos.

## 5.4 LENGUAJES DE DOMINIO ESPECÍFICO

Los Lenguajes de Dominio Específico (DSLs) [29], son lenguajes que se destinan a la representación de los conceptos propios de un dominio.

Ofrecen notaciones y construcciones orientadas a un dominio particular, presentando ganancias sustanciales en términos de expresividad y facilidad de uso en comparación con los lenguajes de propósito general (GPL) para el dominio en cuestión. Estas ganancias repercuten en mejoras de la productividad y reducción de los costes de mantenimiento [30].

## 5.5 XML

eXtensible Markup Language (XML) es un lenguaje de marcado que se encuentra estandarizado por el W3C [31]. Se emplea para el almacenamiento y transporte de datos. La versatilidad del lenguaje redica en que no dispone de etiquetas predefinidas, por lo que puede ser extendido en función de las necesidades de uso.

## 5.6 XML SCHEMA

Un XML Schema permite especificar la estructura que deberá seguir un documento XML. De manera más precisa, un XML Schema tiene por objetivo definir que los bloques que conforman un archivo XML sean válidos de acuerdo a diferentes criterios [32], como los elementos y atributos que pueden aparecer en el documento, el número y el orden de los elementos "hijo", los tipos de datos para elementos y atributos o los valores fijos y por defecto que pueden tomar estos.

XML Schema presenta una serie de ventajas frente a alternativas como el uso de DTDs. Los esquemas soportan un sistema de tipado de datos, utilizan sintaxis XML, permiten una comunicación fiable entre un emisor y un receptor y posibilitan la detección de errores más allá de el correcto formateo del documento.

## 5.7 XMI

XML Metadata Interchange (XMI) es un estándar que se emplea para representar información orientada a objetos utilizando XML. La primera versión (XMI 1.0) fue publicada por el Object Management Group [27] en febrero de 1999. La última versión estable fue publicada en junio de 2015 (versión 2.5.1 [33]).

Como resultado de la capacidad de XMI para representar diferentes variantes de información orientada a objetos, el software que soporta esta tecnología puede ser empleado para proporcionar métodos de integración entre diferentes aplicativos. La principal ventaja que representa XMI frente XML es que XML no es orientado a objetos, de este modo XMI emplea modelos a fin de asegurar que los objetos se comparten de manera consistente.

Para el proyecto CrowDSL emplearemos esta tecnología con el fin de validar que los modelos que generan los usuarios de la aplicación se ajustan al metamodelo que se presentará en posteriores apartados del presente documento. De igual modo XMI favorece también la portabilidad y la reusabilidad de los modelos e instancias con los que se opera permitiendo además establecer validaciones sobre el tipo de los campos que permiten representar las diferentes entidades que intervienen en el dominio del problema a resolver.

## 5.8 PLATAFORMAS DE GESTIÓN DE INCIDENCIAS EN SMART CITIES

Previamente al inicio de esta investigación se ha llevado a cabo un análisis de diferentes proyectos y plataformas que ya han sido implantados en diferentes ciudades a fin de conocer en qué líneas de investigación se está trabajando en esta materia y el impacto o aceptación de esto proyectos en la sociedad. Se han seleccionado un conjunto de cuatro proyectos cuyo propósito y funcionamiento se expone a continuación.

### 5.8.1 mPass y WhenMyBus [34]

Este proyecto tiene como objetivo trazar rutas personalizadas para usuarios con necesidades especiales [34]. El sistema se compone de dos subsistemas que operan de manera conjunta, *mPass* y *WhenMyBus*. *mPass* tiene la función de recopilar datos acerca de accesibilidad urbana, tanto por medio del empleo de sensores como por datos proporcionados por los propios usuarios mediante el uso de una app. Como resultado se obtienen datos georeferenciados, permitiendo detectar barreras arquitectónicas y proveyendo de rutas peatonales para los ciudadanos. Ambos módulos comparten un submódulo independiente que gestiona el perfil de los usuarios, en el que se puede definir información personal, del dispositivo que se utiliza, velociades medias, hábitos de tránsito, etc.

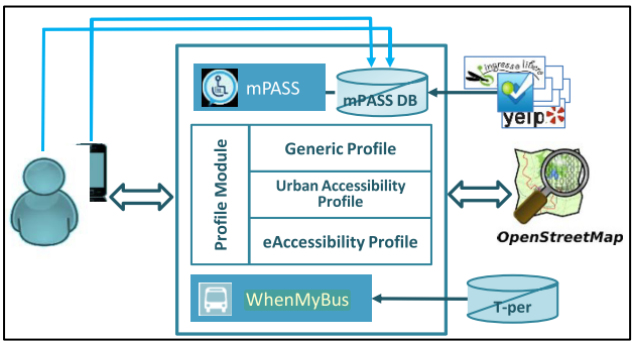


Figura 3: Arquitectura mPass y WhenMyBus

*WhenMyBus* ha sido desarrollado para proporcionar información útil a los ciudadanos que viejan en autobús por la ciudad. Permite obtener datos en tiempo real acerca de la disponibilidad y equipamiento de estos medios de transporte, así como indicar los vehículos que disponen de alguna barrera que impide su uso a personas con discapacidades.

### 5.8.2 CrowdOut [35]

Uno de los retos principales de una Smart City es asegurar la seguridad de los ciudadanos. CrowdOut permite a los usuarios reportar incidencias de tráfico que ellos presencian en tiempo real [35], de modo que estas puedan ser monitorizadas de igual modo en tiempo real.

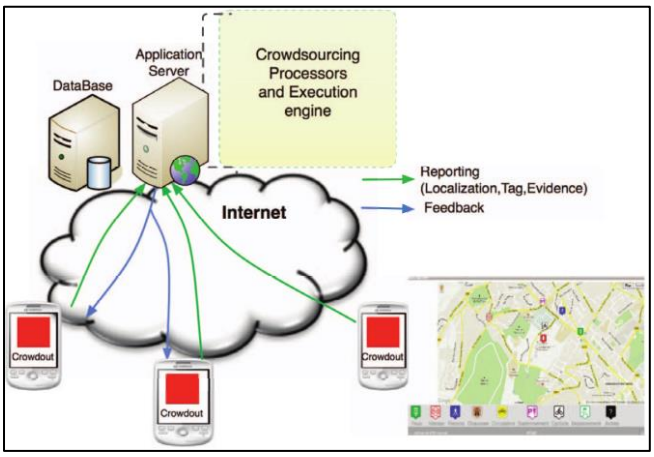


Figura 4: Diagrama CrowdOut

Hoy en día diversos dispositivos, como por ejemplo los smartphones, proporcionan una amplia variedad de sensores y componentes tales como GPS, giroscopio, acelerómetro o micrófono, que permiten que en muchos casos se pueda tanto enviar como recibir datos, habiendo cambiando el rol del usuario de consumidor a productor de información.

El principal objetivo de CrowdOut es el reporte de incidencias de seguridad relacionadas con las vías públicas (excesos de velocidad, coches incorrectamente estacionados, incluplimiento de la señalización de tráfico, etc…), agrupando estas incidencias en diferentes categorías (iluminación de los vehículos, peatonales,

parkings, carretera, velocidad, tráfico, ciclistas, adelantamientos, etc…). De esta forma todos estos acontecimientos se representan sobre un mapa y se posibilita hacer una distinción entre zonas seguras e inseguras. Además, a cada momento, los usuarios pueden seleccionar las incidencias que están visualizando. La aplicación recoge un conjunto de atributos que permiten conocer la ubicación desde dónde la incidencia ha sido reportada.

### 5.8.3 FixMyStreet [36]

*FixMyStreet* fue uno de los primeros proyectos basados en *crowdsourcing* del Reino Unido. En enero de 2007, mySociety [37], una fundación que trabaja para lograr la participación pública de los ciudadanos a través de múltiples iniciativas, puso en funcionamiento un proyecto que permitía a los ciudadanos reportar, visualizar y discutir incidencias detectadas de manera local, tales como, grafities, mal estado de las vías o en el alumbrado público. Estas incidencias son reportadas a la autoridad competente para su tramitación [36].

Figura 5: FixMyStreet

Las incidencias se clasifican en diversas categorías (estacionamiento de vehículos, griffities, baches, vehículos abandonados, etc…). En el caso de que un ciudadano necesite reportar una incidencia que no se pueda clasificar en alguna de las categorías anteriores tiene potestad para contactar de manera directa con la autoridad implicada. Una de las principales ventajas del proyecto es la monitorización y seguimiento de las incidencias de los usuarios. De este modo, FMS contacta con el usuario que reportó la incidencia pasadas cuatro semanas a fin de comprobar si la incidencia ha sido subsanada, y analizando el estado de la misma. El proceso de autenticación de los usuarios puede efectuarse de dos modos [38]: de manera anónima o por medio de un perfil (permite efectuar el seguimiento de las incidencias reportadas por el propio usuario).

La parte negativa de esta iniciativa es que en muchos casos las incidencias permanecen en estado "pendiente de solución" durante largos periodos de tiempo, aunque los usuarios pueden actualizar su estado a cada momento. Nuevamente se emplean mapas para facilitar la localización de las incidencias.

### 5.8.4 CrowdSC [39]

CrowdSC es un proyecto que tiene el objetivo de responder preguntas como qué carreteras se encuentran dañadas en un determinado lugar o qué áreas de una ciudad se encuentran en peor estado de mantenimiento [39]. De este modo el ayuntamiento de la ciudad en cuestión consulta a los ciudadanos a fin de conocer el estado de una ubicación en concreto. Los autores dividen el proceso global de la aplicación en tres subprocesos: recopilación de datos, selección de datos y evaluación de datos. La fase de captura o recolección de datos consiste en solicitar a unos pocos ciudadanos un conjunto de imágenes de carreteras en mal estado de asfaltado o casuísticas similares. Únicamente se encuesta a unos pocos usuarios a fin de no obtener información sesgada o con carencia de objetividad. En la sergunda fase otro conjunto de usuarios seleccionan la imagen que consideran más representativa acerca del estado de la incidencia. La fase final de evaluación consiste en que otros usuarios valoren la prioridad de cada incidencia dentro del conjunto, basándose en la imagen elegida en la fase intermedia.

Con este planteamiento se ha diseñado un *framework* llamado CrowdSC que implementa diferentes estrategias para manejar los procesos, obteniendo como salida todas las incidencias reportadas sobre un mapa y clasificadas de acuerdo a tres posibles estados: sin daños, dañado o muy dañado.

Figura 6: Vista de mapa de CrowdSC

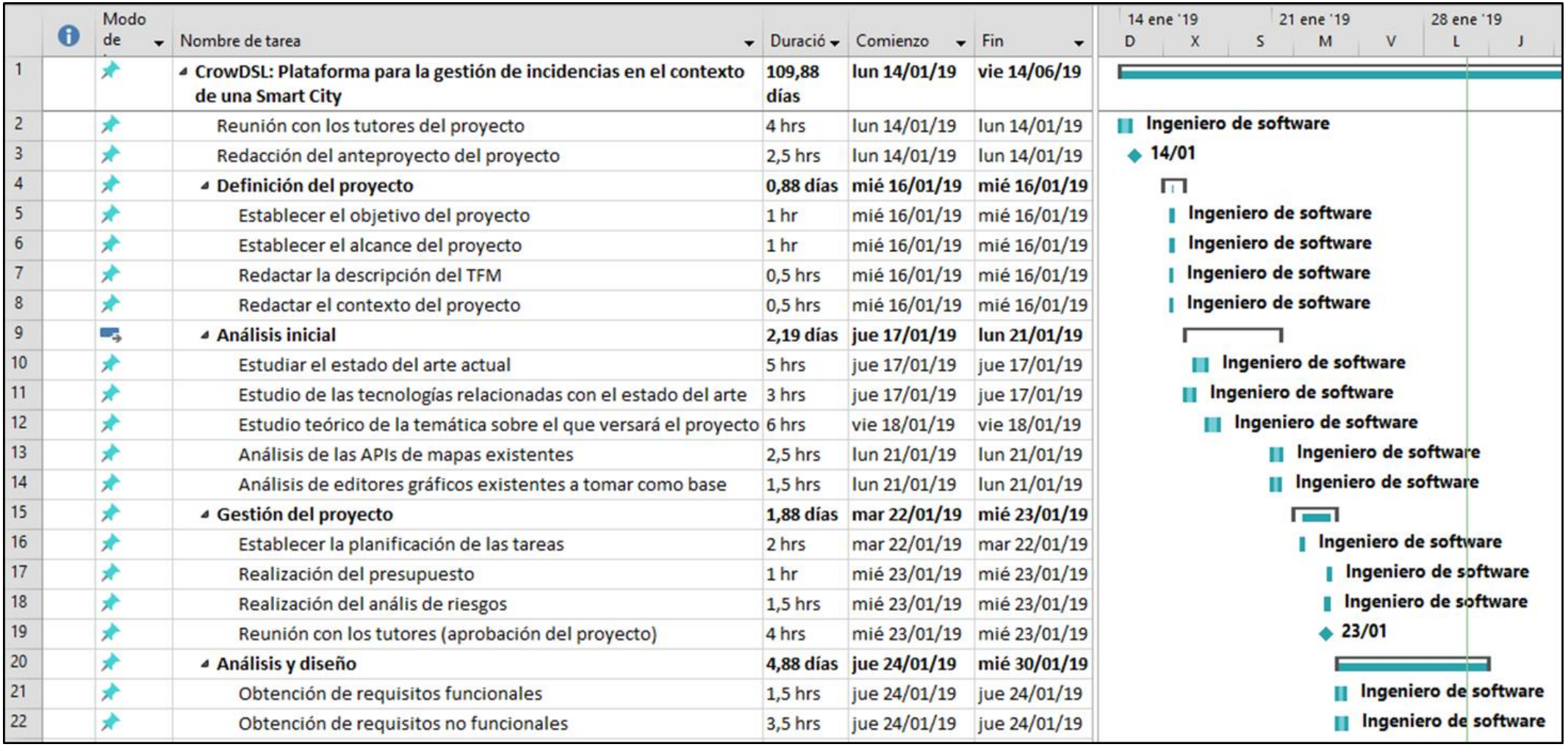
Una vez analizadas las plataformas y sistemas que se han expuesto en el presente apartado, se estudian las diferencias existentes entre ellas con respecto a la solución que se plantea en nuestra investigación. Existen características y funcionalidades comunes a todos los proyectos analizados, como es el caso del empleo de mapas para efectuar la representación de las incidencias. De igual modo, muchas de las soluciones emplean un perfil de usuario a fin de poder monitorizar toda la actividad del usuario en la plataforma. Donde existe una mayor diversidad de enfoques o planteamientos es en la naturaleza de las incidencias o eventos que se gestionan a través de cada plataforma. En algunas de ellas se reportan desperfectos en vías o espacios públicos, en otras acciones propias de la circulación de vehículos, en otras barreras arquitectónicas, etc.

La principal innovación que aporta el proyecto es el empleo de un DSL para la captura de los datos de los elementos que intervienen en la incidencia. De este modo, se obtienen gran parte de las ventajas que aporta el uso de MDE, como por ejemplo el hecho de que se permita facilitar la reusabilidad e interoperabilidad así como favorecer la integración de la solución con otros sistemas que pudiesen estar implantados en las ciudades. También se contempla la parametrización de componentes, como es el caso del mapa sobre el que se representarán cada una de las incidencias. Se permiten utilizar diferentes tipologías de mapas, especificar el *zoom* de estos, radios de peligrosidad, etc.

# 6 Planificación de tareas (inicial)

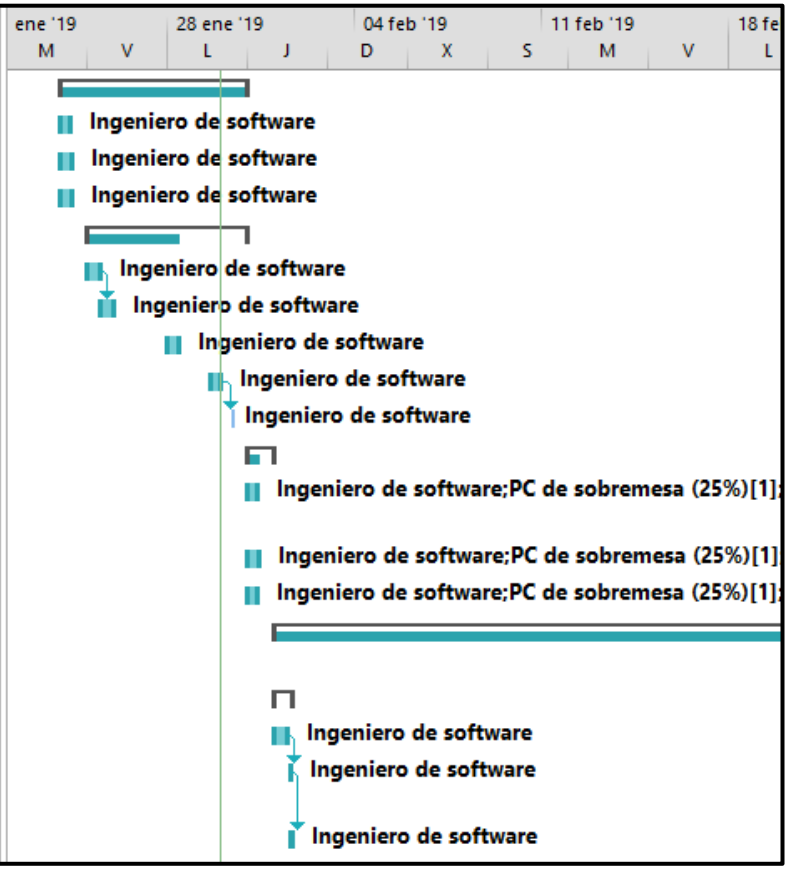
A continuación se exponen el conjunto de tareas en las que se ha subdividido tanto el desarrollo del proyecto como la parte de gestión. Se incorpora, por una parte, el desglose de las tareas (duración y fechas de comienzo y finalización) y por otra parte el diagrama de Gantt asociado a la investigación.

## 6.1 DESGLOSE DE TAREAS

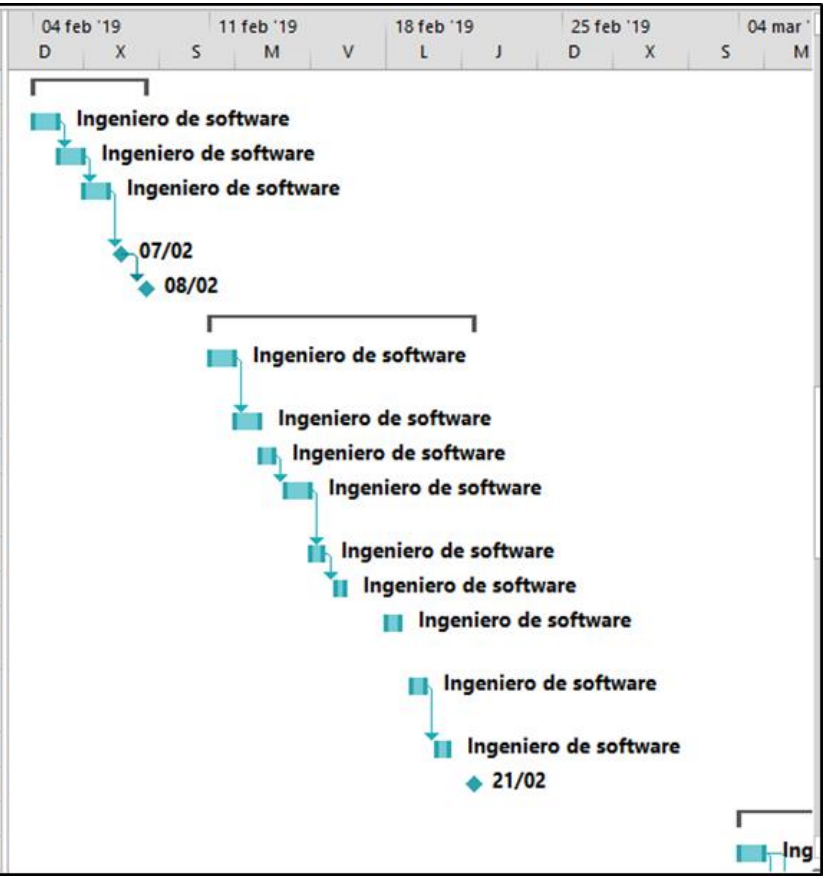


| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 1 | | **CrowDSL: Plataforma para la gestión de incidencias en el contexto de una Smart City** | **109,88 días** | **lun 14/01/19** | **vie 14/06/19** |
| 2 | | Reunión con los tutores del proyecto | 4 hrs | lun 14/01/19 | lun 14/01/19 |
| 3 | | Redacción del anteproyecto del proyecto | 2,5 hrs | lun 14/01/19 | lun 14/01/19 |
| 4 | | **Definición del proyecto** | **0,88 días** | **mié 16/01/19** | **mié 16/01/19** |
| 5 | | Establecer el objetivo del proyecto | 1 hr | mié 16/01/19 | mié 16/01/19 |
| 6 | | Establecer el alcance del proyecto | 1 hr | mié 16/01/19 | mié 16/01/19 |
| 7 | | Redactar la descripción del TFM | 0,5 hrs | mié 16/01/19 | mié 16/01/19 |
| 8 | | Redactar el contexto del proyecto | 0,5 hrs | mié 16/01/19 | mié 16/01/19 |
| 9 | | **Análisis inicial** | **2,19 días** | **jue 17/01/19** | **lun 21/01/19** |
| 10 | | Estudiar el estado del arte actual | 5 hrs | jue 17/01/19 | jue 17/01/19 |
| 11 | | Estudio de las tecnologías relacionadas con el estado del arte | 3 hrs | jue 17/01/19 | jue 17/01/19 |
| 12 | | Estudio teórico de la temática sobre el que versará el proyecto | 6 hrs | vie 18/01/19 | vie 18/01/19 |
| 13 | | Análisis de las APIs de mapas existentes | 2,5 hrs | lun 21/01/19 | lun 21/01/19 |
| 14 | | Análisis de editores gráficos existentes a tomar como base | 1,5 hrs | lun 21/01/19 | lun 21/01/19 |
| 15 | | **Gestión del proyecto** | **1,88 días** | **mar 22/01/19** | **mié 23/01/19** |
| 16 | | Establecer la planificación de las tareas | 2 hrs | mar 22/01/19 | mar 22/01/19 |
| 17 | | Realización del presupuesto | 1 hr | mié 23/01/19 | mié 23/01/19 |
| 18 | | Realización del anális de riesgos | 1,5 hrs | mié 23/01/19 | mié 23/01/19 |
| 19 | | Reunión con los tutores (aprobación del proyecto) | 4 hrs | mié 23/01/19 | mié 23/01/19 |
| 20 | | **Análisis y diseño** | **4,88 días** | **jue 24/01/19** | **mié 30/01/19** |
| 21 | | Obtención de requisitos funcionales | 1,5 hrs | jue 24/01/19 | jue 24/01/19 |
| 22 | | Obtención de requisitos no funcionales | 3,5 hrs | jue 24/01/19 | jue 24/01/19 |

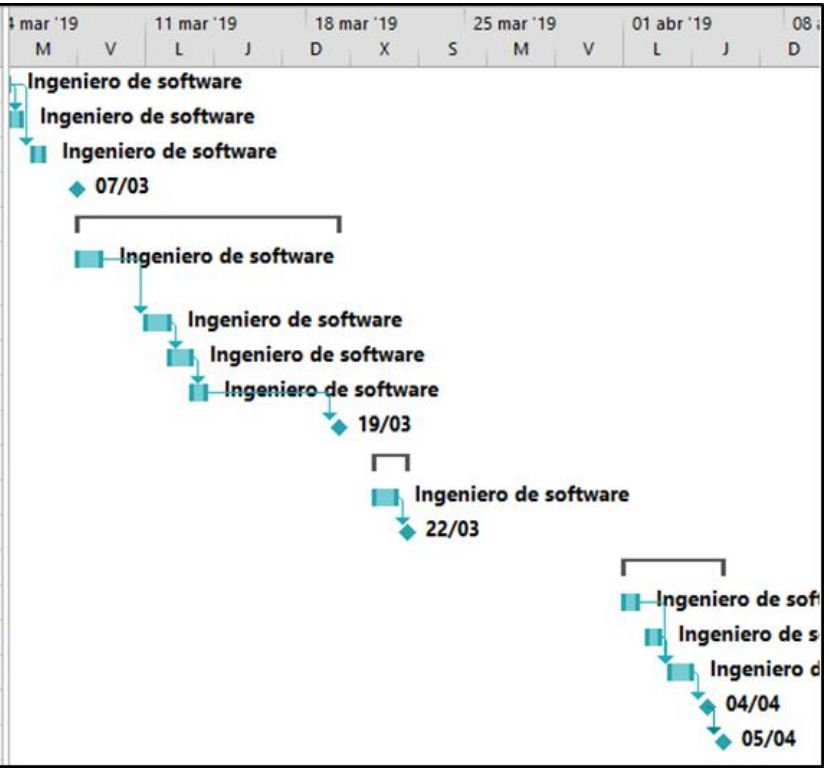
| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 20 | | **Análisis y diseño** | **4,88 días** | **jue 24/01/19** | **mié 30/01/19** |
| 21 | | Obtención de requisitos funcionales | 1,5 hrs | jue 24/01/19 | jue 24/01/19 |
| 22 | | Obtención de requisitos no funcionales | 3,5 hrs | jue 24/01/19 | jue 24/01/19 |
| 23 | | Definición de los casos de uso | 3 hrs | jue 24/01/19 | jue 24/01/19 |
| 24 | | **Definición de la arquitectura de la aplicación** | **3,88 días** | **vie 25/01/19** | **mié 30/01/19** |
| 25 | | Establecer la división en módulos | 4 hrs | vie 25/01/19 | vie 25/01/19 |
| 26 | | Establecer las dependencias entre módulos | 4 hrs | vie 25/01/19 | vie 25/01/19 |
| 27 | | Establecer la trazabilidad con los RF y RNF | 2,5 hrs | lun 28/01/19 | lun 28/01/19 |
| 28 | | Establecer los prototipos de las interfaces | 6 hrs | mar 29/01/19 | mar 29/01/19 |
| 29 | | Reunión de aprobación del análisis con tutores | 4 hrs | mié 30/01/19 | mié 30/01/19 |
| 30 | | **Preparación del entorno** | **0,88 días** | **jue 31/01/19** | **jue 31/01/19** |
| 31 | | Instalación y configuración de los lenguajes de programación empleados | 2 hrs | jue 31/01/19 | jue 31/01/19 |
| 32 | | Instalación y gestión de IDEs y otras herramientas de gestión | 2,5 hrs | jue 31/01/19 | jue 31/01/19 |
| 33 | | Instalación de motores de bases de datos | 1,5 hrs | jue 31/01/19 | jue 31/01/19 |
| 34 | | **Desarrollo** | **45,88 días** | **vie 01/02/19** | **vie 05/04/19** |
| 35 | | **Diseño del metamodelo de la aplicación** | **0,69 días** | **vie 01/02/19** | **vie 01/02/19** |
| 36 | | Definición de las entidades del modelo | 5 hrs | vie 01/02/19 | vie 01/02/19 |
| 37 | | Definición de las relaciones y cardinalidades entre entidades | 1 hr | vie 01/02/19 | vie 01/02/19 |
| 38 | | Definición de las restricciones en entidades y atributos | 0,5 hrs | vie 01/02/19 | vie 01/02/19 |



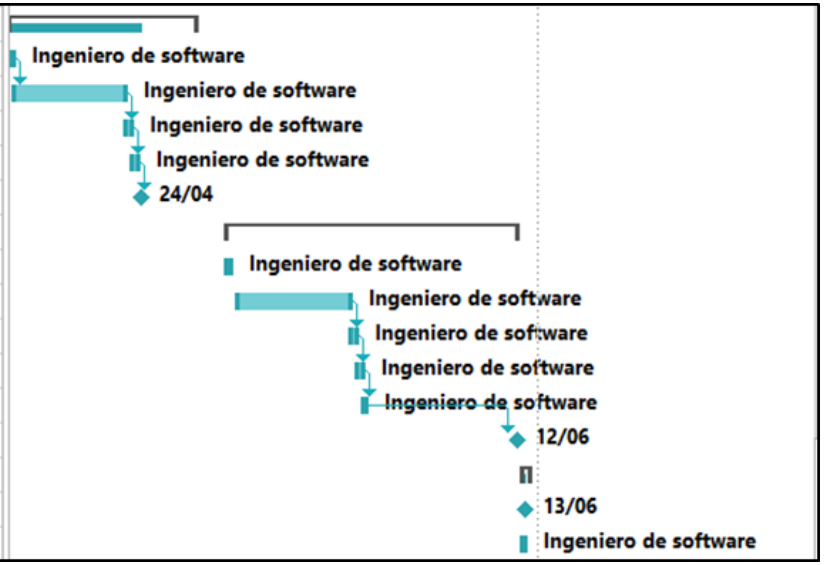
| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 39 | | **Desarrollo del validador de los modelos generados** | **4,38 días** | **lun 04/02/19** | **vie 08/02/19** |
| 40 | | Serialización del metamodelo en formato XML Schema | 8 hrs | lun 04/02/19 | lun 04/02/19 |
| 41 | | Serialización de los modelos generados en formato JSON | 8 hrs | mar 05/02/19 | mar 05/02/19 |
| 42 | | Validación de los modelos generados contra el metamodelo | 8 hrs | mié 06/02/19 | mié 06/02/19 |
| 43 | | Pruebas unitarias de diseño iniciales | 5 hrs | jue 07/02/19 | jue 07/02/19 |
| 44 | | Reunión de validación inicial | 4 hrs | vie 08/02/19 | vie 08/02/19 |
| 45 | | **Creación del editor** | **8,44 días** | **lun 11/02/19** | **jue 21/02/19** |
| 46 | | Definición de la representación gráfica para cada entidad (icono) | 7 hrs | lun 11/02/19 | lun 11/02/19 |
| 47 | | Desarrollo de la interfaz gráfica del editor | 8 hrs | mar 12/02/19 | mar 12/02/19 |
| 48 | | Configuración y parametrización del editor base | 6 hrs | mié 13/02/19 | mié 13/02/19 |
| 49 | | Implemetación de las restricciones básicas para definir las relaciones entre entidades | 7 hrs | jue 14/02/19 | jue 14/02/19 |
| 50 | | Desarrollo de la paleta para la gestión de los componentes | 4 hrs | vie 15/02/19 | vie 15/02/19 |
| 51 | | Creación del sistema de log para la generación del modelo | 1,5 hrs | sáb 16/02/19 | sáb 16/02/19 |
| 52 | | Definición y desarrollo del componente para la gestión de las propiedades de cada entidad | 6 hrs | lun 18/02/19 | lun 18/02/19 |
| 53 | | Desarrollo del componente para la visualización del modelo generado en formato JSON | 6 hrs | mar 19/02/19 | mar 19/02/19 |
| 54 | | Pruebas unitarias y de integración del módulo del editor | 4,5 hrs | mié 20/02/19 | mié 20/02/19 |
| 55 | | Reunión de validación con los directores | 4,5 hrs | jue 21/02/19 | jue 21/02/19 |
| 56 | | **Desarrollo del sistema de autenticación y registro** | **3,88 días** | **lun 04/03/19** | **jue 07/03/19** |
| 57 | | Registro de nuevos usuarios | 7 hrs | lun 04/03/19 | lun 04/03/19 |



| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 57 | | Registro de nuevos usuarios | 7 hrs | lun 04/03/19 | lun 04/03/19 |
| 58 | | Auntenticación para usuarios ya registrados | 5 hrs | mar 05/03/19 | mar 05/03/19 |
| 59 | | Actualización de los datos del perfil del usuario | 4,5 hrs | mié 06/03/19 | mié 06/03/19 |
| 60 | | Pruebas unitarias e integración del modo de autenticación | 8 hrs | jue 07/03/19 | jue 07/03/19 |
| 61 | | **Representación de las incidencias sobre mapas** | **7,38 días** | **vie 08/03/19** | **mar 19/03/19** |
| 62 | | Parametrización de la visualización del mapa de acuerdo a las especificaciones del usuario | 1 día | vie 08/03/19 | vie 08/03/19 |
| 63 | | Ubicación y etiquetado de cada incidencia por tipo | 7 hrs | lun 11/03/19 | lun 11/03/19 |
| 64 | | Visualización de los elementos asociados a la incidencia | 8 hrs | mar 12/03/19 | mar 12/03/19 |
| 65 | | Pruebas unitarias y de integración del módulo de mapas | 6 hrs | mié 13/03/19 | mié 13/03/19 |
| 66 | | Reunión de validación con los tutores | 4 hrs | mar 19/03/19 | mar 19/03/19 |
| 67 | | **Visualización y filtrado de incidencias** | **1,38 días** | **jue 21/03/19** | **vie 22/03/19** |
| 68 | | Filtrado de incidencias por los criterios disponibles | 8 hrs | jue 21/03/19 | jue 21/03/19 |
| 69 | | Pruebas unitarias y de integración del módulo de consulta | 4 hrs | vie 22/03/19 | vie 22/03/19 |
| 70 | | **Valoración y comentarios de las incidencias** | **4,06 días** | **lun 01/04/19** | **vie 05/04/19** |
| 71 | | Registro de comentarios relativos a incidencia | 6 hrs | lun 01/04/19 | lun 01/04/19 |
| 72 | | Registro de valoraciones de la incidencia | 6 hrs | mar 02/04/19 | mar 02/04/19 |
| 73 | | Visualización de comentarios y valoraciones | 7 hrs | mié 03/04/19 | mié 03/04/19 |
| 74 | | Pruebas unitarias y de integración del módulo de opiniones | 6 hrs | jue 04/04/19 | jue 04/04/19 |
| 75 | | Reunión de validación final de desarrollo | 1,5 hrs | vie 05/04/19 | vie 05/04/19 |



| | | | | | | |
|---|---|---|---|---|---|---|
| 76 | | **Artículo de investigación** | **17,88 días** | **lun 08/04/19** | **mié 01/05/19** | |
| 77 | | Pruebas de usabilidad y accesibilidad con usuarios reales | 6 hrs | lun 08/04/19 | lun 08/04/19 | |
| 78 | | Redacción del artículo | 10,38 días | lun 08/04/19 | lun 22/04/19 | |
| 79 | | Reunión de revisión de la escritura del artículo | 7 hrs | lun 22/04/19 | mar 23/04/19 | |
| 80 | | Estudio de la revista a la que realizar el envío | 6 hrs | mar 23/04/19 | mié 24/04/19 | |
| 81 | | Envío del artículo | 4 hrs | mié 24/04/19 | mié 24/04/19 | |
| 82 | | **Integración e implantación** | **27,38 días** | **lun 06/05/19** | **mié 12/06/19** | |
| 83 | | Diseño y desarrollo de casos de uso de ejemplo | 6 hrs | lun 06/05/19 | lun 06/05/19 | |
| 84 | | Redacción de la memoria final | 10,88 días | mar 07/05/19 | mar 21/05/19 | |
| 85 | | Revisión final de la memoria | 7 hrs | mar 21/05/19 | mié 22/05/19 | |
| 86 | | Redacción de la presentación del proyecto ante tribunal | 7 hrs | mié 22/05/19 | jue 23/05/19 | |
| 87 | | Ensayo de la exposicion oral | 6 hrs | jue 23/05/19 | jue 23/05/19 | |
| 88 | | Presentación del proyecto ante tribunal | 4 hrs | mié 12/06/19 | mié 12/06/19 | |
| 89 | | **Cierre del proyecto** | **0,88 días** | **jue 13/06/19** | **jue 13/06/19** | |
| 90 | | Reunión final de evaluación | 3,5 hrs | jue 13/06/19 | jue 13/06/19 | |
| 91 | | Estudio del estado de publicación del artículo de investigación | 3 hrs | jue 13/06/19 | jue 13/06/19 | |




## 6.2 RECURSOS

| | Nombre del recurso | Tipo | Etiqueta de material | Iniciales | Grupo | Capacidad máxima | Tasa estándar | Tasa horas extra | Costo/Uso | Acumular | Calendario base | Cód |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Ingeniero de software | Trabajo | | IS | SOFT | 150% | 45,00 €/hr | 0,00 €/hr | 0,00 € | Prorrateo | Calendario TFM | IS |
| 2 | PC de sobremesa (25%) | Material | | PCSOBR | | | 337,50 € | | 0,00 € | Prorrateo | | |
| 3 | PC portátil (25%) | Material | | PORT | | | 287,50 € | | 0,00 € | Prorrateo | | |
| 4 | Servidor para despliegue y pruebas | Material | | ANDR | | | 235,00 € | | 0,00 € | Prorrateo | | |

# 7 Presupuesto (inicial)

En relación a los costes que se deberán asumir durante el desarrollo del proyecto se han realizado dos presupuestos, el presupuesto interno de empresa y el que se presentaría al cliente.

*NOTA*: Todos los conceptos se muestran sin aplicar el impuesto del IVA

## 7.1 PRESUPUESTO INTERNO DE EMPRESA

Para el presupuesto interno comenzamos por calcular el coste por unidad de obra para cada uno de los módulos del proyecto.

### Costes por unidad de obra

| Módulo | Medición (h) | Coste unit. (€) | Coste total (€) |
|---|---|---|---|
| Definición del proyecto | 9,5 | 45 € | 427,50 € |
| Análisis inicial | 18 | 45 € | 810,00 € |
| Gestión del proyecto | 8,5 | 45 € | 382,50 € |
| Análisis y diseño | 28,5 | 45 € | 1282,50 € |
| Preparación del entorno | 6 | 45 € | 270,00 € |
| Diseño del metamodelo de la aplicación | 6,5 | 45 € | 292,50 € |
| Desarrollo del validador de los modelos generados | 33 | 45 € | 1485,00 € |
| Creación del editor | 54,5 | 45 € | 2452,50 € |
| Desarrollo del sistema de autenticación y registro | 24,5 | 45 € | 1102,50 € |
| Representación de las incidencias sobre mapas | 25 | 45 € | 1125,00 € |
| Visualización y filtrado de incidencias | 12 | 45 € | 540,00 € |
| Valoración y comentarios de las incidencias | 26,5 | 45 € | 1192,50 € |
| Artículo de investigación | 143 | 45 € | 6435,00 € |
| Redacción de la memoria | 120 | 45 € | 5400,00 € |

| | | | |
|---|---|---|---|
| Integración y preparativos finales | 32,5 | 45 € | 1462,50 € |
| **TOTAL** | | **548 h** | **24660,00€** |

El coste unitario por hora se ha calculado teniendo en cuenta el perfil del desarrollador del proyecto (Ingeniero de software) y atendiendo a los costes que deberá asumir la empresa en cuanto al pago de impuestos, salarios y otros conceptos relativos al trabajador.

Calculamos ahora los costes de licencias de software, costes de hardware y costes indirectos.
Para los diferentes elementos se establece un tiempo de amortización de 5 años. En cuanto a Sistemas Operativos y paquetes de ofimática se aplica un 10% y un 25% respectivamente dado que las licencias se entienden adquiridas para más proyectos y por tanto se aplica la parte proporcional.

### Coste de software

| Concepto | Unidades | Coste unitario (€) | Coste total (€) |
|---|---|---|---|
| Microsoft Windows 10 | 1 (10%) | 127 | 12,70 € |
| Microsoft Office 365 | 1 (25%) | 110 | 27,50 € |
| PyCharm Community Edition (Jetbrains IDE) | 1 | 0 | 0,00 € |
| Python 3.6 | 1 | 0 | 0,00 € |
| Flask framework | 1 | 0 | 0,00 € |
| BDD Mongo DB | 1 | 0 | 0,00 € |
| **TOTAL** | | | **40,20 €** |

### Costes de hardware

| Concepto | Unidades | Coste unitario (€) | Coste total (€) |
|---|---|---|---|
| Ordenador de sobremesa | 1 (25%) | 1350 € | 337,50 € |
| Ordenador portátil | 1 (25%) | 1150 € | 287,50 € |
| Servidor para despliegue y pruebas | 1 | 235 € | 235,00 € |
| **TOTAL** | | | **859,00 €** |

### Costes indirectos

| Concepto | Unidades | Coste unitario (€) | Coste total (€) |
|---|---|---|---|
| Electricidad | 6 | 45 € | 270,00 € |
| Agua | 6 | 7 | 42,00 € |
| Internet y comunicaciones | 6 | 29 | 174,00 € |
| Desplazamiento | 6 | 37 | 222,00 € |
| Oficina | 6 | 4,25 | 25,50 € |
| Seguros de hardware | 6 | 6 | 36,00 € |
| **TOTAL** | | | **769,50 €** |

| | |
|---|---|
| Subtotal | 26328,70 € |
| Beneficio (15%) | 3949,31 € |
| IVA (21%) | 6358,38 € |
| **TOTAL** | **36636,39 €** |

## 7.2 PRESUPUESTO DE CLIENTE

| **Concepto** | **Coste total (€)** |
|---|---|
| Diseño del metamodelo de la aplicación | 1085,65 € |
| Desarrollo del validador de los modelos generados | 2278,15 € |
| Creación del editor | 3245,65 € |
| Desarrollo del sistema de autenticación y registro | 1895,65 € |
| Representación de las incidencias sobre mapas | 1918,15 € |
| Visualización y filtrado de incidencias | 1333,15 € |
| Valoración y comentarios de las incidencias | 1985,65 € |
| Artículo de investigación | 7228,15 € |
| Redacción de la memoria | 6193,15 € |
| Integración y preparativos finales | 2255,65 € |
| ***Subtotal*** | **29419,01 €** |
| Hardware | 859,00 € |
| ***Subtotal*** | **30278,01 €** |
| ***IVA (21%)*** | **6358,38 €** |
| **TOTAL** | **36636,39 €** |

# 8 CrowDSL: Diseño del metamodelo

CrowDSL es una plataforma diseñada para la gestión de las incidencias que se originan a diario en el ámbito de una Smart City. Dentro de la citada plataforma se contemplan varios procesos que permiten seguir los ciclos o estados que atraviesan dichas incidencias desde que son reportadas por los ciudadanos hasta que son resultas por la autoridad competente. A continuación se expone la solución que se ha planteado de acuerdo al contexto expuesto, comenzando por el diseño del metamodelo.

## 8.1 METAMODELO DEL DOMINIO DE CONOCIMIENTO

Como ya se comentó anteriormente se estableció como punto de partida la definición del metamodelo del dominio de conocimiento. Para llevar a cabo la definición de este metamodelo se empleó como meta-metamodelo Ecore [26].

Finalizado el proceso de abstracción para el modelado de las entidades que conforman el dominio del problema se obtuvo como resultado el metamodelo que se presenta en la Figura 7:Metamodelo de la aplicación. El metamodelo se centra de manera más directa en la relevancia de los componentes más usados del lenguaje final. Dichos componentes permiten representar al usuario que realiza el reporte de la incidencia, la incidencia con sus datos asociados, las entidades físicas que intervienen en ella y otro tipo de elementos que posibilitan completar la información que introduce el usuario, como por ejemplo, el empleo de imágenes, videos, tweets, etc.

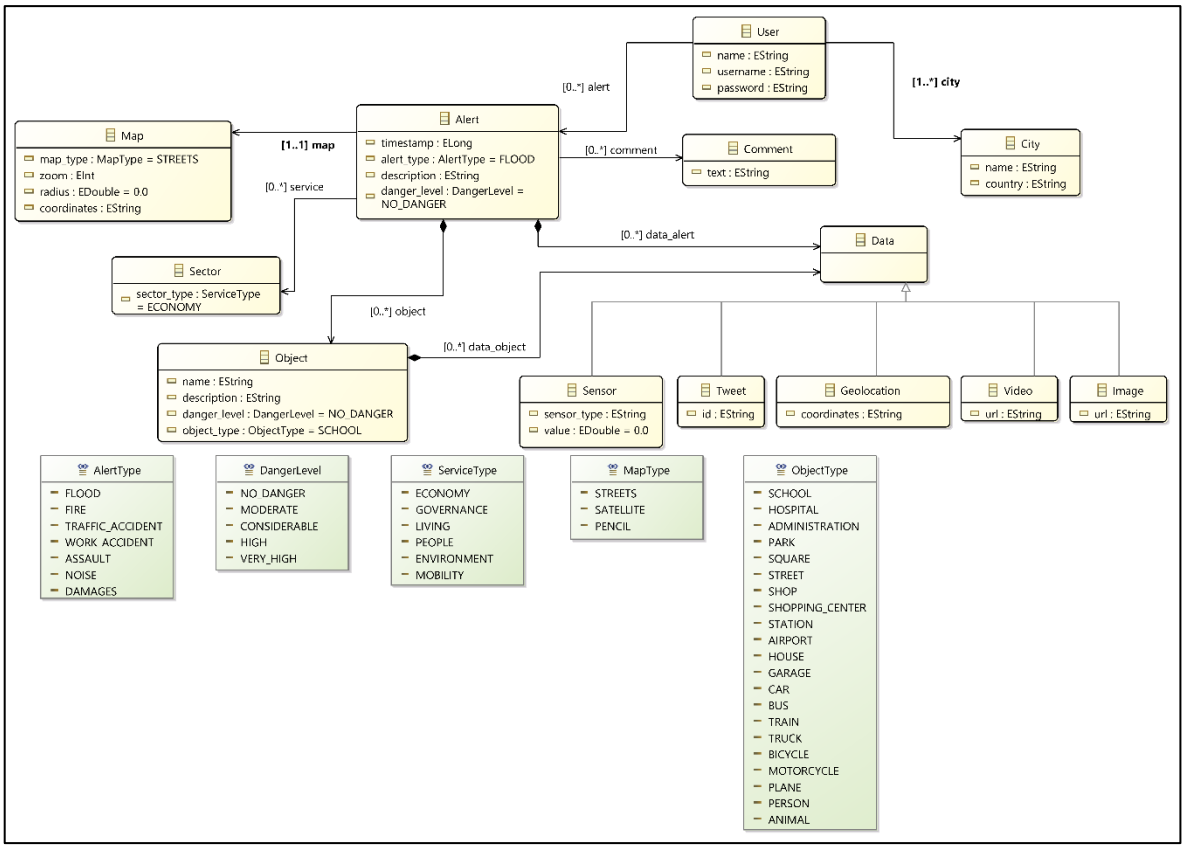



Figura 7:Metamodelo de la aplicación

De acuerdo a este metamodelo la aplicación está formada por usuarios (“User”), los cuales residen o emplean parte de su tiempo en una ciudad (“City”). Cada usuario tiene la opción de reportar diferentes alertas (“Alert”) que él mismo presencia en su día a día. Cada una de estas alertas se representa sobre un mapa (“Map”) y se asocia a uno de los seis sectores de los que se compone una Smart City según [22]. Las alertas se clasifican de igual modo en una de las siguientes siete categorías: inundaciones, incendios, accidentes de tráfico, accidentes laborales, agresiones, niveles anómalos de ruido y daños producidos en las vías públicas.

Mediante los objetos (“Object”) se modelan todas las entidades físicas que podrían intervenir en una incidencia de manera general, pudiendo especificarse descripciones así como el nivel de daño o de riesgo que podrían generar esos elementos para la seguridad de forma individual. A fin de aumentar el grado de realismo de las incidencias que van siendo reportadas, se permiten asociar elementos multimedia (“Data”) tanto a las incidencias como a los objetos. Estos elementos multimedia están compuestos por: imágenes, videos, geolocalización, captaciones de sensores y tweets de la red social Twitter. De manera análoga, se ofrece a cada usuario la posibilidad de incluir comentarios (“Comment”) en las incidencias que reportan el resto de los usuarios o él mismo, tratando de aglutinar la mayor cantidad de información para cada uno de los reportes.

# 9 CrowDSL: Diseño del DSL

Con la ayuda del Lenguaje de Dominio Específico (DSL) que se ha construido y que se expondrá en esta sección, el usuario podrá modelar todas las entidades, escenarios y sectores de la ciudad que intervienen en el suceso de una incidencia. Para llevar a cabo el reporte de las citadas incidencias se ha implementado un editor web que permita al usuario familiarizarse rápidamente con el entorno e ir adquiriendo de manera paulatina los conocimientos necesarios para operar de manera normalizada con el DSL. Una vez las incidencias son reportadas y validadas, estas son publicadas de manera que sirvan de ayuda al resto de los ciudadanos que conviven en la misma ciudad, eso sí, realizándose previamente un análisis de la información generada a fin de asegurar el cumplimiento de las políticas de protección de datos que pudieran ser vulneradas. Las incidencias publicadas pueden ser consultadas desde la misma plataforma, desde donde se dispone de toda la información, elementos multimedia, geolocalización de la incidencia y un apartado de comentarios para el que el resto de usuarios pueda mostrar sus opiniones.

## 9.1 ARQUITECTURA PROPUESTA

De acuerdo a lo expuesto en la Figura 8: Arquitectura de CrowDSL el prototipo se estructura entorno a tres capas que contemplan el total de la funcionalidad de la plataforma. La primera de las capas, “Data Definition”, abarca el proceso de definición de los modelos por parte del usuario. Esta capa está compuesta por el Lenguaje de Dominio Específico (DSL) con el que se opera por medio del editor Web diseñado para tal fin. En esta primera fase se obtiene una representación de la información en formato XML de lo que el usuario modela por medio de los diferentes elementos que conforman el lenguaje gráfico. Esta información se traspasa a la segunda capa, “Data Processing” donde se parte de ese XML para representar un árbol en memoria que se corresponde con la estructura del fichero recibido de la capa superior. Esta segunda capa tiene el objetivo de validar que la estructura del fichero cumple las especificaciones definidas en el metamodelo que se expondrá a continuación en cuanto a relación entre las entidades y tipos de datos definidos por el usuario. Una vez superado todo el proceso de validación, las incidencias pueden ser almacenadas en la base de datos de la aplicación. Para este almacenamiento se emplea una base de datos

capaz de almacenar documentos en formato JSON. Una vez se almacenan las incidencias en la base de datos estas se harán publicas a través de la tercera capa de la arquitectura, "Data Visualisation", que permite al resto de los usuarios de la aplicación consultar toda la información que se ofrece adjunta. Se contemplan múltiples formas de información para esta tercera capa: por una parte la información especificada por el usuario en modo texto, los elementos multimedia que incorpora el usuario que reportó la incidencia (imágenes, videos, tweets, geolocalización o captaciones de sensores), un mapa parametrizado por el usuario que permite obtener la ubicación de la incidencia y, finalmente, un apartado de comentarios para que cada usuario valore de manera individual cualquier hecho reseñable acerca del suceso que se publica.

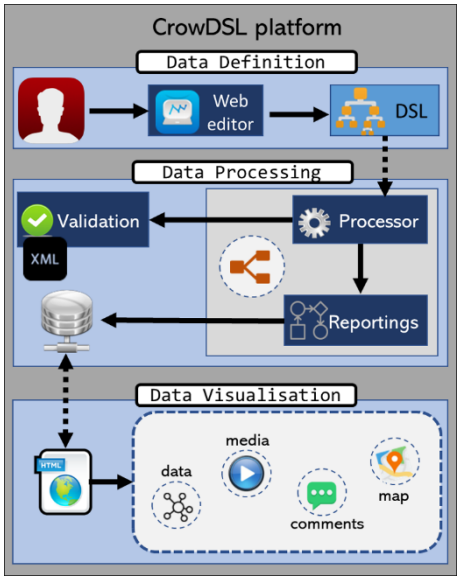



Figura 8: Arquitectura de CrowDSL

## 9.2 DEFINICIÓN DE LA SINTAXIS CONCRETA DE LOS ELEMENTOS

Una vez implementado el metamodelo y diseñada la arquitectura se continuó por definir la sintaxis concreta de los diferentes elementos a partir de la sintaxis abstracta. La tabla inferior muestra la correspondencia entre los dos tipos de sintaxis.

| **Sintaxis abstracta** | **Sintaxis concreta** | |
|---|---|---|
| | **Tipo** | **Representación gráfica** |
| **Usuario** | |  |
| **Ciudad** | |  |
| **Mapa** | |  |
| **Alerta** | Inundación |  |
| | Fuego |  |
| | Accidente de tráfico |  |
| | Accidente laboral |  |
| | Agresión |  |
| | Niveles altos de ruido |  |
| | Daños en espacios públicos |  |
| **Sector** | Economía |  |
| | Gobierno |  |
| | Calidad de vida |  |
| | Personas |  |
| | Medio ambiente |  |
| | Transporte |  |

| | | |
|---|---|---|
| **Objetos** | Persona |  |
| | Animal |  |
| | Coche |  |
| | Autobús |  |
| | Tren |  |
| | Camión |  |
| | Bicicleta |  |
| | Motocicleta |  |
| | Avion |  |
| | Administración |  |
| | Aeropuerto |  |
| | Garaje |  |
| | Hospital |  |
| | Casa |  |
| | Parque |  |
| | Colegio |  |
| | Tienda |  |
| | Centro comercial |  |
| | Plaza |  |
| | Estación |  |
| | Calle |  |

| | | |
|---|---|---|
| **Media** | Sensor |  |
| | Tweet |  |
| | Geolocalización |  |
| | Video |  |
| | Imagen |  |

Tabla 1: Sintaxis concreta y abstracta de CrowDSL

# 10 CrowDSL: Prototipo

A continuación se expondrá, de manera detallada, el funcionamiento de la plataforma, apoyándonos en la aplicación web diseñada tanto para el manejo del DSL como para la posterior consulta de las incidencias reportadas. A través de esta aplicación se expondrá la forma en que se opera con el lenguaje así como el proceso de tratamiento y validación de la información a través de las diferentes capas que conforman la arquitectura propuesta.

## 10.1 SISTEMA DE AUTENTICACIÓN

El reporte de cada incidencia debe ir asignado de manera obligatoria a un usuario, por lo que en todo momento se debe poder conocer qué información ha aportado cada usuario de la aplicación. Mediante este sistema se trata de conseguir tal objetivo, de manera que no se publique información anómina o sesgada que pueda tener fines distintos a aquellos para los que fue diseñada la plataforma.

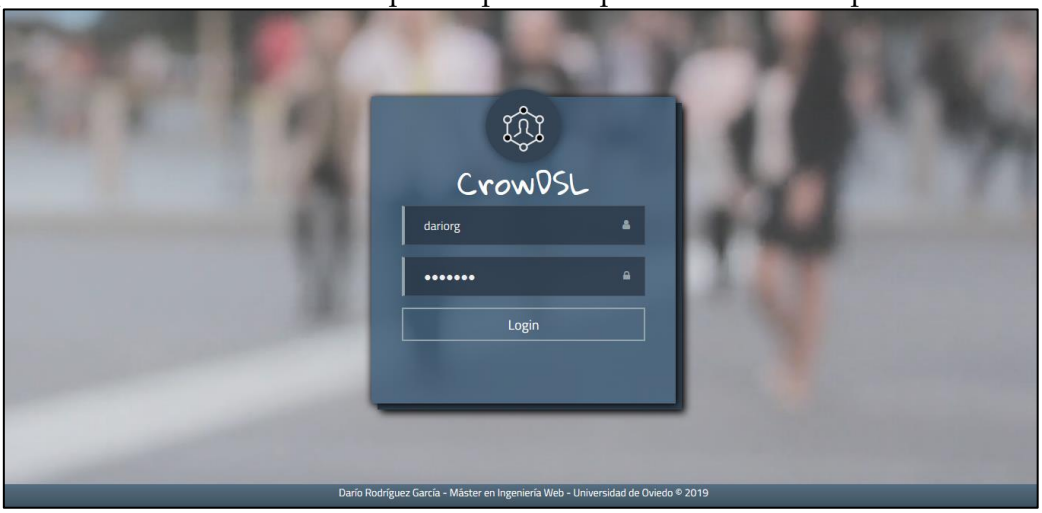



Figura 9: Interfaz de autenticación

## 10.2 EDITOR WEB GRÁFICO

Mediante el editor que se presenta en la **Error! Reference source not found.** se permite realizar el proceso de modelado de las incidencias detectadas por los usuarios para su posterior envío a la plataforma. El editor dispone de diferentes apartados para representar toda la información que se necesita para operar con el mismo. A continuación se exponen los diferentes componentes de los que consta el editor:

- **A - Menú lateral**: Permite seleccionar de la paleta de componentes aquellos que el usuario desea utilizar para el modelado de la incidencia. La paleta se distribuye en los diferentes elementos de los que consta el lenguaje: elementos generales, alertas, sectores de la ciudad, objetos y elementos multimedia.
- **B - Barra de herramientas**: Permite realizar diferentes operaciones sobre el editor gráfico. Dichas operaciones incluyen las funciones de cortar, copiar, pegar, borrar, hacer/deshacer, dimensionar y centrar.
- **C - Editor**: Mediante este componente se permite relacionar los diferentes elementos que se arrastran de la paleta de componentes.
- **D - Propiedades**: Permite visualizar las propiedades asociadas a cada uno de los elementos que se encuentran definidos en el editor. Para el ejemplo que se presenta se muestran las propiedades del mapa en el que se representa la incidencia.
- **E - Log**: Mediante este componente se muestra al usuario los errores surgidos de la validación de las incidencias que se encuentran editando. Se muestran tanto errores de tipado de datos como de enlace entre los elementos del lenguaje.
- **F - Estructura de la incidencia**: Una vez la incidencia se establece como válida su estructura se representa en formato JSON-LD para facilitar su futura representación.
- **G - Panel de botones**: Permite ejecutar la validación de la incidencia así como el envío de la misma cuando la validación es favorable.

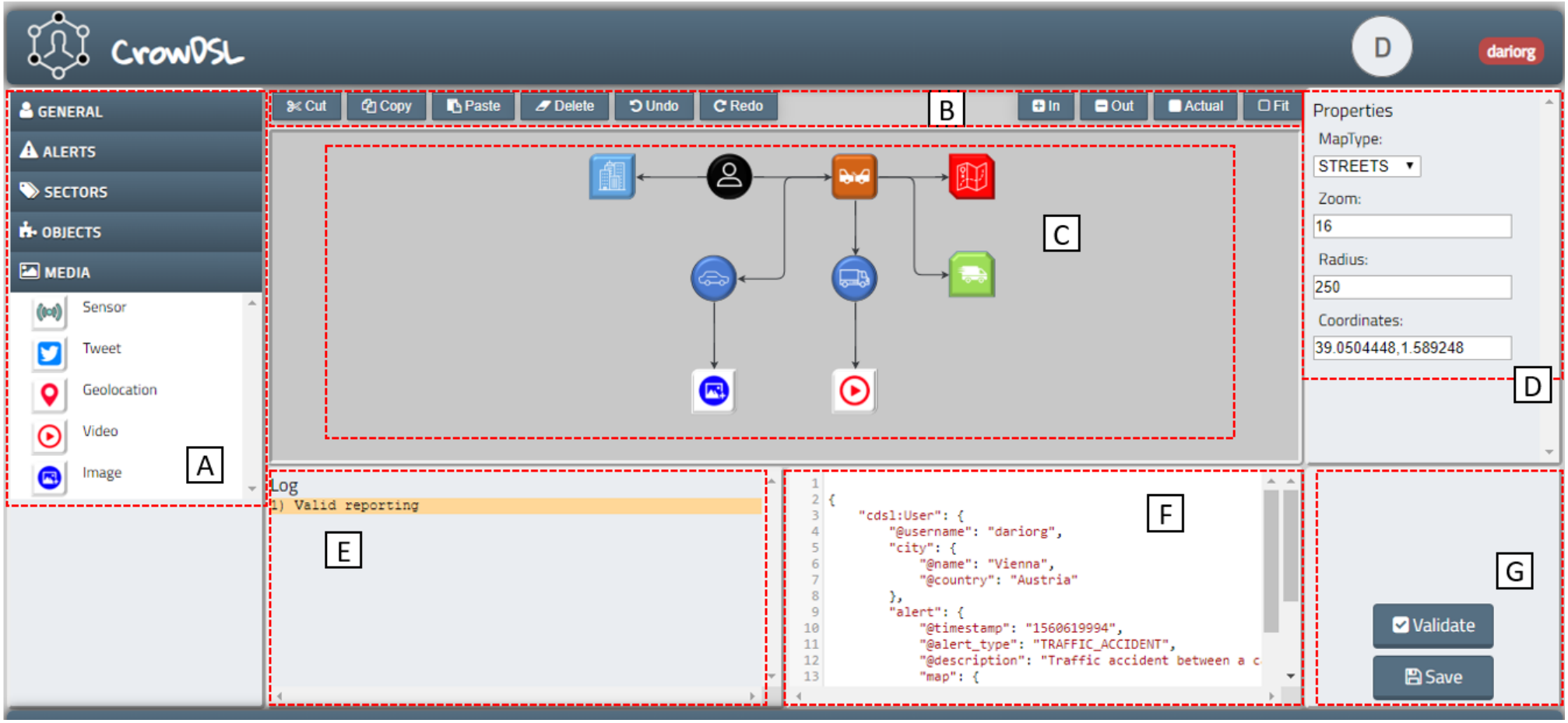



Figura 10: Editor web de CrowDSL

## 10.3 EL LENGUAJE DE DOMINIO ESPECÍFICO CROWDSL

Como ya se ha comentado en el apartado anterior, los nodos de los que se compone el lenguaje se estructuran en 5 apartados. Por una parte un conjunto de elementos generales que estarán presentes en todos los modelos generados por el usuario, que son: el usuario que realiza el envío (se muestra por defecto en el editor), la ciudad para la que se reporta la incidencia tratada y el mapa que permite georeferenciar y representar la incidencia.

En otro conjunto se agrupan los diferentes tipos de incidencias que se van a manejar en la plataforma. Se dispone de un conjunto de siete tipos de incidencias y de acuerdo a lo definido en el metamodelo siempre se asocian al usuario que realiza el reporte de esa incidencia.

Por otra parte se encuentran los 6 sectores clave en los que se divide una Smart City según [22]. De este modo podremos indicar a qué sector afecta el suceso de la alerta que se reporta. Nuevamente este elemento se asocia a un nodo incidencia.

El cuarto de los grupos de elementos engloba todas las entidades físicas que se contemplan en una primera versión del lenguaje. Se incluyen diferentes elementos que se podrían categorizar en seres vivos, medios de transporte y edificios y vías públicas. Estos elementos permiten describir las entidades que intervienen en una incidencia concreta.

Finalmente, se muestran los elementos multimedia que se pueden enlazar tanto a nodos objeto como a nodos alerta. Estos componentes se usan para asociar material gráfico a los datos de los reportes.

### 10.3.1 VALIDACIÓN Y ENVÍO DE LAS ALERTAS

En la captura de pantalla superior en la que se muestra el editor web se presenta el reporte de una incidencia. Esta incidencia se corresponde con un accidente de tráfico en el que intervienen un coche y un camión. En la Figura 11: Ejemplo de incidencia se muestra esta representación.

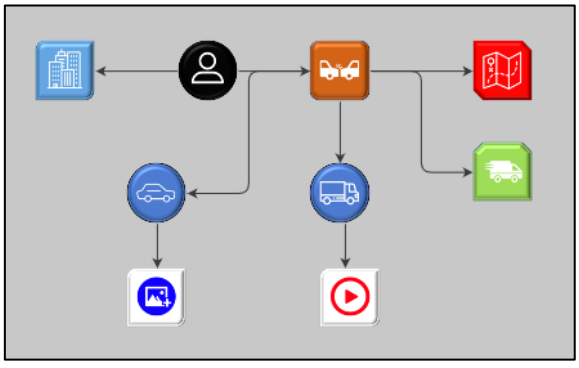


Figura 11: Ejemplo de incidencia

A medida que el usuario modela la relación entre los diferentes nodos que componen el lenguaje, internamente se genera una representación del modelo en un formato XML propio del *framework* JavaScript que se ha utilizado para el

desarrollo del editor. Cuando el usuario acciona el botón "Validar" se desencadena el proceso que permite llevar a cabo la validación de la estructura y los datos del modelo.

Partiendo del metamodelo de la plataforma se ha obtenido una representación de éste en formato XMI, utilizando un XML Schema para XMI. Una vez definido este Schema lo emplearemos para hacer las validaciones de los modelos que se vayan generando.

Por tanto, el primer paso que se lleva a cabo es efectuar una transformación entre el XML propio del *framework* a un formato XMI que nos permita validar dicho modelo contra el XML Schema para XMI obtenido. Esta conversión se materializa representando primeramente el XML de origen como un árbol en memoria en el que cada nodo se genera por medio de una clase del modelo que define la estructura del elemento. Partiendo de esa representación se genera un fichero XMI aplicando un patrón *Visitor* sobre el árbol a fin de poder efectuar la validación mencionada. A continuación se expone la representación en formato XMI de la incidencia con la que se trabaja en el ejemplo de la Figura 11: Ejemplo de incidencia.

```
<cdsl:User username="dariorg">
    <city name="Viena" country="Austria"></city>
    <alert timestamp="1558197718" alert_type="TRAFFIC_ACCIDENT" description="Choque frontal">
        <map map_type="STREETS" zoom="16" radius="250" coordinates="42.4353792,-8.8252416"></map>
        <service service_type="MOBILITY"></service>
        <object name="BMW" description="Serie 3" danger_level="MODERATE" object_type="CAR">
            <data_object url="https://db.com/image"></data_object>
        </object>
        <object name="Mercedes" description="Benz" danger_level="HIGH" object_type="TRUCK">
            <data_object url="https://db.com/video"></data_object>
        </object>
    </alert>
</cdsl:User>
```

Figura 12: Representación en formato XMI de una incidencia

Una vez obtenido el fichero que se muestra en la Figura 12: Representación en formato XMI de una incidencia, ya se estaría en condiciones de validar dicha estructura contra el XML Schema del modelo. Si se detecta algún error en el proceso de validación dichos errores se representan en el log del editor a fin de que el usuario pueda corregirlos para hacer el envío de la alerta de manera satisfactoria.

Si el proceso de validación concluye sin errores el usuario puede hacer el envío de la incidencia mediante el botón "Save". En el momento que la estructura y el contenido de una incidencia se valida se genera una representación de la misma en formato JSON-LD que facilita su almacenamiento en una base de datos NoSQL orientada a documentos con la que interactúa la plataforma. En la Figura 13: Representación en formato JSON-LD de la incidenciase muestra la representación en este formato de la incidencia del ejemplo.

```
{
  "cdsl:User": {
    "@username": "dariorg",
    "city": {
      "@name": "Viena",
      "@country": "Austria"
    },
    "alert": {
      "@timestamp": "1558199210",
      "@alert_type": "TRAFFIC_ACCIDENT",
      "@description": "Choque frontal",
      "map": {
        "@map_type": "STREETS",
        "@zoom": "1",
        "@radius": "250",
        "@coordinates": "42.4353792,-8.8252416"
      },
      "service": {
        "@service_type": "MOBILITY"
      },
      "object": [
        {
          "@name": "BMW",
          "@description": "Serie 3",
          "@danger_level": "MODERATE",
          "@object_type": "CAR",
          "data_object": {
            "@url": "https://db.com/image"
          }
        },
        {
          "@name": "Mercedes",
          "@description": "Benz",
          "@danger_level": "HIGH",
          "@object_type": "TRUCK",
          "data_object": {
            "@url": "https://db.com/video"
          }
        }
      ]
    }
  }
}
```


Figura 13: Representación en formato JSON-LD de la incidencia

## 10.4 CONSULTA DE INCIDENCIAS

Una vez finalizado el proceso de reporte de las incidencias por parte de los usuarios, éstas se publican para que puedan ser consultadas por el resto de los usuarios de la plataforma. La Figura 14: Consulta de incidencia en CrowDSL muestra una representación de la incidencia del ejemplo del accidente.

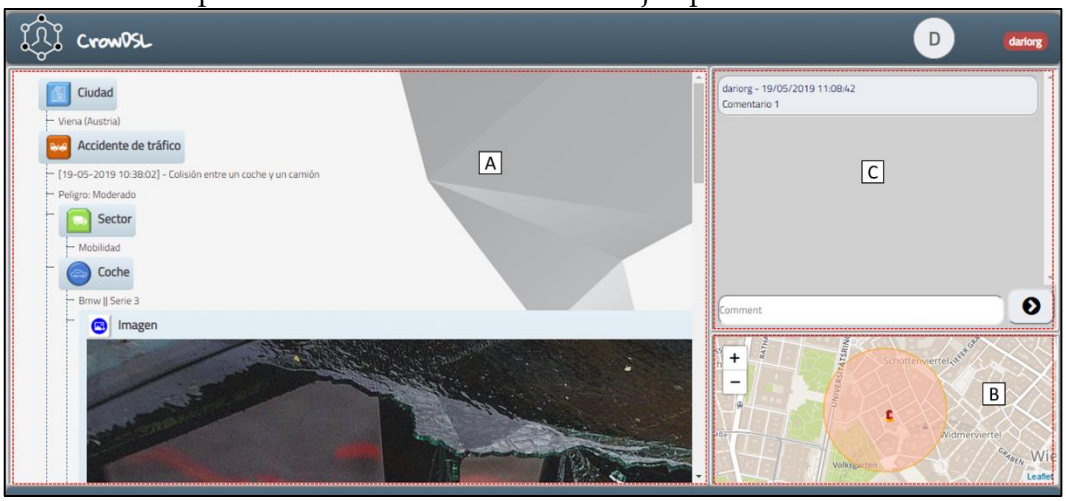



Figura 14: Consulta de incidencia en CrowDSL

La vista que se emplea para mostrar la incidencia se compone nuevamente de diversos apartados que permiten ofrecer al usuario toda la información que se

ha recopilado de la incidencia. La interfaz se estructura en los siguientes componentes:

- **A - Datos de la incidencia**: Partiendo de la representación de la incidencia en formato JSON-LD se genera una representación en formato HTML con los datos de la misma, tanto los datos introducidos por el usuario en modo texto como los elementos multimedia adjuntos. Con todo ello se genera una representación en forma de árbol que permite conocer la forma en la que se relacionan los elementos del modelo.
- **B - Mapa de ubicación**: Mediante este elemento se pude obtener la ubicación exacta de la incidencia. Este componente es parametrizado por el usuario desde el DSL. Se permite escoger el tipo de mapa a visualizar (calles, satélite, …), el radio de peligrosidad (círculo de la imagen) y el zoom del mapa. En el centro se sitúa un marcador que indica la ubicación exacta del origen de la alerta.
- **C - Comentarios**: Apartado para que los usuarios dejen los comentarios que estimen oportunos acerca de la incidencia.

## 10.5 SOFTWARE Y HARDWARE UTILIZADO

El desarrollo del prototipo del proyecto, así como las pruebas se han realizado sobre un sistema con las siguientes características:

- Software:
  - Lenguaje de programación principal: **Python 3.6.3** [40]
  - Lenguaje de programación del editor**: JavaScript** [41]
  - Framework: **Flask 1.0.2** [42]**,** Motor de plantillas: **Jinja2 2.4** [43]
  - Base de datos: **MongoDB**
- Hardware:
  - Sistema operativo: **Microsoft Windows 10 Home**
  - Memoria RAM: **8 GB**
  - Tipo de sistema: **PC basado en x64**
  - Procesador: **Intel(R) Core(TM) i7-6498DU CPU @ 2.50GHz, 2601 MHz**
  - Las pruebas de usabilidad del proyecto se han realizado empleando tres navegadores distintos: **Google Chrome**, **Mozilla Firefox** y **Microsoft Edge**.

# 11 Evaluación y discusión

Concluidas las labores de diseño y la posterior implementación del prototipo que pretende validar la hipótesis de la investigación, se continuó por plantear la evaluación del proyecto. Dicha evaluación se estructuró en dos bloques independientes, por una parte una prueba realizada con usuarios reales operando con el editor web desarrollado y por otra una encuesta con el objetivo de medir el grado de satisfacción del usuario con la plataforma. A continuación se expone la forma en la que se llevaron a cabo sendas evaluaciones.

## 11.1 EVALUACIÓN DEL EDITOR WEB POR PARTE DE USUARIOS REAL

Para la realización de esta prueba se seleccionó un conjunto de 20 usuarios con edades comprendidas entre los 20 y los 40 años, con formación en el campo de la informática, si bien no disponen de conocimientos relacionados con el Internet de las Cosas y de manera más directa al ámbito de este proyecto tampoco hay utilizado ninguno de ellos plataformas ciudadanas para la gestión de incidencias. A cada uno de los usuarios se les expuso detalladamente la forma de operar tanto con el lenguaje como con el editor. Para que comprendiesen tanto el manejo del lenguaje como del editor se les explicó la forma de realizar un conjunto de operaciones básicas, tales como: agregar y enlazar elementos, especificar propiedades para los atributos y realizar una validación de los datos. A fin de asegurar que el grado de dificultad fuese el mismo para todos los individuos se planteó un caso práctico que debían seguir todos los usuarios durante el proceso de modelado de la incidencia. A continuación se expone dicho caso práctico.

> "El usuario presencia el suceso de un accidente de tráfico en el que intervienen dos vehículos, un coche y un camión. La colisión tiene lugar en la ciudad de Viena (Austria). La gravedad del accidente puede ser considerada como alta. El coche sufre daños graves y el camión daños moderados. El radio de incidencia del accidente es de 50 metros"
>
> ** Se proporciona al usuario una imagen y un video que se asocian al coche y al camión implicados respectivamente. Este material se asume como tomado por parte del usuario en el momento del suceso del evento a reportar.*

Una vez planteado el caso práctico se instó a los usuarios al inicio de la prueba. Para la primera parte, manejo del editor, se empleó el software Mousotron

[44] a fin de obtener la medición de los tiempos de la prueba y el número de clics de ratón (pos. izquierda). De igual modo, durante la realización de la prueba, se anotaron los errores cometidos por los propios usuarios a fin de establecer una relación con aquellos más comunes. Para la realización de la prueba se estudió el siguiente conjunto de errores:

| | **Error** |
|---|---|
| **A** | Faltan elementos generales requeridos para la validación: usuario, ciudad, mapa, alerta |
| **B** | Error en el tipo de los datos de un elemento |
| **C** | Faltan campos requeridos por rellenar |
| **D** | Error en la relación de los elementos del lenguaje |
| **E** | No se comprende el funcionamiento del editor para la selección de componentes |
| **F** | No se interpretan correctamente los mensajes de error del sistema de log |
| **G** | No se comprende la funcionalidad de algunos elementos del lenguaje |
| **H** | No se comprende la diferencia entre la validación y el envío de la incidencia. |

Tabla 2: Errores de los usuarios analizados

En cuanto a la medición de tiempos, pulsaciones de teclado y clics de ratón los datos obtenidos por usuario son los siguientes:

| | **Tiempo empleado (s)** | **Pulsaciones de ratón** | **Pulsaciones de teclado** |
|---|---|---|---|
| **U1** | 140 | 64 | 87 |
| **U2** | 162 | 71 | 82 |
| **U3** | 175 | 84 | 113 |
| **U4** | 113 | 61 | 156 |
| **U5** | 159 | 66 | 141 |
| **U6** | 210 | 93 | 62 |
| **U7** | 172 | 67 | 178 |
| **U8** | 105 | 59 | 125 |
| **U9** | 112 | 61 | 148 |
| **U10** | 103 | 65 | 93 |
| **U11** | 148 | 76 | 144 |
| **U12** | 146 | 85 | 106 |
| **U13** | 124 | 67 | 85 |
| **U14** | 136 | 94 | 122 |
| **U15** | 203 | 102 | 201 |
| **U16** | 198 | 88 | 212 |
| **U17** | 123 | 64 | 98 |
| **U18** | 137 | 59 | 89 |
| **U19** | 145 | 81 | 71 |
| **U20** | 169 | 87 | 134 |

Tabla 3: Duración de la prueba, clics de ratón y pulsaciones de teclado por usuario

Una vez obtenidos los resultados se continuó por aplicar diferentes métricas estadísticas a fin de obtener una mejor interpretación de los resultados. Dicha información se especifica en la tabla inferior.

| | Tiempo empleado (s) | Pulsaciones de ratón | Pulsaciones de teclado |
|---|---|---|---|
| **Media** | 149 | 74,7 | 122,35 |
| **Máximo** | 210 | 102 | 212 |
| **Mínimo** | 103 | 59 | 62 |
| **Mediana** | 145,5 | 69 | 117,5 |

Tabla 4: Métricas estadísticas para tiempos, clics y pulsaciones

Los datos arrojan un tiempo medio para la realización de la prueba de 149 segundos, equivalente a 2,48 minutos (la mediana resulta de 145,5 segundos, 2,43 minutos). El usuario que más rápido ha realizado la prueba lo ha hecho en 103 segundos (1,71 minutos) y por el contrario, el que más tiempo ha empleado ha utilizado 210 segundos (3,5 minutos).

Con respecto a los clics de ratón el valor medio se sitúa en 74,7 clics (mediana de 69 clics), un valor máximo de 202 y un mínimo de 59.

Para las pulsaciones de teclado la media muestra un valor de 122,5 pulsaciones (mediana de 117,5), un máximo de 212 pulsaciones y un mínimo de 62 pulsaciones. Esta variable se puede considerar la menos representativa de todo el conjunto puesto que cada usuario tenía la libertad de especificar la descripción que dispusiese para cada elemento, resultando textos de longitud variable.

Analizamos ahora los errores cometidos por los usuarios. La tabla inferior muestra el número de errores total cometidos por cada usuario en cada uno de los errores que se estudiaron.

| Error | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| **U1** | 2 | 4 | 2 | 2 | 1 | 2 | 1 | 0 |
| **U2** | 3 | 6 | 1 | 3 | 0 | 1 | 3 | 0 |
| **U3** | 1 | 4 | 3 | 5 | 0 | 2 | 2 | 0 |
| **U4** | 1 | 2 | 3 | 5 | 0 | 2 | 1 | 0 |
| **U5** | 2 | 6 | 3 | 6 | 0 | 1 | 0 | 0 |
| **U6** | 3 | 7 | 3 | 2 | 0 | 3 | 0 | 0 |
| **U7** | 1 | 1 | 4 | 7 | 1 | 2 | 0 | 1 |
| **U8** | 1 | 4 | 6 | 7 | 1 | 2 | 0 | 0 |
| **U9** | 1 | 5 | 8 | 6 | 0 | 1 | 0 | 0 |
| **U10** | 1 | 5 | 2 | 6 | 1 | 3 | 1 | 0 |
| **U11** | 1 | 4 | 3 | 5 | 0 | 4 | 0 | 0 |
| **U12** | 2 | 1 | 5 | 2 | 2 | 1 | 1 | 1 |
| **U13** | 3 | 6 | 6 | 3 | 0 | 2 | 2 | 0 |
| **U14** | 2 | 7 | 3 | 7 | 0 | 0 | 0 | 0 |
| **U15** | 2 | 7 | 2 | 8 | 1 | 0 | 0 | 0 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **U16** | 1 | 2 | 2 | 6 | 0 | 1 | 0 | 0 |
| **U17** | 1 | 1 | 3 | 6 | 0 | 2 | 1 | 0 |
| **U18** | 1 | 3 | 1 | 5 | 0 | 0 | 0 | 0 |
| **U19** | 1 | 6 | 3 | 2 | 0 | 1 | 0 | 0 |
| **U20** | 1 | 5 | 3 | 1 | 2 | 3 | 0 | 0 |

Tabla 5: Número de errores por usuario

Aplicamos nuevamente medidas estadísticas sobre los datos recogidos acerca de los errores cometidos por los usuarios durante la prueba con el editor web.

| | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** |
|---|---|---|---|---|---|---|---|---|
| **Mínimo** | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| **Cuartil 1** | 1 | 2,25 | 2 | 2,25 | 0 | 1 | 0 | 0 |
| **Mediana** | 1 | 4,5 | 3 | 5 | 0 | 2 | 0 | 0 |
| **Cuartil 3** | 2 | 6 | 3,75 | 6 | 1 | 2 | 1 | 0 |
| **Máximo** | 3 | 7 | 8 | 8 | 2 | 4 | 3 | 1 |
| **Rango** | 2 | 6 | 7 | 7 | 2 | 4 | 3 | 1 |
| **Rango IC** | 1 | 3,75 | 1,75 | 3,75 | 1 | 1 | 1 | 0 |
| **Moda** | 1 | 4 | 3 | 6 | 0 | 2 | 0 | 0 |

Tabla 6: Medidas estadísticas de errores durante la prueba

Representamos los datos que se recogen en la gráfica inferior. Para ellos se ha empleado un gráfico de cajas y bigotes en el que se establece la comparativa entre los diferentes errores que se han incluido en el estudio.

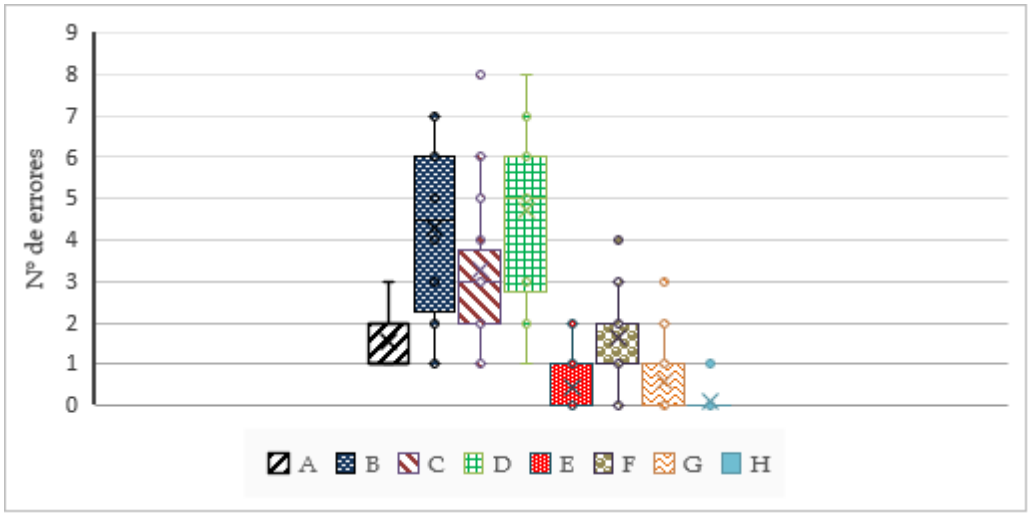



Figura 15: Número de errores por tipo

Los resultados muestran que los errores que aglutinan un mayor número de fallos son el C (faltan campos requeridos por rellenar) y el D (no se relacionan de manera correcta diferentes elementos del lenguaje), con un valor máximo de 8 errores. Por el contrario los errores que se dan en menos ocasiones con el E (error en la selección de componentes), F (interpretación de los errores del log), G (funcionalidad de algunos elementos) y H (diferencia entre validación y envío), con valores mínimos de 0. De manera global, el estudio muestra que los usuarios cometen, de manera principal, tres fallos: B (error en el tipado de datos de

atributos), C (elementos requeridos no especificados) y D (relaciones entre los diferentes componentes del lenguaje).

Procedemos ahora a analizar los resultados de la encuesta final.

## 11.2 ENCUESTA

Tras la realización de la prueba se pidió a cada uno de los usuarios que realizasen una encuesta compuesta por 10 declaraciones a fin de medir el grado de satisfacción de cada uno de ellos con respecto a la plataforma desarrollada. La citada encuesta se compone de las declaraciones que se exponen a continuación, pudiendo ser cada una de ellas valorada por el usuario en un rango de 1 a 5 de acuerdo a la escala Likert [45], siendo 1 totalmente en desacuerdo, 2 en desacuerdo, 3 neutral, 4 de acuerdo y 5 totalmente de acuerdo.

| D | Declaración |
|---|---|
| **1** | Los elementos del lenguaje están bien definidos y se comprende su finalidad. |
| **2** | Consideras que los elementos del lenguaje son suficientes para modelar cualquier tipo de incidencia. |
| **3** | Resulta sencillo especificar los valores para los atributos de los elementos (nivel de peligro, descripción, …) |
| **4** | Se comprende la estructura lógica de las relaciones entre entidades. |
| **5** | Prefieres esta metodología a la opción de hacer un envío de las incidencias en modo texto. |
| **6** | La forma en la que se estructuran los componentes del editor permite operar con él de manera sencilla. |
| **7** | Consideras que el editor web resulta útil para el fin que se ha diseñado. |
| **8** | La retroalimentación aportada por el log de la aplicación resulta comprensible por el usuario. |
| **9** | Consideras que la implantación de este proyecto en una Smart City aportaría ventajas a esta. |
| **10** | De manera global, el grado de satisfacción del usuario con la plataforma es bueno. |

**De manera adicional se formuló a cada usuario una última pregunta en la que se le instaba a que propusiese las mejoras que estimase oportunas de cara a una próxima versión de la plataforma.*

La Tabla 7: Respuesta de cada usuario a las preguntas muestra las respuestas de los usuarios a las preguntas de la encuesta:

| D | U1 | U2 | U3 | U4 | U5 | U6 | U7 | U8 | U9 | U10 | U11 | U12 | U13 | U14 | U15 | U16 | U17 | U18 | U19 | U20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 4 | 5 | 5 | 5 | 5 | 4 | 5 | 3 | 5 | 5 | 5 | 4 | 3 | 5 | 4 | 4 | 5 | 5 | 5 | 5 |
| **2** | 3 | 4 | 4 | 4 | 3 | 4 | 5 | 5 | 5 | 4 | 5 | 4 | 3 | 3 | 4 | 5 | 4 | 4 | 5 | 4 |
| **3** | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 5 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **4** | 3 | 4 | 3 | 3 | 4 | 3 | 3 | 3 | 5 | 3 | 3 | 4 | 4 | 3 | 3 | 4 | 5 | 4 | 4 | 4 |
| **5** | 5 | 5 | 5 | 5 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 4 | 5 | 5 | 5 | 5 |
| **6** | 5 | 5 | 5 | 5 | 5 | 3 | 4 | 5 | 5 | 5 | 5 | 5 | 4 | 4 | 4 | 4 | 5 | 5 | 5 | 5 |
| **7** | 5 | 4 | 5 | 5 | 5 | 4 | 4 | 4 | 5 | 5 | 5 | 5 | 4 | 4 | 4 | 3 | 4 | 4 | 5 | 5 |
| **8** | 2 | 3 | 1 | 2 | 2 | 1 | 0 | 2 | 3 | 1 | 0 | 3 | 4 | 5 | 1 | 2 | 2 | 2 | 3 | 1 |
| **9** | 5 | 5 | 5 | 5 | 4 | 4 | 5 | 5 | 4 | 4 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| **10** | 4 | 5 | 4 | 4 | 4 | 4 | 5 | 3 | 4 | 5 | 5 | 5 | 4 | 4 | 5 | 4 | 4 | 5 | 5 | 5 |

Tabla 7: Respuesta de cada usuario a las preguntas

Trasladamos nuevamente estos datos a una tabla en la que se recogen las variables estadísticas de valores mínimo y máximo, mediana, moda, cuartiles 1 y 3, rango y rango inter-cuartílico.

| | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|
| **Mínimo** | 3 | 3 | 4 | 3 | 4 | 3 | 3 | 0 | 4 | 3 |
| **Cuartil 1** | 4 | 4 | 5 | 3 | 5 | 4 | 4 | 1 | 4,25 | 4 |
| **Mediana** | 5 | 4 | 5 | 3,5 | 5 | 5 | 4,5 | 2 | 5 | 4 |
| **Cuartil 3** | 5 | 5 | 5 | 4 | 5 | 5 | 5 | 3 | 5 | 5 |
| **Máximo** | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| **Rango** | 2 | 2 | 1 | 2 | 1 | 2 | 2 | 5 | 1 | 2 |
| **Rango IC** | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 2 | 0,75 | 1 |
| **Moda** | 5 | 4 | 5 | 3 | 5 | 5 | 5 | 2 | 5 | 4 |

Tabla 8: Medidas estadísticas de la encuesta

Representamos esta información en un gráfico de cajas y bigotes a fin de obtener una mejor interpretación de los resultados obtenidos.

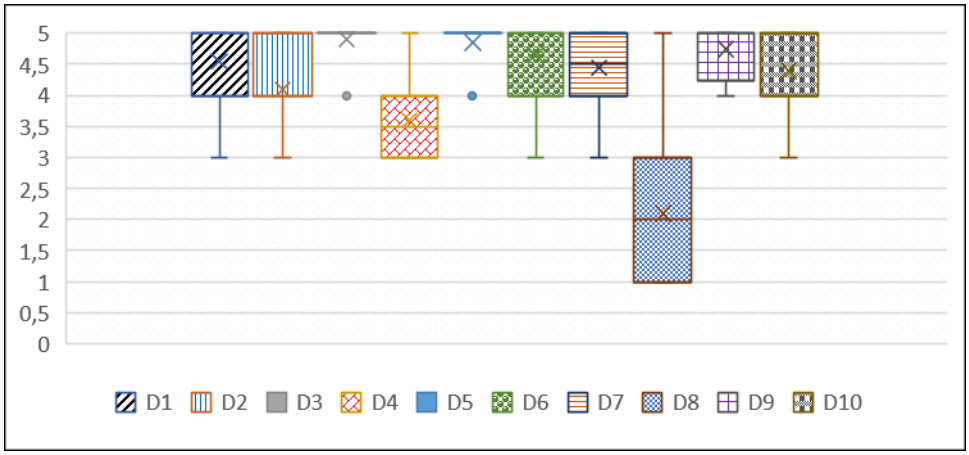



Figura 16: Valoraciones por declaración

Posteriormente se procedió a analizar el grado de satisfacción de los usuarios estudiando la valoración de cada una de las declaraciones. La Tabla 8: Medidas estadísticas de la encuesta, recoge el número de respuestas global por cada declaración y valoración.

| Pregunta | Totamente en desacuerdo | En desacuerdo | Neutral | De acuerdo | Totamente de acuerdo |
|---|---|---|---|---|---|
| **D1** | 0 | 0 | 2 | 5 | 13 |
| **D2** | 0 | 0 | 4 | 10 | 6 |
| **D3** | 0 | 0 | 0 | 2 | 18 |
| **D4** | 0 | 0 | 10 | 8 | 2 |
| **D5** | 0 | 0 | 0 | 3 | 17 |
| **D6** | 0 | 0 | 1 | 5 | 14 |
| **D7** | 0 | 0 | 1 | 9 | 10 |
| **D8** | 7 | 7 | 4 | 1 | 1 |
| **D9** | 0 | 0 | 0 | 5 | 15 |
| **D10** | 0 | 0 | 1 | 10 | 9 |

Tabla 9: Número de respuestas por pregunta

Representamos los datos en sendas gráficas de barras y barras apiladas.

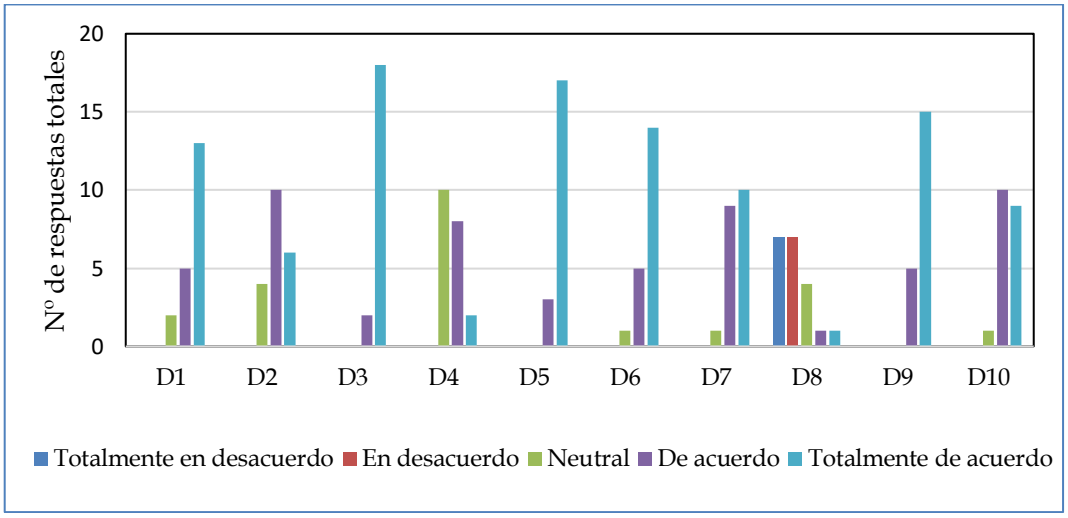



Figura 17: Distribución de respuestas por declaración

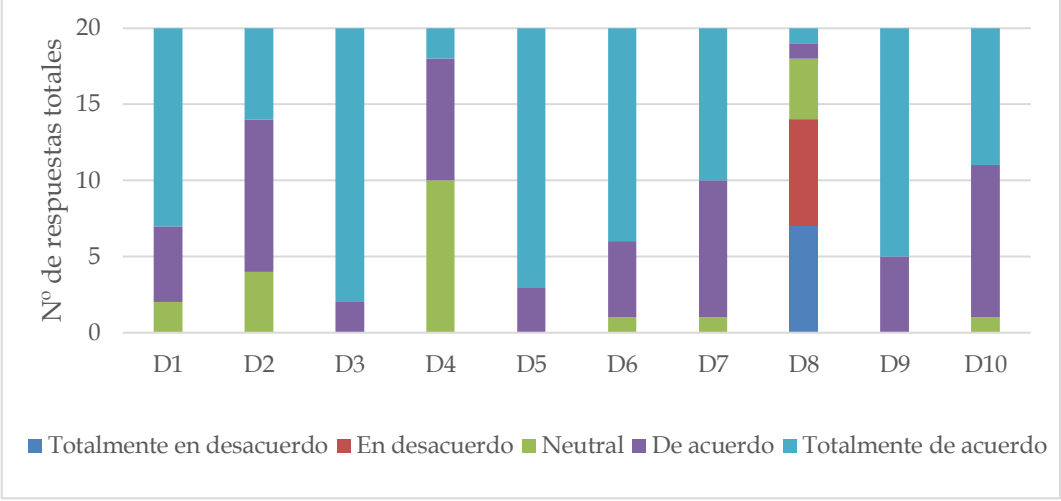



Figura 18: Distribución de respuestas por declaración (apiladas)

Discutimos ahora los resultados:

- A la declaración de si los usuarios consideran que los elementos del lenguaje se encuentran bien definidos, 15 de los 20 reponden que sí, representando el 75% de las respuestas.
- El 30% está totalmente de acuerdo y el 50 % está de acuerdo en que los elementos del lenguaje que se han incluido son suficientes para modelar cualquier incidencia.
- 18 de los 20 usuarios (90%) considera que resulta sencilla la forma de indicar los valores para los atributos.
- En cuanto a la estructura lógica de los componentes, una amplia mayoría (50% neutral y 40% de acuerdo) considera, por tanto, que en algunos casos no se comprende la forma de relacionar las entidades.
- Con respecto a la metodología, el 85% afirma preferir este método a lo que sería un envío en modo texto mediate un email o envío a otra plataforma.
- Respecto a la forma de estructurar los componentes en el editor, 14 de 20 personas (70%) manifiesta entenderlo de manera completa.
- El 95% de los usuarios está de acuerdo o completamente de acuerdo en que el editor resulta útil para el fin que se diseñó.
- La gran mayoría de los usuarios afirma no haber obtenido una buena retroalimentación de la aplicación para el tratamiento de errores., ya que el 70% de los enciestados manifiesta estar totalmente en desacuerdo o en desacuerdo con esta afirmación.
- Respecto a si consideran que la implantación en una Smart City aportaría ventajas a la misma, el 100% de los encuestados manifiesta que sí.
- Finalmente, de manera general, el grado de satisfacción del usuario con la plataforma se puede considerar bueno en el 95% de los casos (50% totalmente de acuerdo y 45% en acuerdo).

Como ya se comentó, de manera adicional a las 10 declaraciones de la encuesta se formuló a cada usuario una pregunta adicional en la que se les instaba a que propusiesen mejoras o ampliaciones a la plataforma con la que acaban de trabajar. Estos son los puntos que se obtuvieron:

- Mejora del sistema de log.
- Inclusión de nuevos componentes.
- Extender el DSL para dar soporte a consultas de los datos.

# 12 Conclusiones

La presente investigación tenía por objetivo, empleando Ingeniería Dirigida por Modelos por medio del diseño y construcción de un Lenguaje de Dominio Específico, lograr que los habitantes de diferentes *Smart Cities* en la que se pudiese implantar el proyecto dispusiesen de un método para realizar el reporte de incidencias de manera que toda información se encuentre etiquetada, enlazada y se facilite su procesamiento y consulta.

Una vez finalizado el desarrollo de la plataforma y su posterior evaluación se pudo comprobar que el grado de satisfacción de los usuarios con el DSL desarrollado es alto o muy alto, con un 95% de los individuos encuestados, resaltando la utilidad del lenguaje al margen de otras características y entendíendose por cumplido el objetivo principal del proyecto.

Partiendo del objetivo principal se enunciaron un conjunto de subobjetivos a fin de asegurar igualmente su consecución y que afectaban de manera directa a la contribución del proyecto.

El primer subobjetivo tenía que ver con el diseño del metamodelo inicial, para el cual los usuarios consideraron que los elementos incluidos en el mismo son suficientes para modelar cualquier tipo de incidencia, entendiéndose también la forma en que se estructuran los componentes del lenguaje.

De igual modo se marcó como subobjetivo la definición de la sintaxis concreta del lenguaje para asegurar que los usuarios asocien cada elemento a una representación gráfica del mismo. La gran mayoría de los usuarios afirman que los elementos del lenguaje se encuentran bien definidos y se entiende su funcionalidad.

El tercer y cuarto subobjetivo tenían que ver con el desarrollo y la posterior evaluación del prototipo de iba a dar soporte a los planteamientos de la investigación. Con respecto a los resultados del editor la encuesta arroja que los usuarios lo consideran útil para el fin para el que se diseñó, prefieren esta metodología a cualquier otra y entienden que la implantación en una ciudad aportaría múltiples ventajas.

Una vez concluidos el análisis de objetivos y subobjetivos nos planteamos las ventajas que aporta nuestra plataforma con respecto a las otras que han sido analizadas en la sección del Estado del arte. La principal diferencia entre esta

plataforma y el resto radica en el uso de Ingeniería dirigida por modelos para llevar a cabo el proceso de captura de datos por medio del Lenguaje de Dominio Específico desarrollado. Debido a que se basa en un metamodelo se aseguran diferentes propiedades como la reusabilidad, portabilidad y la interoperabilidad. Además, mediante este método, la información que se captura ya se encuentra etiquetada, por lo que se conoce al elemento al que describe. Esta circunstancia podría dar aplicabilidad a la solución en múltiples tecnologías, como la Web Semántica o técnicas de Big Data, debido al empleo de formatos abiertos (JSON) para el almacenamiento de los datos.

Una vez analizados los resultados de la investigación por medio de las conclusiones finales, nos preguntamos ahora que trabajo se plantea de cara a una próxima versión de la plataforma una vez obtenida una retroalimentación por parte de los usuarios.

# 13 Trabajo futuro

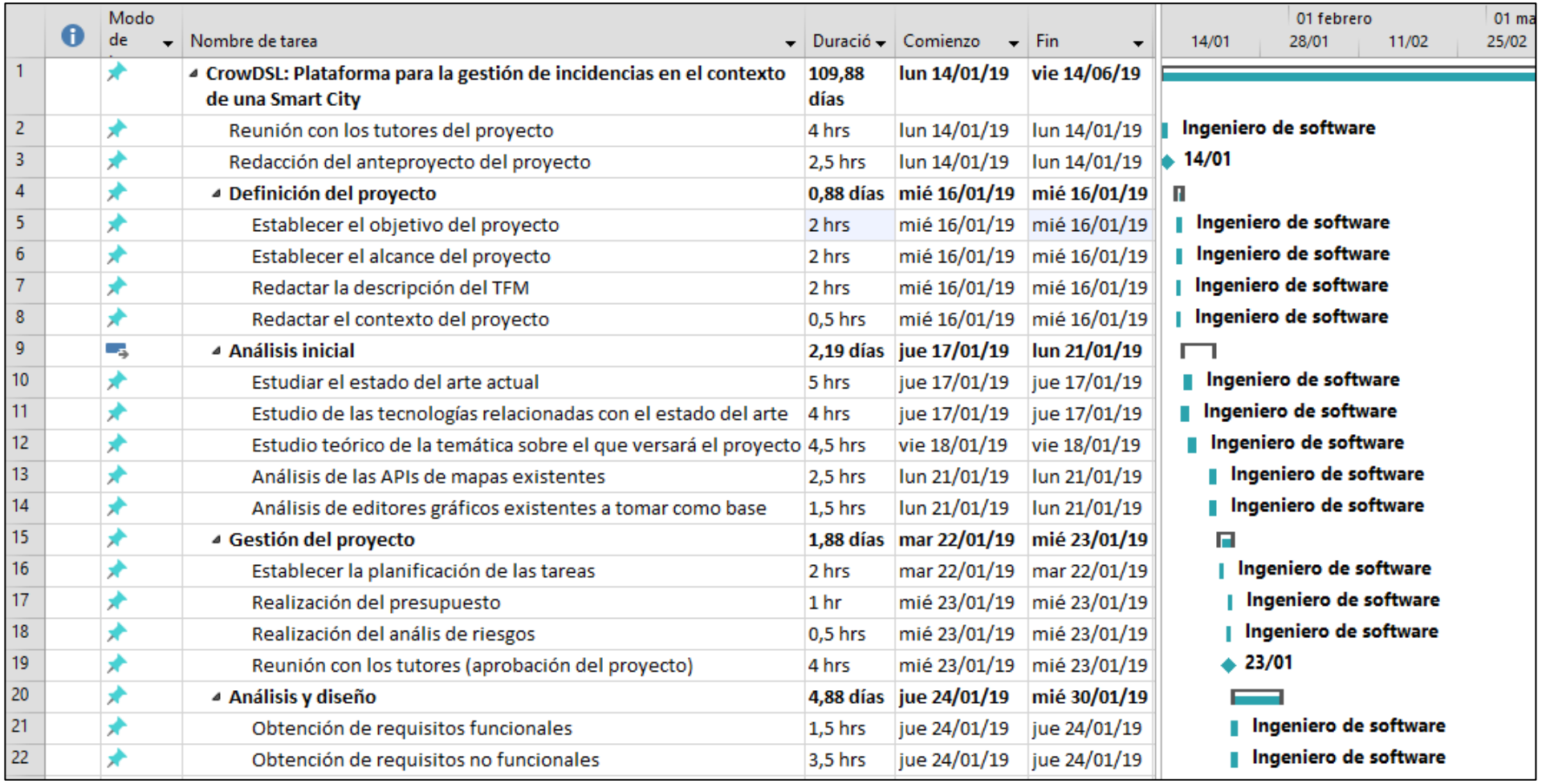
En el presente apartado se procede a detallar determinados aspectos del proyecto sobre los que se podrían plantear correcciones y/o ampliaciones. Una vez analizadas las opiniones de los usuarios hemos entendido que existen partes del mismo a mejorar. Alguna de estas correcciones/ampliaciones podrían ser:

- **Mejora del sistema de log**: Algunos usuarios sin demasiada experiencia en el mundo del desarrollo de software manifestaron que en ocasiones se les dificultaba ligeramente la labor de interpretación de los mensajes de log de la aplicación. Se podría estudiar, por tanto, que la aplicación mostrase mensajes más explícitos para el usuario.
- **Inclusión de nuevos elementos al lenguaje**: En determinados casos algunos usuarios manifestaron que resultaría interesante incluir nuevos elementos al lenguaje. Incluso un objeto «Otros» a fin de poder representar entidades que no se recogen en esta primera versión.
- **Extender el DSL para que también dé soporte a la parte de consulta de las incidencias**: Ahora mismo las incidencias se almacenan en formato JSON-LD. Una opción sería extender el lenguaje para que se permitiese las labores de filtrado y de como mostrar la información almacenada.
- **Mejora del editor Web y creación de *app***: El editor actual podría ser mejorado en múltiples aspectos, por ejemplo a la hora de adjuntar archivos, ya que por el momento esta labor se hace a través de una URL. Además para facilitar su uso de manera cotidiana a los ciudadanos de una Smart City se podría crear la versión *app*.
- **Integración de la plataforma con Smart Objects**: En una futura versión de la plataforma se podría estudiar la conectividad con diversos objetos inteligentes a fin de intercambiar información, obtener datos, enviar notificaciones, etc.
- **Técnicas de Big Data aplicadas al los datos almacenados**: A medida que se vayan generando grandes volúmenes de datos podría resultar interesante llevar a cabo un procesamiento de los datos a fin de obtener estadísticas, indicadores o cualquier otra información que pueda resultar relevante.

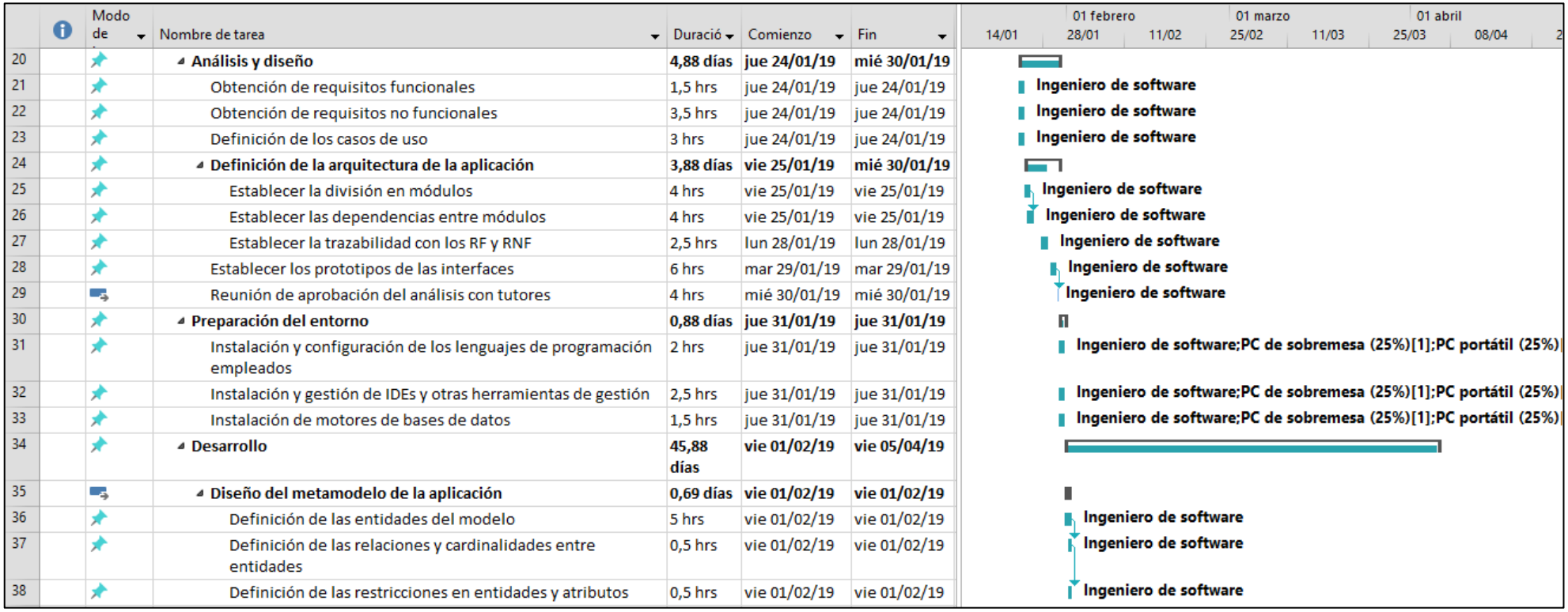
# 14 Planificación de tareas (final)

A continuación se exponen los periodos de tiempo y la duración final de todas y cada una de las tareas en las que se había desglosado el proyecto en la planificación inicial efectuada al inicio del mismo.

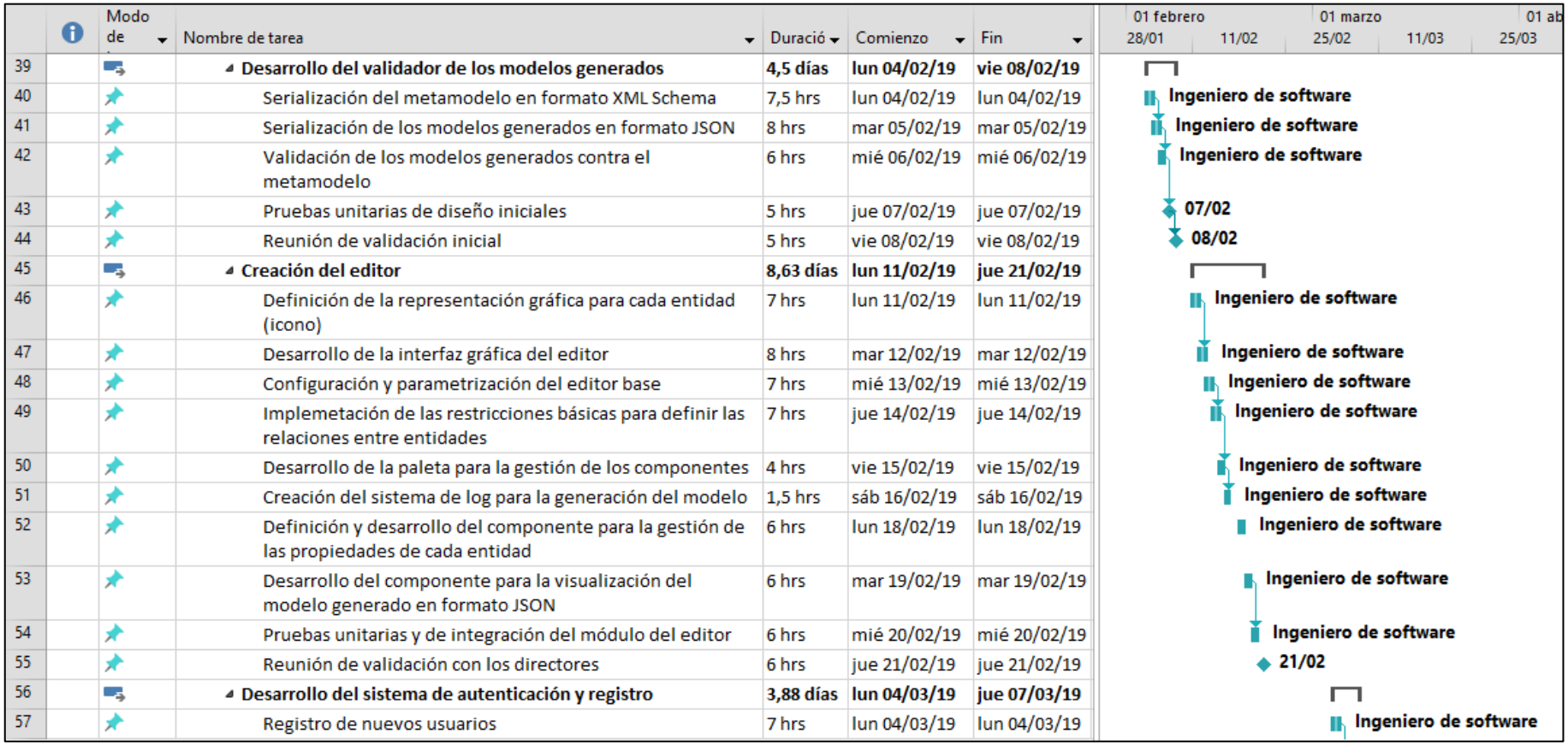
## 14.1 DESGLOSE DE TAREAS



| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 1 | | **CrowDSL: Plataforma para la gestión de incidencias en el contexto de una Smart City** | **109,88 días** | **lun 14/01/19** | **vie 14/06/19** |
| 2 | | Reunión con los tutores del proyecto | 4 hrs | lun 14/01/19 | lun 14/01/19 |
| 3 | | Redacción del anteproyecto del proyecto | 2,5 hrs | lun 14/01/19 | lun 14/01/19 |
| 4 | | **Definición del proyecto** | **0,88 días** | **mié 16/01/19** | **mié 16/01/19** |
| 5 | | Establecer el objetivo del proyecto | 2 hrs | mié 16/01/19 | mié 16/01/19 |
| 6 | | Establecer el alcance del proyecto | 2 hrs | mié 16/01/19 | mié 16/01/19 |
| 7 | | Redactar la descripción del TFM | 2 hrs | mié 16/01/19 | mié 16/01/19 |
| 8 | | Redactar el contexto del proyecto | 0,5 hrs | mié 16/01/19 | mié 16/01/19 |
| 9 | | **Análisis inicial** | **2,19 días** | **jue 17/01/19** | **lun 21/01/19** |
| 10 | | Estudiar el estado del arte actual | 5 hrs | jue 17/01/19 | jue 17/01/19 |
| 11 | | Estudio de las tecnologías relacionadas con el estado del arte | 4 hrs | jue 17/01/19 | jue 17/01/19 |
| 12 | | Estudio teórico de la temática sobre el que versará el proyecto | 4,5 hrs | vie 18/01/19 | vie 18/01/19 |
| 13 | | Análisis de las APIs de mapas existentes | 2,5 hrs | lun 21/01/19 | lun 21/01/19 |
| 14 | | Análisis de editores gráficos existentes a tomar como base | 1,5 hrs | lun 21/01/19 | lun 21/01/19 |
| 15 | | **Gestión del proyecto** | **1,88 días** | **mar 22/01/19** | **mié 23/01/19** |
| 16 | | Establecer la planificación de las tareas | 2 hrs | mar 22/01/19 | mar 22/01/19 |
| 17 | | Realización del presupuesto | 1 hr | mié 23/01/19 | mié 23/01/19 |
| 18 | | Realización del anális de riesgos | 0,5 hrs | mié 23/01/19 | mié 23/01/19 |
| 19 | | Reunión con los tutores (aprobación del proyecto) | 4 hrs | mié 23/01/19 | mié 23/01/19 |
| 20 | | **Análisis y diseño** | **4,88 días** | **jue 24/01/19** | **mié 30/01/19** |
| 21 | | Obtención de requisitos funcionales | 1,5 hrs | jue 24/01/19 | jue 24/01/19 |
| 22 | | Obtención de requisitos no funcionales | 3,5 hrs | jue 24/01/19 | jue 24/01/19 |

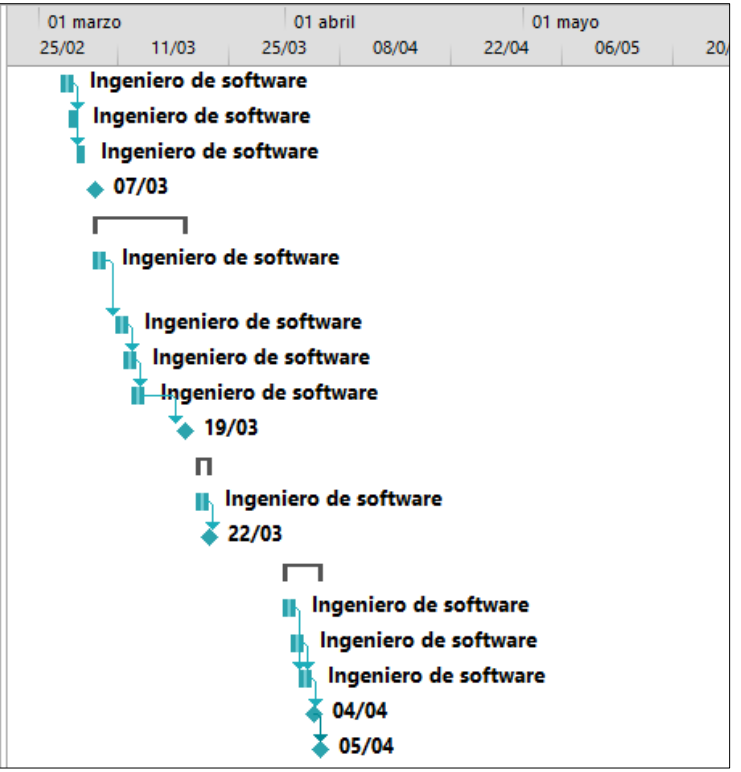


| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 20 | | **Análisis y diseño** | **4,88 días** | **jue 24/01/19** | **mié 30/01/19** |
| 21 | | Obtención de requisitos funcionales | 1,5 hrs | jue 24/01/19 | jue 24/01/19 |
| 22 | | Obtención de requisitos no funcionales | 3,5 hrs | jue 24/01/19 | jue 24/01/19 |
| 23 | | Definición de los casos de uso | 3 hrs | jue 24/01/19 | jue 24/01/19 |
| 24 | | **Definición de la arquitectura de la aplicación** | **3,88 días** | **vie 25/01/19** | **mié 30/01/19** |
| 25 | | Establecer la división en módulos | 4 hrs | vie 25/01/19 | vie 25/01/19 |
| 26 | | Establecer las dependencias entre módulos | 4 hrs | vie 25/01/19 | vie 25/01/19 |
| 27 | | Establecer la trazabilidad con los RF y RNF | 2,5 hrs | lun 28/01/19 | lun 28/01/19 |
| 28 | | Establecer los prototipos de las interfaces | 6 hrs | mar 29/01/19 | mar 29/01/19 |
| 29 | | Reunión de aprobación del análisis con tutores | 4 hrs | mié 30/01/19 | mié 30/01/19 |
| 30 | | **Preparación del entorno** | **0,88 días** | **jue 31/01/19** | **jue 31/01/19** |
| 31 | | Instalación y configuración de los lenguajes de programación empleados | 2 hrs | jue 31/01/19 | jue 31/01/19 |
| 32 | | Instalación y gestión de IDEs y otras herramientas de gestión | 2,5 hrs | jue 31/01/19 | jue 31/01/19 |
| 33 | | Instalación de motores de bases de datos | 1,5 hrs | jue 31/01/19 | jue 31/01/19 |
| 34 | | **Desarrollo** | **45,88 días** | **vie 01/02/19** | **vie 05/04/19** |
| 35 | | **Diseño del metamodelo de la aplicación** | **0,69 días** | **vie 01/02/19** | **vie 01/02/19** |
| 36 | | Definición de las entidades del modelo | 5 hrs | vie 01/02/19 | vie 01/02/19 |
| 37 | | Definición de las relaciones y cardinalidades entre entidades | 0,5 hrs | vie 01/02/19 | vie 01/02/19 |
| 38 | | Definición de las restricciones en entidades y atributos | 0,5 hrs | vie 01/02/19 | vie 01/02/19 |

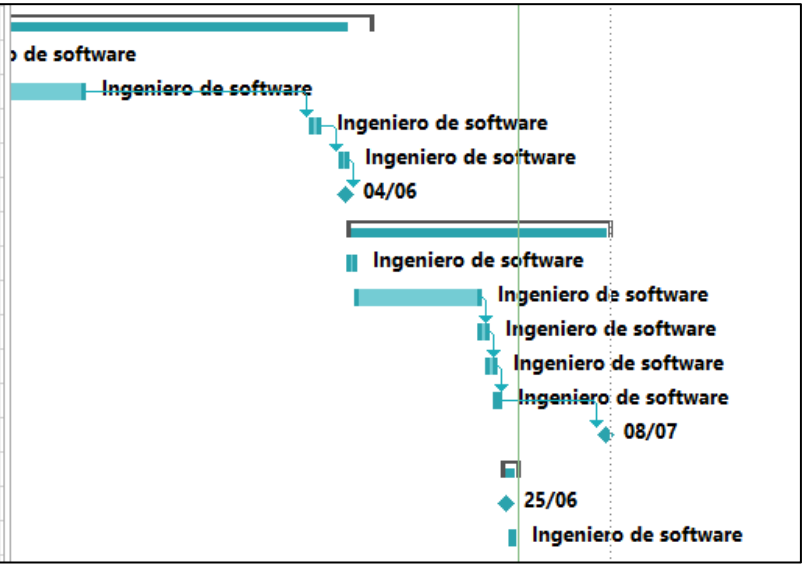


| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 39 | | **Desarrollo del validador de los modelos generados** | **4,5 días** | **lun 04/02/19** | **vie 08/02/19** |
| 40 | | Serialización del metamodelo en formato XML Schema | 7,5 hrs | lun 04/02/19 | lun 04/02/19 |
| 41 | | Serialización de los modelos generados en formato JSON | 8 hrs | mar 05/02/19 | mar 05/02/19 |
| 42 | | Validación de los modelos generados contra el metamodelo | 6 hrs | mié 06/02/19 | mié 06/02/19 |
| 43 | | Pruebas unitarias de diseño iniciales | 5 hrs | jue 07/02/19 | jue 07/02/19 |
| 44 | | Reunión de validación inicial | 5 hrs | vie 08/02/19 | vie 08/02/19 |
| 45 | | **Creación del editor** | **8,63 días** | **lun 11/02/19** | **jue 21/02/19** |
| 46 | | Definición de la representación gráfica para cada entidad (icono) | 7 hrs | lun 11/02/19 | lun 11/02/19 |
| 47 | | Desarrollo de la interfaz gráfica del editor | 8 hrs | mar 12/02/19 | mar 12/02/19 |
| 48 | | Configuración y parametrización del editor base | 7 hrs | mié 13/02/19 | mié 13/02/19 |
| 49 | | Implemetación de las restricciones básicas para definir las relaciones entre entidades | 7 hrs | jue 14/02/19 | jue 14/02/19 |
| 50 | | Desarrollo de la paleta para la gestión de los componentes | 4 hrs | vie 15/02/19 | vie 15/02/19 |
| 51 | | Creación del sistema de log para la generación del modelo | 1,5 hrs | sáb 16/02/19 | sáb 16/02/19 |
| 52 | | Definición y desarrollo del componente para la gestión de las propiedades de cada entidad | 6 hrs | lun 18/02/19 | lun 18/02/19 |
| 53 | | Desarrollo del componente para la visualización del modelo generado en formato JSON | 6 hrs | mar 19/02/19 | mar 19/02/19 |
| 54 | | Pruebas unitarias y de integración del módulo del editor | 6 hrs | mié 20/02/19 | mié 20/02/19 |
| 55 | | Reunión de validación con los directores | 6 hrs | jue 21/02/19 | jue 21/02/19 |
| 56 | | **Desarrollo del sistema de autenticación y registro** | **3,88 días** | **lun 04/03/19** | **jue 07/03/19** |
| 57 | | Registro de nuevos usuarios | 7 hrs | lun 04/03/19 | lun 04/03/19 |

| | Modo de | Nombre de tarea | Duració | Comienzo | Fin |
|---|---|---|---|---|---|
| 57 | | Registro de nuevos usuarios | 7 hrs | lun 04/03/19 | lun 04/03/19 |
| 58 | | Auntenticación para usuarios ya registrados | 5 hrs | mar 05/03/19 | mar 05/03/19 |
| 59 | | Actualización de los datos del perfil del usuario | 4,5 hrs | mié 06/03/19 | mié 06/03/19 |
| 60 | | Pruebas unitarias e integración del modo de autenticación | 8 hrs | jue 07/03/19 | jue 07/03/19 |
| 61 | | **Representación de las incidencias sobre mapas** | **7,38 días** | **vie 08/03/19** | **mar 19/03/19** |
| 62 | | Parametrización de la visualización del mapa de acuerdo a las especificaciones del usuario | 1 día | vie 08/03/19 | vie 08/03/19 |
| 63 | | Ubicación y etiquetado de cada incidencia por tipo | 8 hrs | lun 11/03/19 | lun 11/03/19 |
| 64 | | Visualización de los elementos asociados a la incidencia | 8 hrs | mar 12/03/19 | mar 12/03/19 |
| 65 | | Pruebas unitarias y de integración del módulo de mapas | 8 hrs | mié 13/03/19 | mié 13/03/19 |
| 66 | | Reunión de validación con los tutores | 4 hrs | mar 19/03/19 | mar 19/03/19 |
| 67 | | **Visualización y filtrado de incidencias** | **1,38 días** | **jue 21/03/19** | **vie 22/03/19** |
| 68 | | Filtrado de incidencias por los criterios disponibles | 8 hrs | jue 21/03/19 | jue 21/03/19 |
| 69 | | Pruebas unitarias y de integración del módulo de consulta | 4 hrs | vie 22/03/19 | vie 22/03/19 |
| 70 | | **Valoración y comentarios de las incidencias** | **4,06 días** | **lun 01/04/19** | **vie 05/04/19** |
| 71 | | Registro de comentarios relativos a incidencia | 8 hrs | lun 01/04/19 | lun 01/04/19 |
| 72 | | Registro de valoraciones de la incidencia | 8 hrs | mar 02/04/19 | mar 02/04/19 |
| 73 | | Visualización de comentarios y valoraciones | 7 hrs | mié 03/04/19 | mié 03/04/19 |
| 74 | | Pruebas unitarias y de integración del módulo de opiniones | 6 hrs | jue 04/04/19 | jue 04/04/19 |
| 75 | | Reunión de validación final de desarrollo | 1,5 hrs | vie 05/04/19 | vie 05/04/19 |



| | | | | | | |
|---|---|---|---|---|---|---|
| 76 | Artículo de investigación | 44,88 días | lun 08/04/19 | vie 07/06/19 | | |
| 77 | Pruebas de usabilidad y accesibilidad con usuarios reales | 6 hrs | lun 08/04/19 | lun 08/04/19 | | |
| 78 | Redacción del artículo | 17 días | lun 08/04/19 | mié 01/05/19 | | |
| 79 | Reunión de revisión de la escritura del artículo | 7 hrs | vie 31/05/19 | vie 31/05/19 | | |
| 80 | Estudio de la revista a la que realizar el envío | 6 hrs | mar 04/06/19 | mar 04/06/19 | | |
| 81 | Envío del artículo | 4 hrs | mar 04/06/19 | mar 04/06/19 | | |
| 82 | Integración e implantación | 23,88 días | mié 05/06/19 | lun 08/07/19 | | |
| 83 | Diseño y desarrollo de casos de uso de ejemplo | 6 hrs | mié 05/06/19 | mié 05/06/19 | | |
| 84 | Redacción de la memoria final | 12 días | jue 06/06/19 | vie 21/06/19 | | |
| 85 | Revisión final de la memoria | 7 hrs | sáb 22/06/19 | sáb 22/06/19 | | |
| 86 | Redacción de la presentación del proyecto ante tribunal | 7 hrs | dom 23/06/19 | dom 23/06/19 | | |
| 87 | Ensayo de la exposicion oral | 6 hrs | lun 24/06/19 | lun 24/06/19 | | |
| 88 | Presentación del proyecto ante tribunal | 4 hrs | lun 08/07/19 | lun 08/07/19 | | |
| 89 | Cierre del proyecto | 1,88 días | mar 25/06/19 | mié 26/06/19 | | |
| 90 | Reunión final de evaluación | 3,5 hrs | mar 25/06/19 | mar 25/06/19 | | |
| 91 | Estudio del estado de publicación del artículo de investigación | 3 hrs | mié 26/06/19 | mié 26/06/19 | | |




## 14.2 RECURSOS

| | Nombre del recurso | Tipo | Etiqueta de material | Iniciales | Grupo | Capacidad máxima | Tasa estándar | Tasa horas extra | Costo/Uso | Acumular | Calendario base | Cód |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Ingeniero de software | Trabajo | | IS | SOFT | 150% | 45,00 €/hr | 0,00 €/hr | 0,00 € | Prorrateo | Calendario TFM | IS |
| 2 | PC de sobremesa (25%) | Material | | PCSOBR | | | 337,50 € | | 0,00 € | Prorrateo | | |
| 3 | PC portátil (25%) | Material | | PORT | | | 287,50 € | | 0,00 € | Prorrateo | | |
| 4 | Servidor para despliegue y pruebas | Material | | ANDR | | | 235,00 € | | 0,00 € | Prorrateo | | |

# 15 Presupuesto (final)

En relación a los costes asumidos durante el desarrollo del proyecto, se muestran el presupuesto interno de empresa y el que se presentaría al cliente obtenidos a partir de las horas empleadas reales.

*NOTA*: Todos los conceptos se muestran sin aplicar el impuesto del IVA

## 15.1 PRESUPUESTO INTERNO DE EMPRESA

Para el presupuesto interno comenzamos por calcular el coste por unidad de obra para cada uno de los módulos del proyecto.

### Costes por unidad de obra

| Módulo | Medición (h) | Coste unit./h (€) | Coste total (€) |
|---|---|---|---|
| Definición del proyecto | 13 | 45 € | 585,00 € |
| Análisis inicial | 17,5 | 45 € | 787,50 € |
| Gestión del proyecto | 7,5 | 45 € | 337,50 € |
| Análisis y diseño | 28,5 | 45 € | 1282,50 € |
| Preparación del entorno | 6 | 45 € | 270,00 € |
| Diseño del metamodelo de la aplicación | 6 | 45 € | 270,00 € |
| Desarrollo del validador de los modelos generados | 31,5 | 45 € | 1417,50 € |
| Creación del editor | 58,5 | 45 € | 2632,50 € |
| Desarrollo del sistema de autenticación y registro | 24,5 | 45 € | 1102,50 € |
| Representación de las incidencias sobre mapas | 36 | 45 € | 1620,00 € |
| Visualización y filtrado de incidencias | 12 | 45 € | 540,00 € |
| Valoración y comentarios de las incidencias | 30,5 | 45 € | 1372,50 € |
| Artículo de investigación | 179 | 45 € | 8055,00 € |
| Redacción de la memoria | 132 | 45 € | 5940,00 € |
| Integración y preparativos finales | 40 | 45 € | 1800,00 € |
| **TOTAL** | | **622,5 h** | **28012,50€** |

El coste unitario por hora se ha calculado teniendo en cuenta el perfil del desarrollador del proyecto (Ingeniero de software) y atendiendo a los costes que deberá asumir la empresa en cuanto al pago de impuestos, salarios y otros conceptos relativos al trabajador.

Calculamos ahora los costes de licencias de software, costes de hardware y costes indirectos.
Para los diferentes elementos se establece un tiempo de amortización de 5 años. En cuanto a Sistemas Operativos y paquetes de ofimática se aplica un 10% y un 25% respectivamente dado que las licencias se entienden adquiridas para más proyectos y por tanto se aplica la parte proporcional.

**Coste de software**

| Concepto | Unidades | Coste unitario (€) | Coste total (€) |
|---|---|---|---|
| Microsoft Windows 10 | 1 (10%) | 127 | 12,70 € |
| Microsoft Office 365 | 1 (25%) | 110 | 27,50 € |
| PyCharm Community Edition (Jetbrains IDE) | 1 | 0 | 0,00 € |
| Python 3.6 | 1 | 0 | 0,00 € |
| Flask framework | 1 | 0 | 0,00 € |
| BDD Mongo DB | 1 | 0 | 0,00 € |
| **TOTAL** | | | **40,20 €** |

**Costes de hardware**

| Concepto | Unidades | Coste unitario (€) | Coste total (€) |
|---|---|---|---|
| Ordenador de sobremesa | 1 (25%) | 1350 € | 337,50 € |
| Ordenador portátil | 1 (25%) | 1150 € | 287,50 € |
| Servidor para despliegue y pruebas | 1 | 235 € | 235,00 € |
| **TOTAL** | | | **859,00 €** |

**Costes indirectos**

| Concepto | Unidades | Coste unitario (€) | Coste total (€) |
|---|---|---|---|
| Electricidad | 6 | 45 € | 270,00 € |
| Agua | 6 | 7 | 42,00 € |
| Internet y comunicaciones | 6 | 29 | 174,00 € |
| Desplazamiento | 6 | 37 | 222,00 € |
| Oficina | 6 | 4,25 | 25,50 € |
| Seguros de hardware | 6 | 6 | 36,00 € |
| **TOTAL** | | | **769,50 €** |

| | |
|---|---|
| Subtotal | 29681,20 € |

| | |
|---|---|
| Beneficio (15%) | 4452,18 € |
| IVA (21%) | 7168,01 € |
| **TOTAL** | **41301,39 €** |

## 15.2 PRESUPUESTO DE CLIENTE

| **Concepto** | **Coste total (€)** |
|---|---|
| Diseño del metamodelo de la aplicación | 1176,16 € |
| Desarrollo del validador de los modelos generados | 2368,66 € |
| Creación del editor | 3336,16 € |
| Desarrollo del sistema de autenticación y registro | 1986,16 € |
| Representación de las incidencias sobre mapas | 2008,66 € |
| Visualización y filtrado de incidencias | 1423,66 € |
| Valoración y comentarios de las incidencias | 2076,16 € |
| Artículo de investigación | 9383,86 € |
| Redacción de la memoria | 7168,74 € |
| Integración y preparativos finales | 2346,16 € |
| ***Subtotal*** | 33274,38 € |
| Hardware | 859,00 € |
| ***Subtotal*** | 34133,38 € |
| ***IVA (21%)*** | 7168,01 € |
| **TOTAL** | **41301,39 €** |

Una vez concluido el proyecto y habiendo estudiado la comparativa entre las planificaciones y los presupuestos iniciales y finales se han observado diferencias sustanciales en cuanto a tiempos empleados y a costes asumidos en determinadas tareas del proyecto. La planificación inicial constaba de un total de 565,5 horas estimadas para el desarrollo del proyecto, sin embargo, para la consecución del proyecto han resultado necesarias 622,5 horas. Este exceso de horas se ha debido fundamentalmente a dos factores, la redacción de la memoria final del proyecto y la redacción y revisión del artículo de investigación requerido para la presentación del presente TFM. Esta desviación temporal ha supuesto prolongar el proyecto 13 días más de lo esperado a fin de poder completar todas las tareas de manera satisfactoria y generando unos sobrecostes de 4665€ con respecto a lo presupuestado inicialmente.

# Referencias


[1] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Ad Hoc Networks Internet of things : Vision , applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things : A survey," *Comput. Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] S. Haller, "The Things in the Internet of Things," no. January 2010, 2016.

[4] O. Bello, S. Zeadally, and S. Member, "Intelligent Device-to-Device Communication in the Internet of Things," pp. 1–11, 2020.

[5] M. Chan, E. Campo, D. Estève, and J. Fourniols, "Maturitas Smart homes — Current features and future perspectives," vol. 64, pp. 90–97, 2009.

[6] R. Giffinger, C. Fertner, H. Kramar, and E. Meijers, "City-ranking of European Medium-Sized Cities," pp. 1–12.

[7] N. R. D. Council, "What are smarter cities?" [Online]. Available: https://smartcitiescouncil.com/members/natural-resources-defense-council. [Accessed: 29-Jan-2019].

[8] Patrice Rios, "Creating 'the smart city,'" 2008.

[9] United Nations, "Population Distribution, Urbanization, Internal Migration and Development: An International Perspective," 2011.

[10] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported," no. September 2013, pp. 81–93, 2014.

[11] X. Sheng, J. Tang, X. Xiao, and G. Xue, "Sensing as a Service : Challenges , Solutions and Future Directions," vol. 13, no. 10, pp. 3733–3741, 2013.

[12] K. C. Ocri, "Guidelines Report," no. February, 2003.

[13] S. Graham and S. Marvin, *Splintering urbanism: networked infrastructures, technological mobilities and the urban condition*. Routledge, 2002.

[14] R. M. S. Bezerra, F. M. S. Nascimento, and J. S. B. Martins, "On computational infraestruture requirements to smart and autonomic cities framework," in *2015 IEEE First International Smart Cities Conference (ISC2)*, 2015, pp. 1–6.

[15] N. Council, "Six technologies with potential impacts on us interests out to 2025," *Disruptive Civ. Technol. 2008*, 2008.

[16] M. Walport, "The Internet of Things: making the most of the Second Digital Revolution," *UK Gov. Chief Sci. Advis.*, 2014.

[17] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future internet: the internet of things architecture, possible applications and key challenges," in *2012 10th international conference on frontiers of information technology*, 2012, pp. 257–260.

[18] Y. Huang and G. Li, "Descriptive models for Internet of Things," in *2010 International Conference on Intelligent Control and Information Processing*, 2010, pp. 483–486.

[19] D. Washburn, U. Sindhu, S. Balaouras, R. A. Dines, N. M. Hayes, and L. E. Nelson, "Helping CIOs Understand ' Smart City ' Initiatives," 2010.

[20] H. Partridge, L. Queensland, and T. Brisbane, "Developing a Human Perspective to the Digital Divide in the Smart City THE DIGITAL DIVIDE : A REVIEW OF THE," 1995.

[21] C. González García, "Departamento de Informática Tesis Doctoral Programa de Doctorado de Ingeniería Informática « MIDGAR : interoperabilidad de objetos en el marco de Internet de las Cosas mediante el uso de Ingeniería Dirigida por Modelos »," no. March, 2017.

[22] "Vienna University of Technology," "European Smart Cities," 2015. [Online]. Available: http://www.smart-cities.eu. [Accessed: 26-Mar-2018].

[23] "IBM Institute for Business Value," "Smart City in China," *R. IBM Off. website, pp.3-45*, 2008.

[24] B. Selic, "From Model-Driven Development to Model-Driven Engineering," 2006.

[25] J. Manuel Cueva Lovelle, "Application of model-driven engineering ( MDA ) for the construction of a tool for domain-specific modeling ( DSM ) and the creation of modules in learning.

[26] Eclipse, "Ecore." [Online]. Available: https://wiki.eclipse.org/Ecore.

[27] OMG, "Object Management Group." [Online]. Available: https://www.omg.org/index.htm. [Accessed: 04-May-2019].

[28] V. García-Díaz, "MDCI : Model-Driven Continuous Integration," 2011.

[29] A. Van Deursen, P. Klint, and J. Visser, "Domain-Specific Languages : An Annotated Bibliography *," vol. 35, no. June, pp. 26–36, 2000.

[30] C. Enrique, "Desarrollo de un lenguaje de dominio específico para sistemas de gestión de aprendizaje y su herramienta de implementación ' KiwiDSM '

mediante ingeniería dirigida por modelos through model-driven engineering Resumen," vol. 15, no. 2, pp. 67–81, 2010.

[31] "World Wide Web Consortium (W3C)." [Online]. Available: https://www.w3c.es/%0A. [Accessed: 25-May-2019].

[32] W3C, "XML Schema (W3C)," 2012. [Online]. Available: https://www.w3schools.com/xml/schema_intro.asp. [Accessed: 05-May-2019].

[33] OMG, "About the XML metadata interchange specification version 2.5.1," 2015. [Online]. Available: https://www.omg.org/spec/XMI/2.5.1/. [Accessed: 04-May-2019].

[34] S. Mirri, C. Prandi, P. Salomoni, F. Callegati, and A. Campi, "On combining crowdsourcing, sensing and open data for an accessible smart city," *Proc. - 2014 8th Int. Conf. Next Gener. Mob. Appl. Serv. Technol. NGMAST 2014*, pp. 294–299, 2014.

[35] E. Aubry, T. Silverston, A. Lahmadi, and O. Festor, "CrowdOut: A mobile crowdsourcing service for road safety in digital cities," *2014 IEEE Int. Conf. Pervasive Comput. Commun. Work. PERCOM Work. 2014*, pp. 86–91, 2014.

[36] S. F. King and P. Brown, "Fix My Street or Else: Using the Internet to Voice Local Public Service Concerns," *Comput. Soc.*, pp. 72–80, 2007.

[37] mySociety, "mySociety." .

[38] B. Baykurt, "Redefining Citizenship and Civic Engagement : political values embodied in FixMyStreet . com," *An. do AoIR – Assoc. Internet Res. Seattle*, pp. 1–18, 2011.

[39] K. Benouaret, R. Valliyur-Ramalingam, F. Charoy, and R. V.-R. F. C. Karim Benouaret, "CrowdSC: Building Smart Cities with Large Scale Citizen Participation," *IEEE Internet Comput.*, vol. 17, p. 1, 2013.

[40] Python Software Foundation, "Python," 2019. [Online]. Available: https://www.python.org/.

[41] "JavaScript." [Online]. Available: https://www.javascript.com/.

[42] "Flask." [Online]. Available: http://flask.pocoo.org/.

[43] "Jinja 2." [Online]. Available: http://jinja.pocoo.org/.

[44] Blacksun software, "Mousotron." 2019.

[45] R. Likert, "A technique for the measurement of attitudes.," *Arch. Psychol.*, 1932.

[46] IEEE, "IEEE Internet of Things." [Online]. Available: https://iot.ieee.org/.

# Anexo I: Acrónimos

**DSL -** Domain-Specific Language

**HTML -** HyperText Markup Language

**IoT -** Internet of Things

**JCR -** Journal Citation Report

**JSON -** JavaScript Object Notation

**MDE** Model-Driven Engineering

**MOF -** Meta-Object Facility

**NoSQL -** No Structured Query Language

**OMG -** Object Manager Group

**RFID -** Radio Frequency IDentification

**XMI** - XML Metadata Interchange

**XML** - eXtensible Markup Language

# Anexo II: Glosario

**Flask:** Framework ligero del Python que permite la generación rápida de aplicaciones. Tiene una licencia BSD.

**Framework:** Agrupación de prácticas y criterios destinados a indicar la forma de enfocar un problema en particular.

**HTML(*HyperText Markup Language*)**: Lenguaje de marcado utilizado en el marcado de páginas Web. Permite definir la estructura de un documento de Internet.

**Internet of Things:** Concepto empleado para designar la interconexión de objetos heterogéneos de la vida cotidiana a través de Internet.

**Javascript**: Lenguaje de programación interpretado categorizado como un dialecto de ECMAScript. Por definición el lenguaje es orientado a objetos, basado en prototipos, imperativo y dinámico.

**Journal Citation Report:** Publicación que evalúa el índice de impacto de diferentes revistas científicas a fin de establecer una relación de importancia.

**JSON (JavaScript Object Notation):** Representa un subconjunto de la notación literal de objetos de JavaScript. Se utiliza como formato de intercambio de texto.

**Lenguaje de Dominio Específico:** Lenguaje de programación destinado a resolver un problema en particular, restringido a un dominio concreto.

**MOF:** Estándar para la Ingeniería Dirigida por Modelos. Tiene como objetivo proporcionar un sistema de tipos para entidades en la arquitectura CORBA.

**NoSQL:** Sistemas de almacenamiento que presentan diferencias sustanciales con respecto a las bases de datos relacionales (RDBMS) en el sistema de consulta, la estructura de la información (no emplean estructuras fijas) ni garantizan atomicidad, coherencia, aislamiento y durabilidad. Se orientan hacia el almacenamiento de grandes cantidades de datos.

**Object Management Group:** Consorcio dedicado a la estandarización de múltiples tecnologías orientadas a objetos, tales como UML, XMI, o BPMN.

**Python**: Lenguaje de programación dinámico diseñado con la intención de potenciar la legibilidad del código que se genera por medio de él. Soporta varios paradigmas, entre ellos el paradigma orientado a objetos y el paradigma funcional. Se define como un lenguaje interpretado, dinámico y multiplataforma.

**RFID (*Radio Frequency Identification*):** Sistema remoto de almacenamiento y obtención de datos que emplea dispositivos denominados etiquetas RFID para transmitir la identidad de un objeto.

**Smart City**: Desarrollo de las ciudades por medio políticas de sostenibilidad para la mejora de los servicios con los que cuentan estas. En muchos casos de emplean políticas basadas en la colaboración de los usuarios (crowdsourcing).

**Smart Earth**: Concepto que trata de expandir el Internet de las Cosas a cualquier elemento de la tierra, tales como edificaciones, carreteras, etc.

**Smart Home**: Concepto que propugna el empleo de tecnología para la realización de tareas domésticas de manera automática en el hogar.

**XMI (*XML Metadata Interchange*)**: Especificación para el intercambio de diagramas, destinada a la compartición de diagramas en formato UML.

**XML** (***eXtensible Markup Language***): Metalenguaje del W3C emplelado en la definición de lenguajes de marcas. Se usa para el almacenamiento de datos de manera legible.

# Anexo III: Artículo JCR - *CrowDSL: Platform for incidents management in a Smart City context*


| |
|---|
| Journal: **IEEE Internet of Things Journal** [46] |
| Manuscript ID: **IoT-7131-2019** |
| Manuscript Type: **Regular Article** |
| Date Submitted by the Author: **19-Jun-2019** |



Complete List of Authors**:**

**Rodríguez García, Darío; University of Oviedo, Computer Science**

**García Díaz, Vicente; University of Oviedo, Computer Science**

**González García, Cristian; University of Oviedo, Computer Science**



Keywords: **Internet of Things, Smart City, Domain-Specific Language, Model-Driven Engineering, Crowdsourcing, Citizens, Incidents reporting**

# CrowDSL: Platform for incidents management in a Smart City Context

Darío Rodríguez-García, Vicente García-Díaz and Cristian González García

***Abstract*—The final objective of Smart Cities is to optimize services and improve the quality of life of their citizens, playing these last an important role due to information they can provide in order to enhance many sectors involved in city activity such as transport, energy or health. Crowdsourcing initiatives focus their efforts on making cities safer places and adapted to the population size they host. In this way, citizens are allowed to report the issues they identify to the competent body so that they could be fixed, and at the same time, they can provide useful information to other citizens. There are several projects aimed at reporting incidents in a Smart City context. In this paper, we propose the use of Model-Driven Engineering by designing a graphical Domain-Specific Language in order to abstract and improve the incidents reporting process. With the use of a DSL we can obtain several benefits in our research. For instance, we can shorten the time for reporting the events by users and at the same time we gain a expressive power compared to another methodologies for incidents reporting.**

***Index Terms*—Smart City, Internet of Things, Domain-Specific Language, Model-Driven Engineering, Crowdsourcing, Citizens, Incidents reporting.**

## I. Introduction

OVER the last years, the Internet of Things (IoT) has been positioned as one of the disciplines with more impact in the world of communication technologies [1][2][3]. Until recently, the use of the Internet was conceived only as a communication network between humans, so that they could communicate and share information among themselves. This paradigm has progressively evolved towards a new approach in which a notion of interconnected "smart" objects form pervasive computing environments [4]. Attending this reasoning, one of the objectives of the IoT is to achieve that different objects that share the same environment are able to interact with each other and cooperate with their neighbors in order to achieve common goals [5]. Within the technologies that the IoT encompasses there are a large number of objects and devices that underline the heterogeneity that exists between the different nodes that we try to communicate. The IoT can be applied to not-smart [6] or inanimate objects like pallets, boxes containing consumer goods, cars, machines, fridges as well as animate objects like animals and humans [7]. To achieve the communication and monitoring of these entities, a wide range of devices are used, such as sensors, actuators or smart tags like Radio Frequency Identification (RFID) and Near Field Communication (NFC), many of which have integrated intelligence and communication capabilities [8].

Over time, new concepts have emerged and achieved a strong impact and acceptance by society in developed countries. Terms such as Smart Object, Smart Home, Smart City or Smart Earth are more and more taken into account due to the the benefits they could provide to the future society. All these concepts pursue a common objective, the improvement of the quality of life of users of these technologies. Many of these tools are focused on optimizing the services that people use, both in a domestic environment and at the level of coexistence in cities. Two clear examples of these are Smart Homes and Smart Cities. The term "smart home" is used to design a residence equipped with technology that allows monitoring ofits inhabitants and/or encourages independence and the maintenance of good health [9]. Many of these concepts are aligned with other demographic and sociological ones that predict a tendency towards an increasingly aging society, being indispensable to condition the habitable spaces of cities to the social circumstances of each moment. There are many definitions of Smart City that have been stated since its inception. These cities are conceived as a space oriented towards the progressive improvement of the government, economy or transport [10]. They also strive to become "smarter", understanding this smartness as its efficiency, sustainability and capacity [11] and sharing their culture and knowledge by encouraging the citizens to participate actively [12] in the activity of the city. These cities will be even more important in future because large cities are expected to increase their volume of population over the next decades in a high percentage. According to the forecasts that are collected, in 2050 the number of people living in the cities could be placed at 5,2 billion, compared to 2,6 billion that were recorded in 2010 [13].

With this information, it seems obvious that one of the fundamental instruments for the cities of the future is the cooperation between citizens in order to achieve an improvement in coexistence and an optimization of services in urban areas. In this paper, we propose the design and development of a platform for reporting and managing the incidents that are originated in the day-to-day of a city. One way of getting that is the use of Model-Driven Engineering by implementing a Domain-Specific Language. This approach can provide multiple advantages to our research. These languages are applied to a concrete domain, abstracting all the information required. There are several projects that use a DSL in order to facilitate the users without development knowledge certain tasks regarding to a specific domain. Furthermore, they also

(Corresponding author: Darío Rodríguez-García.)

D. Rodríguez-García, V. García-Díaz and C. González García. Department of Computer Science, University of Oviedo, 33007 Oviedo, Spain (e-mail:uo231097@uniovi.es; garciavicente@uniovi.es; gonzalezcristian@uniovi.es).

can reduce the difficulty of a task getting a better adaptation to a concrete problem. Some examples are MOCSL (Midgar Object Creation Specific Language) [14], a graphical language designed to interconnect Smart Objects, MUCSL (Midgar Use Case Specification Language) [15], with the same purpose but a textual DSL and XPDML (eXtensible Process Definition Markup Language) [16], used in food traceability processes. In this work, we have designed a Domain-Specific Language (DSL) in order to allow citizens of Smart Cities to abstract data collection process and report all the anomalous events originated in their path. Using this DSL, users can categorize the incidents that are reported as well as specify the elements involved in them (means of transport, buildings, human beings, etc.). The goal of this proposal is to make the information accessible to the greatest number of people.

Most of the existing projects make use of different approaches and technologies in order to perform the incidents capture, processing and storage process. In order to involve the users in the city activity and achieve increasingly a wider participation of them, we must make the issues reporting process as easy and intuitive as possible.

The rest of the work is structured as follows: Section 2 presents the state of the art of our paper, including the related work. Then we explain the case of study in Section 3, The following part is used to evaluate and discuss the results (Section 4) and, finally, we present the conclusions and future work.

## II. State of the art

In this section, we explain some concepts such as Smart City, Model-Driven Engineering, Domain-Specific Languages and the related work in order to present the theoretical frame of our research.

### A. Smart City

There are a lot of definitions about the Smart City concept. In [17] a Smart City is defined as the use of Smart Computing technologies to make the critical infrastructure components and services of a city more intelligent, interconnected, and efficient. Meanwhile, in [18], the authors present this concept as a city well performing in a forward-looking way in six characteristics: smart economy, smart people, smart governance, smart mobility, smart environment and smart living [19], built on the "smart" combination of endowments and activities of self-decisive, independent and aware citizens. From the point of view of information management, Smart City is defined in [20] as the use of communication technologies in order to strengthen the freedom of speech and the accessibility to public information and services.

### B. Model-Driven Engineering

Model-Driven Engineering (MDE) proposes the creation of software models as the focus and primary artifacts of development rather than programs [21], with the goal of both simplifying (making easier) and formalizing (standardizing, so that automation is possible) the various activities and tasks that comprise the software life cycle [22]. MDE tools impose domain-specific constraints and perform model checking that can detect and prevent many errors early in the life cycle [23]. This set of technologies is usually the basis for designing and development of the Domain-Specific Languages (DSLs).

### C. Domain-Specific Languages

A Domain-Specific Language (DSL) is a language or executable specification language that offers expressive power focused on, and usually restricted to, a particular problem domain [24]. DSLs represent one of the main practical applications of MDE. They provide substantial gains in expressiveness and ease of use compared with general-purpose programming languages in their domain of application [25]. A prerequisite for the design of a DSL is a detailed analysis and structuring of the application domain [26].

### D. Related work

Our project focuses on incidents reporting process in a Smart City context. But previously of the development of this project some other platforms had been designed and implanted in several cities. This section shows the main features of some of the most important platforms.

*1) mPASS and WhenMyBus [27]:* This project was developed in order to provide personalized paths to users with special needs. System architecture is composed of two services that work together, mPASS and WhenMyBus. mPASS has the function of collecting data about urban accessibility, both from sensors and data gathered via crowd-sourcing by app users. As a result, the platform obtains georeferenced data, allowing us to identify some urban barriers and providing accessible pedestrian paths to citizens. mPass has an urban accessibility module that helps to define preferences about aPOIs (accessibility Points of Interest). This way, a user can assign states (*neutral, like, dislike, avoid*) to a certain aPOI (*gap, cross, obstruction, parking, surface, pathway, bus stop, bus*) in order to calculate an optimized path for a certain destination. WhenMyBus has been developed to provide useful information to citizens who travel by bus in the city. In addition, WhenMyBus allow users to get real time information about bus availability and equipment, specifying those buses which present some barriers for citizens with disabilities.

*2) CrowdOut [28]:* CrowdOut is a platform that allows users to report traffic offences they witness in real time to map them on a city plan.

The main objective of the app is the reporting of road safety issues (excessive speed, cars illegally parked, signs and signals) grouping them in several categories (road light, pedestrians, parking, road, speed, road flow, cyclist, overtake and others) so that these could be reported to other citizens, representing them in a map and differentiating between dangerous driving areas and safety zones. App users can select which offenses they are watching at any time. User reports contain a set of attributes that allow system to locate the place from which the offense has been sent and its description.

*3) FixMyStreet (FMS) [29] [30]:* FixMyStreet is a system that enabled citizens to report, view and discuss local problems such as graffiti, fly tipping, broken paving slabs or street lighting, and to track their resolution by the local government concerned [29]. Problems reported by citizens are delivered to the relevant authority or body via email. Issues are categorized into several categories (*car parking, graffiti, potholes, abandoned vehicles and so on*). In case citizens need to report a non-listed issue they are advised to contact their councils directly [30]. One of the best advantages this project has is the monitoring and evaluation of citizen reports. Thus, the problem originator is contacted by FMS four weeks later to see if the problem has been fixed [29] in order to evaluate the issue state.

*4) CrowdSC [31]:* CrowdSC aims to answer questions such as the roads that are damaged in a certain place or the areas that are slovenly in specific locations. In this way, a council ask for citizens collaboration in order to know the state of a certain point of a city. Authors divide the whole process into other three sub-processes: data collection, data selection and data assessment. The data collection phase consists in querying just a few citizens, who provide a set of images of damaged roads or similar. In this phase only some citizens are queried in order to avoid receiving subjective or mistaken judgments from them. In second phase, known as data selection, some other citizens are asked to select the most representative photo for each location. In the last phase, data assessment, other citizens assess the priority of each selected photo.

All studied platforms have similar features and functionalities. They all focus on incidents management in a city context using several technologies and approaches in order to achieve their objectives. For example, it is common in all solutions the use of maps in order to represent the incidents or events reported by users. Another common feature in many projects is the fact that many of them incorporate an user profile in order to track the user activity in the platform. Our proposal includes many of this features, but the main contribution of our paper is the use of Model-Driven Engineering, making use of a Domain-Specific Language in order to abstract and simplify the incidents modelling process. These languages can provide several benefits to our projects, such as simplicity, usability or integrability [32], because they have notations and constructs tailored toward a particular application domain [25]. Furthermore, DSLs embody domain knowledge, and thus enable the conservation and reuse of this knowledge [24]. In the following section we explain how our platform was designed and the way it works.

## III. Case of Study: CrowDSL

The aim of our proposal is to allow Smart City citizens to model all the entities, scenarios and sectors of the city that are involved in the occurrence of an incident. In order to carry out the report of the aforementioned incidents, we have developed a web editor that allows users to quickly become familiar with the environment and gradually acquire the knowledge necessary to operate in a normal way with the Domain-Specific Language developed. Once the incidences are reported and validated, they are published in the same platform.

### A. Architecture

The platform of our research is structured in three layers that cover the full functionality of project. Figure 1 shows a diagram of our proposed architecture. The first layer, "Data definition", deals with the models definition process. This layer is composed of the DSL that can be used by means of the editor. In this first phase, we obtain an XML (eXtensible Markup Language) document that contains the entities data and the relationships among them modelled by the user. This information is sent to the second layer, "Data Processing", where the XML document is used to represent a tree in memory that replicates the structure of the file received from the upper layer. This second layer has the objective of validating that model structure complies with the specifications defined in the application meta-model in multiple aspects, such as the entities relationships and data types specified by application users. Once the entire validation process has been completed, the incidents are stored in JSON format in the application database. When the incidents are stored, they are published through the third layer of the architecture, "Data visualisation", which allows the rest of the users of the application to consult all the attached information. Multiple forms of information are proposed for this third layer: the information specified by the user linked to each entity in plain text, multimedia elements provided by the user who reports the incident (images, videos, tweets or geolocation), a parameterized map that allows users to know the location of the incident and, finally, a comments section for each user to give a personal view.

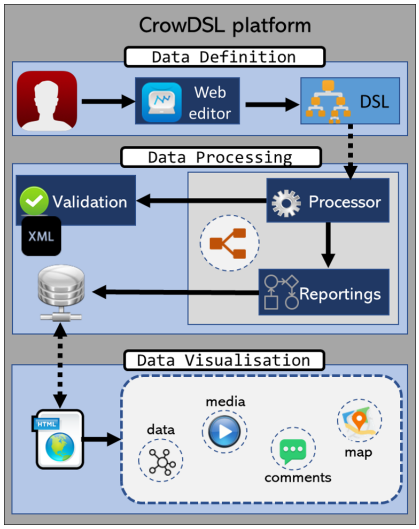



Fig. 1. CrowDSL architecture

### B. CrowDSL: the Domain-Specific Language

The first step was the definition of application meta-model. This meta-model contains a wide range of elements that try to represent all existing entities, sectors and scenarios of a Smart City. The meta-model consists of 5 sets of elements, including general elements (user, city, map), alert types (flood, fire, accident, damages, ...), objects (person, animal, house, car, ...), city sectors [19] (economy, people, governance, mobility, environment, living) and multimedia elements (images, videos,

geolocation, tweets, ...).
Following one of the principles of MDE, the next step was to define the concrete syntax of elements from the abstract syntax. Table 1 shows the correspondence between the two syntax for some of the main elements of language.

Table 1. CrowDSL abstract and concrete syntax

| Abstract syntax | Concrete syntax |
|---|---|
| User | |
| City | |
| Map | |
| Fire (Alert) | |
| Person (Object) | |
| Mobility (Sector) | |
| Image (Media) | |

As has been mentioned previously, we have developed a web editor in order to work with CrowDSL. Figure 2 shows the look and feel of web editor. This editor is composed of multiple sections that allow users to specify all the information related to the incidence they are reporting at any time. Now we detail the purpose of each section:

- A - Palette of components. With this tool, users drag and drop components to the editor and define relationships among elements.
- B - Editor. This element allows users to watch the incident structure they are reporting at that time.
- C - Properties. This panel shows all properties of the component selected in the editor. This way, users can provide values to entities attributes.
- D - Log. This element shows the feedback to users. Once incidents are sent, they can contain validation errors. These errors are displayed through this component.
- E - Incident representation. Once an incident is validated, in this box it is shown a JSON format representation of it in order to check all data and structure specified.

Now we are going to explain the incidents validation process since they are modelled in the editor until they are stored in a database and published for other users.

### C. Incidents validation process

In Figure 2, where the web editor is displayed, it is presented an example of incident. This incident describes a traffic accident where a car and a truck are involved. The incident representation contains several elements: the user of application who reports the incident (black circle), the city where the incident occurs (blue square), the type of incident (traffic accident, orange square), the map on which the incident will be represented (red square), the "objects" implied (car and truck, blue circles), the sector of the city affected and the media elements provided by the user (image and video, white squares). Figure 3 shows the graphical representation of this incident.

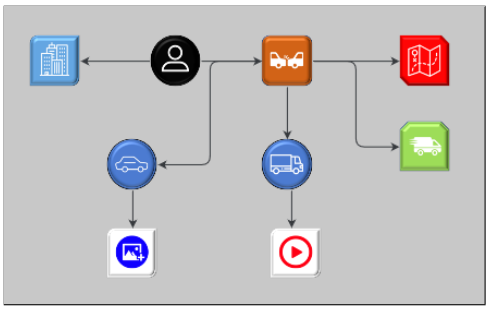


Fig. 3. Example of incident reporting

As users model the relationships between the different nodes that make up the language, an internal representation of the model is generated in an intermediate XML format. When the user clicks the "Validate" button, the process of validation of structure and data of the model is triggered. Validation process in done by means of an XML Schema for an XMI (XML Metadata Interchange) document that represents the meta-model of the application. This XML Schema follows the XMI standard [33] and allows us to perform all validations specified during the meta-model design phase. Therefore, the first step is to make a transformation between the native XML to an XMI format in order to validate each model against the XML Schema for XMI obtained. This conversion is materialized by first representing the source XML as a tree in memory in which each node is generated by means of a model class that defines the structure of the element. Starting from that representation, an XMI file is generated by using a Visitor design pattern over the tree in order to perform the aforementioned validation. If during the process an error is detected in the validation process, these errors are represented in the editor in order to provide a feedback to the user. If the validation process concludes with no errors, the user is allowed to send the incident using the "Save" button. When the structure and content of an incident is validated, a representation of it is generated in JSON-LD format that facilitates its storage in a NoSQL documents store database to be used later. Figure 4 shows the resulting graphical representation of the incident. This information will be published to all platform users.

Once incident has been reported, it is published in the platform. Figure 4 shows a representation of the incident.

## IV. Evaluation and Discussion

In this section, we explain the process followed to evaluate our proposal by introducing the methodology used and the results obtained.
The objective of this proposal was to check if it was possible to abstract the incidents capture process in a Smart City context by means of a Domain-Specific Language.

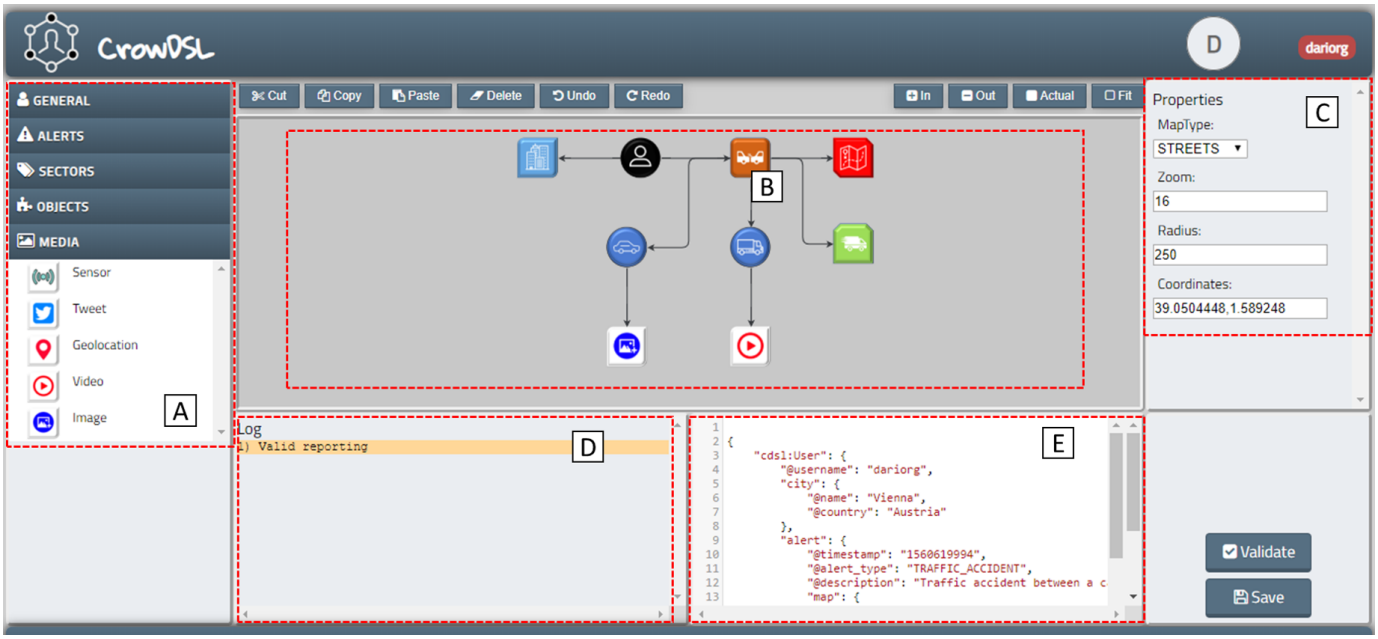



Fig. 2. CrowDSL Web editor

```
{
  "cdsl:User": {
    "@username": "dariorg",
    "city": {
      "@name": "Viena",
      "@country": "Austria"
    },
    "alert": {
      "@timestamp": "1558255082",
      "@alert_type": "TRAFFIC_ACCIDENT",
      "@description": "Collision between a car and a truck",
      "map": {
        "@map_type": "STREETS",
        "@zoom": "15",
        "@radius": "250",
        "@coordinates": "48.210033,16.363449"
      },
      "service": {
        "@service_type": "MOBILITY"
      },
      "object": [
        {
          "@name": "BMW",
          "@description": "Serie 3",
          "@danger_level": "NO_DANGER",
          "@object_type": "CAR",
          "data_object": {
            "@url": "https://db.com/image"
          }
        },
        {
          "@name": "Mercedes",
          "@description": "Benz",
          "@danger_level": "NO_DANGER",
          "@object_type": "TRUCK",
          "data_object": {
            "@url": "https://db.com/video"
          }
        }
      ]
    }
  }
}
```

## A. Methodology

In order to validate our research questions, we used a methodology focused on the evaluation of DSLs. This methodology has been used in another works such as [15] and [14]. In this way, we asked a set of 20 users for evaluating our prototype following a practical example. The practical example was as follows:

*"User witnesses a traffic accident involving two vehicles, a car and a truck. The collision takes place in the city of Vienna (Austria). The severity of the accident can be considered as high. The car suffers serious damages and the truck is damaged moderately. The incident radius of the accident is 50 meters".*

Users who participate in evaluation have different levels of knowledge in the field of IT (Information Technology) and The Internet of Things, representing a wide range of profiles: novel, middle-level and professional users.

Before the beginning of evaluation each user was instructed in making use of both the DSL and the editor. Explanation just took a few minutes and contained a tiny set of operations with editor, such as add and connect elements, specify attributes of objects and interpret log errors, based on another small and similar practical case. In order to allow users to include multimedia elements, an image and a video were provided to them. These elements were supposed to be captured by users in the place of the incident.

Once users finished the incident modelling, they were asked for filling out a survey in order to measure their degree of satisfaction with the platform. This survey contains the following declarations as shows Table 2.

Table 2. Survey declarations

| # | Declaration |
|---|---|
| **D1** | The elements of the language are well defined and their purpose are understood by users. |
| **D2** | The elements of the language are adequate to model any type of incidence. |
| **D3** | It is easy to specify the values for the attributes of elements (level of danger, description, ...). |
| **D4** | The logical structure of the relationships between entities is understood. |
| **D5** | You prefer this methodology rather than sending the incidents in text mode |
| **D6** | The editor elements are well structured and it is easy to work with it. |
| **D7** | The web editor is useful for modelling city incidents. |
| **D8** | The user understands the log errors of the application. |
| **D9** | This work would be useful in a Smart City. |
| **D10** | The degree of satisfaction with platform is high. |

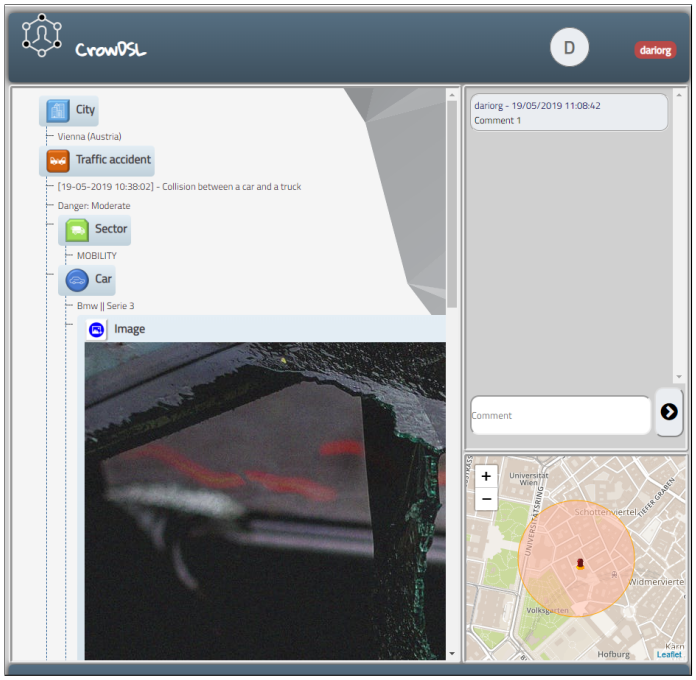



Fig. 4. Visualization of an incident

Table 3. Users responses to survey declarations

| D/U | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 |
|---|---|---|---|---|---|---|---|---|---|---|
| U1 | 4 | 3 | 5 | 3 | 5 | 5 | 5 | 2 | 5 | 4 |
| U2 | 5 | 4 | 5 | 4 | 5 | 5 | 4 | 3 | 5 | 5 |
| U3 | 5 | 4 | 5 | 3 | 5 | 5 | 5 | 1 | 5 | 4 |
| U4 | 5 | 4 | 5 | 3 | 5 | 5 | 5 | 2 | 5 | 4 |
| U5 | 5 | 3 | 5 | 4 | 4 | 5 | 5 | 2 | 4 | 4 |
| U6 | 4 | 4 | 5 | 3 | 5 | 3 | 4 | 1 | 4 | 4 |
| U7 | 5 | 5 | 5 | 3 | 5 | 4 | 4 | 0 | 5 | 5 |
| U8 | 3 | 5 | 5 | 3 | 5 | 5 | 4 | 2 | 5 | 3 |
| U9 | 5 | 5 | 4 | 5 | 5 | 5 | 5 | 3 | 4 | 4 |
| U10 | 5 | 4 | 5 | 3 | 5 | 5 | 5 | 1 | 4 | 5 |
| U11 | 5 | 5 | 4 | 3 | 5 | 5 | 5 | 0 | 4 | 5 |
| U12 | 4 | 4 | 5 | 4 | 5 | 5 | 5 | 3 | 5 | 5 |
| U13 | 3 | 3 | 5 | 4 | 5 | 4 | 4 | 4 | 5 | 4 |
| U14 | 5 | 3 | 5 | 3 | 5 | 4 | 4 | 5 | 5 | 4 |
| U15 | 4 | 4 | 5 | 3 | 4 | 4 | 4 | 1 | 5 | 5 |
| U16 | 4 | 5 | 5 | 4 | 4 | 4 | 3 | 2 | 5 | 4 |
| U17 | 5 | 4 | 5 | 5 | 5 | 5 | 4 | 2 | 5 | 4 |
| U18 | 5 | 4 | 5 | 4 | 5 | 5 | 4 | 2 | 5 | 5 |
| U19 | 5 | 5 | 5 | 4 | 5 | 5 | 5 | 3 | 5 | 5 |
| U20 | 5 | 4 | 5 | 4 | 5 | 5 | 5 | 1 | 5 | 5 |

In order to establish a minimum and a maximum value for the responses it has been used the Likert Scale [34], giving the users the following options to evaluate each declaration: 1 as strongly disagree, 2 as disagree, 3 as somewhat agree, 4 as agree, and 5 as strong agree.

### *B. Results*

Table 3 shows all the responses provided by the users for each declaration.

Once we gathered all data, we studied several descriptive statistics applied for all the set. We analyzed the minimum, quartile 1, median, quartile 3, maximum, range, interquartile range and mode of each declaration of the survey. Then, we represent all data using a box and whiskers plot diagram. Table 4 and Figure 5 show the results.

Table 4. Descriptive statistics

| | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Min** | 3 | 3 | 4 | 3 | 4 | 3 | 3 | 0 | 4 | 3 |
| **Quartile 1** | 4 | 4 | 5 | 3 | 5 | 4 | 4 | 1 | 4,25 | 4 |
| **Median** | 5 | 4 | 5 | 3,5 | 5 | 5 | 4,5 | 2 | 5 | 4 |
| **Quartile 3** | 5 | 5 | 5 | 4 | 5 | 5 | 5 | 3 | 5 | 5 |
| **Max** | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| **Range** | 2 | 2 | 1 | 2 | 1 | 2 | 2 | 5 | 1 | 2 |
| **Inter Qrt.-Range** | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 2 | 0,75 | 1 |
| **Mode** | 5 | 4 | 5 | 3 | 5 | 5 | 5 | 2 | 5 | 4 |

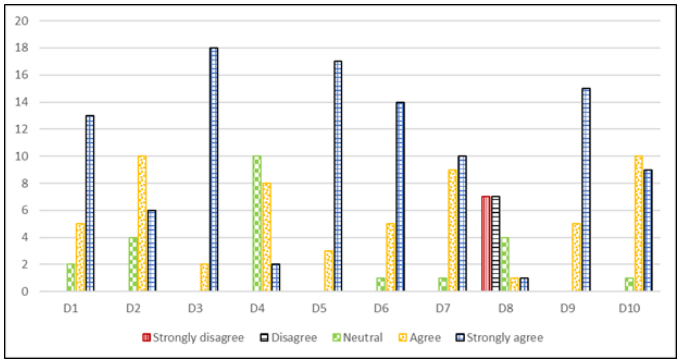



Fig. 6. Frequency of answers by declaration

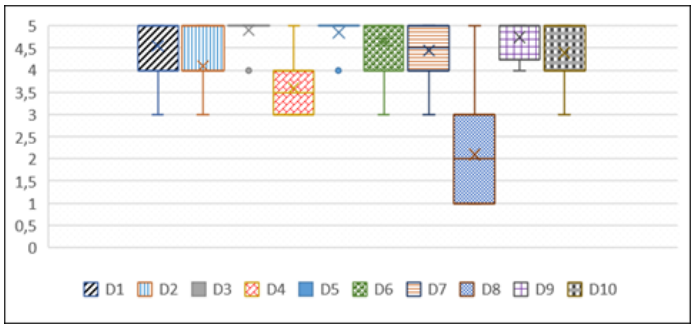



Fig. 5. Descriptive statistics in box and whiskers plot diagram

By analyzing the results we can do some interpretations:

- D3, D5 and D9 have the highest minimum, so the vast majority of people agreed with those ones, at least.
- D1, D3, D5, D6 and D9 have the highest median, so users agreed in platform utility and usability.
- The negative side of this study is shown in D8, where users estimate that the log errors of the application are not understandable enough. This is why the mode and median have a value of 2 (disagreement).
- All declarations have a high maximum, with a value of 5 (strong agreement).
- The mode is between 4 (agree) and 5 (strongly agree) in 8 of the 10 declarations, so mostly, users agreed with those statements.

Regarding the frequency of answers by declaration, we tried to find some other patterns that could not been interpreted in the data presented until now. Table 5 and Figure 6 shows the frequency of answers by each declaration in survey.

This way, we can get new interpretations of data:

Table 5. Frequency of answers by declaration

| Declaration | | Strongly disagree | Disagree | Neutral | Agree | Strongly agree |
|---|---|---|---|---|---|---|
| **D1** | # | 0 | 0 | 2 | 5 | 13 |
| | % | 0% | 0% | 10% | 25% | 65% |
| **D2** | # | 0 | 0 | 4 | 10 | 6 |
| | % | 0% | 0% | 20% | 50% | 30% |
| **D3** | # | 0 | 0 | 0 | 2 | 18 |
| | % | 0% | 0% | 0% | 10% | 90% |
| **D4** | # | 0 | 0 | 10 | 8 | 2 |
| | % | 0% | 0% | 50% | 40% | 10% |
| **D5** | # | 0 | 0 | 0 | 3 | 17 |
| | % | 0% | 0% | 0% | 15% | 85% |
| **D6** | # | 0 | 0 | 1 | 5 | 14 |
| | % | 0% | 0% | 5% | 25% | 70% |
| **D7** | # | 0 | 0 | 1 | 9 | 10 |
| | % | 0% | 0% | 5% | 45% | 50% |
| **D8** | # | 7 | 7 | 4 | 1 | 1 |
| | % | 35% | 35% | 20% | 5% | 5% |
| **D9** | # | 0 | 0 | 0 | 5 | 15 |
| | % | 0% | 0% | 0% | 25% | 75% |
| **D10** | # | 0 | 0 | 1 | 10 | 9 |
| | % | 0% | 0% | 5% | 50% | 45% |

- Users strongly agree in a very high percentage in D3 (90%), highlighting the facility to provide data to language attributes of elements.
- In D5, a 85% of users prefer this reporting methodology rather than sending incidents in text mode.
- Half of all users (50%) agreed in being satisfied with the platform and a 45% strongly agree with this statement.

## V. Conclusions

This research aimed to provide a new method for incidents reporting to citizens of Smart Cities. Our proposal uses Model-Driven Engineering by designing and building a Domain-Specific Language in order to tag, link and process all information gathered via crowdsourcing by users. In this way, information could be used by several technologies due to the fact that information is stored using a JSON format, ensuring it interoperability.

Once we finished the platform development and its future assessment, we checked that the vast majority of users showed a very good opinion about our DSL, with a 95% of satisfaction with platform. Users highlight in the survey some features and characteristics such as the usability, utility and well design of solution. But there are also some features of the DSL and the platform that could be improved, for instance the log system or the possibility of adding new elements to the language in the future.

The main contribution of our research compared to the other studied platforms was the using of MDE. This approach allowed us to provide many features to our application such as reusability, interoperability and portability.

## References


[1] N. Council, "Six technologies with potential impacts on us interests out to 2025," *Disruptive Civil Technologies 2008*, 2008.

[2] I. Property and O. Informatics, "Eight Great Technologies The Internet of Things."

[3] M. Walport, "The internet of things: making the most of the second digital revolution," *UK Government Chief Scientific Adviser*, 2014.

[4] D. Miorandi, S. Sicari, F. D. Pellegrini, and I. Chlamtac, "Ad Hoc Networks Internet of things : Vision , applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2012.02.016

[5] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things : A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2010.05.010

[6] C. González García, D. Meana Llorián, B. C. Pelayo García-Bustelo, and J. M. Cueva Lovelle, "A review about smart objects, sensors, and actuators," *International Journal of Interactive Multimedia and Artificial Intelligence*, 2017.

[7] S. Haller, "The Things in the Internet of Things," no. January 2010, 2016.

[8] O. Bello, S. Zeadally, and S. Member, "Intelligent Device-to-Device Communication in the Internet of Things," pp. 1–11, 2020.

[9] M. Chan, E. Campo, D. Estève, and J.-y. Fourniols, "Maturitas Smart homes — Current features and future perspectives," vol. 64, pp. 90–97, 2009.

[10] R. Giffinger, C. Fertner, H. Kramar, and E. Meijers, "City-ranking of European Medium-Sized Cities," pp. 1–12.

[11] N. R. D. Council, "What are smarter cities?" [Online]. Available: https://smartcitiescouncil.com/members/natural-resources-defense-council

[12] Patrice Rios, "Creating "the smart city"," 2008. [Online]. Available: https://bit.ly/2ZsEV4u

[13] United Nations, "Population Distribution, Urbanization, Internal Migration and Development: An International Perspective," 2011.

[14] C. G. García, B. C. P. G-Bustelo, J. P. Espada, and G. Cueva-Fernandez, "Midgar: Generation of heterogeneous objects interconnecting applications. a domain specific language proposal for internet of things scenarios," *Computer Networks*, vol. 64, pp. 143–158, 2014.

[15] C. González García, L. Zhao, and V. García-Díaz, "A user-oriented language for specifying interconnections between heterogeneous objects in the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3806–3819, April 2019.

[16] V. García Díaz, H. Fernandez, E. Palacios-González, B. Pelayo García-Bustelo, O. Sanjuán, and J. Cueva Lovelle, "Automated code generation support for bi with mda talisman," *International Journal of Artificial Intelligence and Interactive Multimedia*, vol. 1, pp. 87–93, 12 2009.

[17] D. Washburn, U. Sindhu, S. Balaouras, R. A. Dines, N. M. Hayes, and L. E. Nelson, "Helping CIOs Understand " Smart City " Initiatives," 2010.

[18] R. Science and M. Studies, "Smart cities Ranking of European medium-sized cities," no. October, 2007.

[19] 'Vienna University of Technology', "European Smart Cities," 2015. [Online]. Available: http://www.smart-cities.eu

[20] T. Nam and T. A. Pardo, "Conceptualizing Smart City with Dimensions of Technology , People , and Institutions," pp. 282–291, 2011.

[21] B. Selic, "Model-driven development of real-time software using omg standards," in *Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, 2003.*, May 2003, pp. 4–6.

[22] B. Hailpern and P. Tarr, "Model-driven development : The good , the bad , and the ugly," vol. 45, no. 3, pp. 451–461, 2006.

[23] D. C. Schmidt, "Model-Driven Engineering," vol. 39, no. 2, pp. 25–31, 2006.

[24] A. V. Deursen, P. Klint, and J. Visser, "Domain-Specific Languages : An Annotated Bibliography *," vol. 35, no. June, pp. 26–36, 2000.

[25] M. Mernik, J. A. N. Heering, and A. M. Sloane, "When and How to Develop Domain-Specific Languages AND," vol. 37, no. 4, pp. 316–344, 2005.

[26] A. V. Deursen and P. Klint, "Domain-Specific Language Design Requires Feature Descriptions *," pp. 1–17, 2002.

[27] S. Mirri, C. Prandi, P. Salomoni, F. Callegati, and A. Campi, "On combining crowdsourcing, sensing and open data for an accessible smart city," *Proceedings - 2014 8th International Conference on Next Generation Mobile Applications, Services and Technologies, NGMAST 2014*, pp. 294–299, 2014.

[28] E. Aubry, T. Silverston, A. Lahmadi, and O. Festor, "CrowdOut: A mobile crowdsourcing service for road safety in digital cities," *2014 IEEE International Conference on Pervasive Computing and Communication Workshops, PERCOM WORKSHOPS 2014*, pp. 86–91, 2014.

[29] S. F. King and P. Brown, "Fix My Street or Else: Using the Internet to Voice Local Public Service Concerns," *Computers and Society*, pp. 72–80, 2007.

[30] B. Baykurt, "Redefining Citizenship and Civic Engagement : political values embodied in FixMyStreet . com," *Anais do AoIR – Association of Internet Research, Seattle*, pp. 1–18, 2011.

[31] K. Benouaret, R. Valliyur-Ramalingam, F. Charoy, and R. V.-R. F. C. Karim Benouaret, "CrowdSC: Building Smart Cities with Large Scale Citizen Participation," *IEEE Internet*

*Computing*, vol. 17, p. 1, 2013. [Online]. Available: www.computer.org/csdl/mags/ic/2013/06/mic2013060057.html
[32] D. S. Kolovos, R. F. Paige, T. Kelly, and F. A. C. Polack, "Requirements for Domain-Specific Languages," pp. 1–4.
[33] OMG, "About the XML metadata interchange specification version 2.5.1," 2015. [Online]. Available: https://www.omg.org/spec/XMI/2.5.1/
[34] R. Likert, "A technique for the measurement of attitudes." *Archives of psychology*, 1932.

**Darío Rodríguez-García** is a Graduated Engineering in Computer Systems from School of Computer Engineering of Oviedo in 2017 (University of Oviedo, Spain). Currently, he is a student of M.S. in Web Engineering. His research interests include Web Engineering, the Internet of Things, Model-Driven Engineering and Domain-Specific Languages.

**Vicente García-Díaz** received the Ph.D. degree in computer science from the University of Oviedo, Oviedo, Spain, in 2011. He is a Software Engineer with the University of Oviedo, where he is an Associate Professor with the Department of Computer Science. His research interests include machine learning, natural language processing, model-driven engineering and domain-specific languages. Dr. García-Díaz is also part of the Editorial and Advisory Board of several journals and has been editor of several special issues in books and journals. He has supervised over 90 academic projects and has had over 70 research papers in journals, conferences and books published.

**Cristian González García** is a Technical Engineer in Computer Systems, M.Sc. in Web Engineering, and a Ph.D. in Computers Science graduated from the School of Computer Engineering of Oviedo, University of Oviedo, Oviedo, Spain, in 2011, 2013, and 2017, respectively. He has been a visiting Ph.D. candidate with the University of Manchester. Besides, he has been in the University of South Florida as visiting professor. He has also been working in different national and regional projects, as well as in projects with private companies. He has published 14 journal articles, 6 conference articles, 3 book chapters, and has been the editor of 1 book. Furthermore, he has been in the advisory board of 1 journal, 1 book, and 1 international conference. His research interests include the Internet of Things, Web Engineering, Mobile Devices, Artificial Intelligence, and Modelling Software with DSL and MDE.