



Universidad de Oviedo

Memoria del Trabajo Fin de Máster realizado por

David González Aguado

para la obtención del título de

Máster en Ingeniería de Automatización e Informática Industrial

**Aplicación de escritorio para visualización de datos de  
análisis del proceso electrolítico del aluminio**

(Junio de 2015)

**Aplicación de escritorio para  
visualización de datos de análisis del  
proceso electrolítico del aluminio**

**Memoria**

**David González Aguado**

# Índice

<b>APLICACIÓN DE ESCRITORIO PARA VISUALIZACIÓN DE DATOS DE ANÁLISIS DEL PROCESO ELECTROLÍTICO DEL ALUMINIO    MEMORIA    DAVID GONZÁLEZ AGUADO .....</b>	<b>0</b>
<b>ÍNDICE.....</b>	<b>1</b>
<b>ÍNDICE DE ILUSTRACIONES.....</b>	<b>3</b>
<b>1    INTRODUCCIÓN .....</b>	<b>4</b>
1.1    Identificación del proyecto.....	4
1.2    Descripción del documento.....	4
<b>2    EL PROCESO DEL ALUMINIO .....</b>	<b>5</b>
2.1    Introducción .....	5
2.2    La producción del aluminio .....	5
2.2.1    Minería .....	5
2.2.2    El proceso Bayer .....	7
2.2.3    El proceso Hall Heroult de electrólisis .....	10
2.3    Las tecnologías de electrólisis de aluminio .....	14
2.3.1    Cubas Soderberg .....	15
2.3.2    Cubas de ánodo precocido .....	16
<b>3    DESCRIPCIÓN DEL SISTEMA .....</b>	<b>19</b>
3.1    La planta de Avilés.....	19
3.2    El precio de la energía .....	20
3.2.1    La interrumpibilidad.....	21
3.3    El precio del aluminio .....	21
3.4    Un sistema Multivariable.....	22

3.4.1	Proceso continuo personalizado.....	24
3.5	El control de las series de electrólisis.....	25
3.5.1	Investigación y mejora .....	26
<b>4</b>	<b>OBJETIVOS BÁSICOS INICIALES .....</b>	<b>28</b>
<b>5</b>	<b>ANÁLISIS DEL CONTEXTO Y SITUACIÓN ACTUAL.....</b>	<b>29</b>
5.1	Organización del trabajo .....	29
5.2	Herramientas de trabajo.....	30
5.2.1	Los equipos y la red .....	30
5.2.2	El trabajo con los servidores .....	31
5.3	Análisis de necesidades .....	34
5.3.1	Generación de documentación .....	34
5.3.2	Filtrado de datos .....	36
5.3.3	Apoyo gráfico en reuniones y conferencias .....	37
5.3.4	Abandono del método de trabajo por hojas Excel.....	38
<b>6</b>	<b>DEFINICIÓN Y JUSTIFICACIÓN DE OBJETIVOS ESPECÍFICOS .....</b>	<b>40</b>
<b>7</b>	<b>REQUERIMIENTOS Y LIMITACIONES.....</b>	<b>41</b>
7.1	Equipos y software disponible para el desarrollo .....	41
7.2	Necesidades estructurales del producto final .....	42
<b>8</b>	<b>ALTERNATIVAS PARA EL DESARROLLO .....</b>	<b>44</b>
8.1	Windows Forms.....	46

# Índice de ilustraciones

Ilustración 1 - Reservas mundiales de bauxita .....	6
Ilustración 2 - Esquema del proceso Bayer .....	8
Ilustración 3 - Propiedades de los aditivos.....	13
Ilustración 4 - Cuba Soderberg .....	16
Ilustración 5 - Cuba de precocido .....	17
Ilustración 6 - Entrada de la fábrica de ALCOA de Avilés.....	19
Ilustración 7 - Esquema de jerarquía del departamento de electrólisis .....	29
Ilustración 8 - Iconos de APG y P-Screens .....	31
Ilustración 9 - Icono de Microsoft Excel.....	31

# 1 Introducción

## 1.1 Identificación del proyecto

Título: Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio

Director: Ricardo Mayo Bayón

Autor: David González Aguado

Fecha: Febrero de 2015

Empresa: ALCOA Inespal S.A. España (Avilés)

Tutores de empresa: Jose Luís García García, Leonor Otero

## 1.2 Descripción del documento

La primera parte de este documento, pretende aportar una visión de contexto necesaria sobre los procesos del aluminio y la fábrica de Alcoa Avilés, necesaria para entender el proyecto y sus motivaciones. Tras ello, se muestra la definición de los objetivos finales a partir de unos objetivos generales previos y ciertas condiciones y requerimientos razonados tras una fase de análisis de situación.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 2 El proceso del Aluminio

### 2.1 Introducción

El aluminio es el tercer elemento más común en la corteza terrestre. Con su baja densidad, alta conductividad térmica y eléctrica, bajo punto de fusión y buenas propiedades mecánicas cuando se usa aleado, este metal no ferroso goza de innumerables aplicaciones, siendo sus usos más frecuentes: el transporte aéreo, la conducción de corriente, la construcción y la fabricación de envases.

La producción industrial del aluminio comenzó a principios de la década de 1900 gracias a las patentes de los procesos Bayer y Hall Héroult, que permitieron su fabricación a un precio razonable. Durante todo el siglo, la industria de producción de aluminio creció rápidamente, sustituyendo a otros metales más extendidos. En la actualidad, se generan al año unas 50, 000,000 toneladas de aluminio en todo el mundo.

### 2.2 La producción del aluminio

#### 2.2.1 Minería

El aluminio se extrae de la bauxita, un tipo de roca sedimentaria con altos contenidos en hidróxidos de aluminio. Existen otras menas de aluminio alternativas, sin embargo sus menores concentraciones no las hacen rentables.

La bauxita, cuyo nombre proviene de Les Baux, ciudad en la que se identificó por primera vez, tiene un color rojizo y una textura variable (más dura o más blanda). Su extracción se realiza normalmente en minas a cielo abierto.

La formación de depósitos de bauxita ha ido aumentando progresivamente a lo largo del tiempo, concentrándose sobre todo en los periodos Devónico, Carbonífero Inferior-Medio, Cretácico y Terciario Superior-Cuaternario. Los yacimientos de bauxita del mundo se distribuyen de forma desigual a lo largo del globo, especialmente en las zonas del trópico. Esta heterogeneidad se puede observar en la siguiente tabla:

País	Producción minera		Reservas	Base de reservas
	2010	2011 (est.)		
 Guinea	17,400	18,000	7,400,000	8,600,000
 Australia	68,400	67,000	6,200,000	7,900,000
 Vietnam	80	80	2,100,000	5,400,000
 Jamaica	8,540	10,200	2,000,000	2,500,000
 Brasil	28,100	31,000	3,600,000	2,500,000
 Guyana	1,760	2,000	850,000	900,000
 India	18,000	20,000	900,000	1,400,000
 China	44,000	46,000	830,000	2,300,000
 Grecia	2,100	2,100	600,000	650,000
 Irán	—	500 <sup>7</sup>	—	—
 Surinam	4,000	5,000	580,000	600,000
 Sierra Leona	1,090	1,700	180,000	?
 Kazajistán	5,310	5,400	160,000	450,000
 Venezuela	2,500	4,500	320,000	350,000
 Rusia	5,480	5,800	200,000	250,000
 Estados Unidos	30 <sup>8</sup>	N/A	20,000	40,000
Otros países	2,630	2,600	3,300,000	3,800,000
Total mundo (redondeado)	209,000	220,000	29,000,000	38,000,000

**Ilustración 1 - Reservas mundiales de bauxita**



### 2.2.2 El proceso Bayer

Una vez extraído, el mineral debe ser refinado para producir alúmina, u óxido de aluminio ( $\text{Al}_2\text{O}_3$ ) mediante el Proceso Bayer.

El Proceso Bayer, patentado por Karl Bayer en el año 1889, constituye hoy en día el principal método para la producción de alúmina a partir de bauxita, y basa su funcionamiento en su disolución con hidróxido sódico a alta temperatura.

Previamente, la alúmina era obtenida por el procedimiento de Le Chatelier (1869). Aunque este método generaba una alúmina con bastantes impurezas, pasó bastante tiempo antes de que fuera sustituido completamente por el Proceso Bayer. Durante al menos 40 años, ambos convivieron a precios de fabricación similares, pero con el tiempo, la mejora de la tecnología Bayer se impuso.

El Proceso Bayer comprende tres fases diferenciadas en las que se realizan distintas reacciones químicas hasta obtener el producto final: Digestión, precipitación y calcinación. Todo ello se realiza siguiendo un procedimiento con múltiples operaciones que se describe en el siguiente diagrama:

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

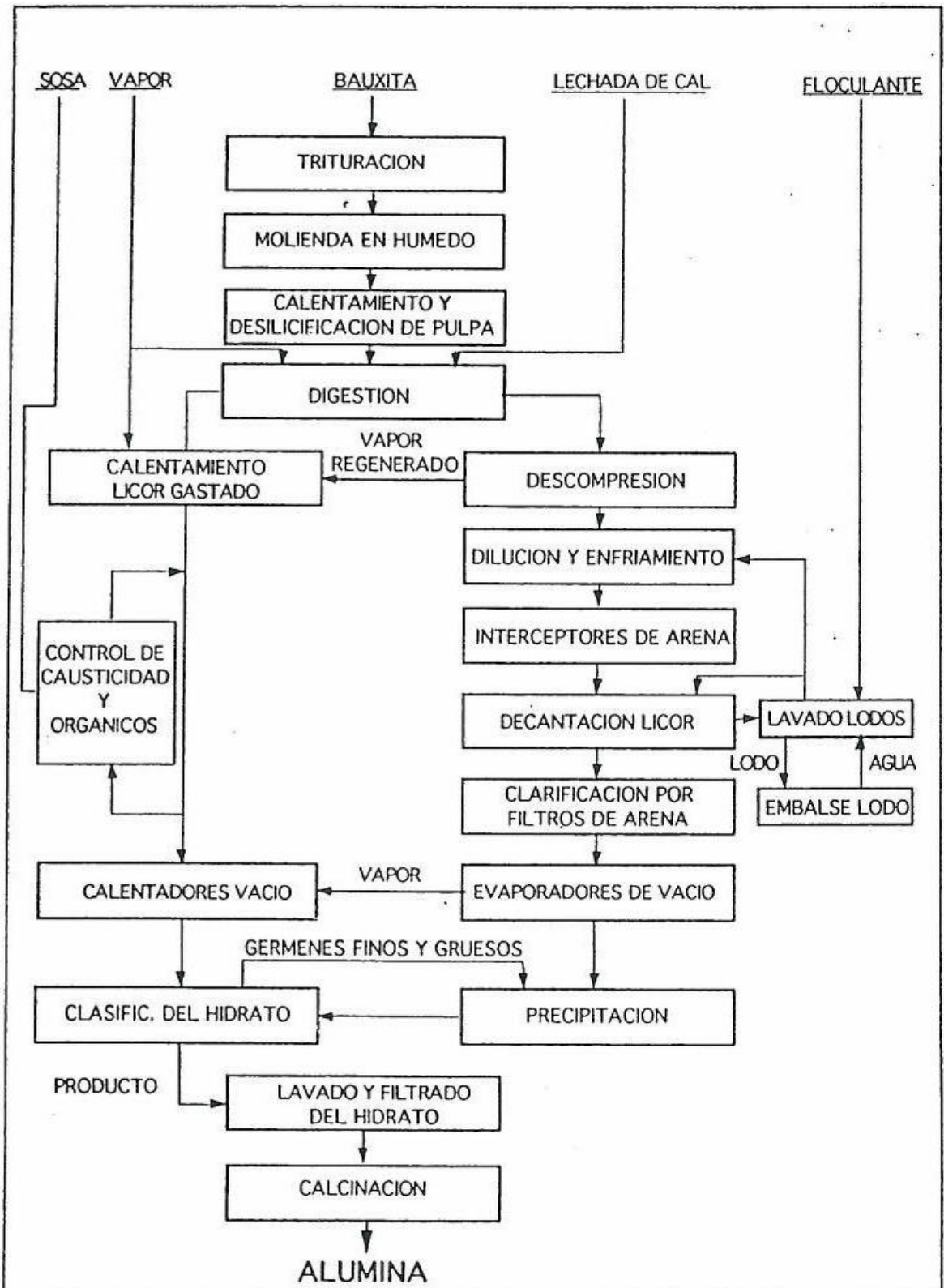
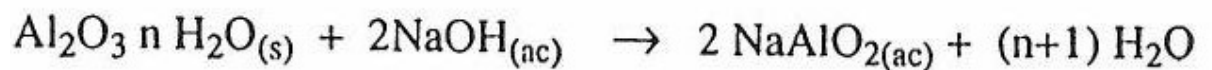


Ilustración 2 - Esquema del proceso Bayer

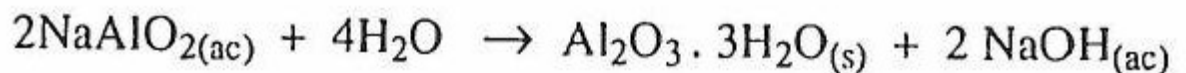
## Digestión

Es la primera fase, y corresponde a la digestión de la bauxita molida en una solución de sosa concentrada. Así, gran parte de las impurezas se retienen como sólidos mientras el aluminio permanece. Las impurezas de hierro son un grave problema para la calidad final del aluminio primario, y una de sus mayores fuentes es la contaminación de la alúmina, por lo que una digestión eficiente mejora sustancialmente el resultado.



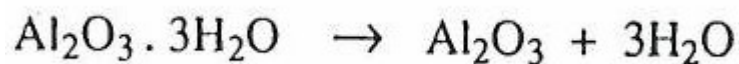
## Precipitación

Mediante el enfriamiento y la dilución de la mezcla, el hidróxido de aluminio se precipita para poder ser separado.



## Calcinación

Es la parte final del proceso. En ella, el trihidrato de aluminio se seca y se calcina obteniéndose finalmente la alúmina.



### 2.2.2.1 Calidad de la alúmina

La calidad de la alúmina está influenciada por una gran cantidad de factores a lo largo de todas las fases, y resulta necesario garantizar que es lo mejor posible. Una buena alúmina debe tener una densidad y granulometría adecuada y no contener exceso de finos. Además, debe incluir pocas impurezas, llevar un grado adecuado de calcinación y estar completamente seca. Aunque todo esto está relacionado con la buena operación de la planta, el factor más determinante en la calidad es con mucha diferencia la bauxita utilizada. Como es de esperar, no todas las bauxitas son igual de buenas, y no todas pueden dar lugar a la misma calidad de alúmina.

### 2.2.3 El proceso Hall Heroult de electrólisis

El proceso Hall Heroult es el método por el cual la alúmina es transformada en aluminio primario. Dado el objeto de este proyecto, constituye la parte más importante a destacar.

Éste método, que toma su nombre de las dos personas que patentaron simultáneamente la técnica (C.M. Hall en EEUU y P.L.T. Héroult en Francia) en el año 1886, revolucionó la producción de aluminio de forma integral. Hasta entonces, únicamente se había producido en pequeñas cantidades a precios muy elevados, pero esta nueva posibilidad rebajó su coste de producción hasta hacer posible su fabricación en masa. Junto con el Proceso Bayer, supuso la explosión de la fabricación industrial del aluminio en el mundo.

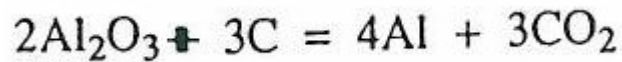
Hoy en día, al igual que ocurre con el Proceso Bayer, no se espera que la sustitución del método Hall Heroult por otro quede cercana.

#### 2.2.3.1 Fundamentos

La obtención de aluminio a partir de alúmina mediante el método de Hall Heroult, se consigue por electrólisis en un baño de sales fundidas.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

La alúmina se disuelve en un electrolito principalmente formado por criolita ( $\text{Na}_3\text{AlF}_6$ ) a alta temperatura, que también suele contener otros aditivos. Entonces, se hace pasar una corriente continua entre dos electrodos, haciendo reaccionar la alúmina con el carbono del que está compuesto el ánodo dándose lugar la descomposición del óxido de aluminio de la siguiente manera:



A medida que la alúmina es consumida, es necesario realizar más aporte de materia prima. Si la concentración de alúmina disuelta en el baño es demasiado baja, el proceso de electrólisis se interrumpirá, haciendo que la energía que ya no se emplea en la reacción química, se utilice en aumentar la temperatura. Esto es lo que se conoce como embalaje, y resulta muy perjudicial para la integridad de la cuba.

Una vez extraído de la cuba el aluminio líquido, se pasa a los hornos de fundición, donde se empleará para hacer lingotes, tochos, placas, u otro producto.

### 2.2.3.2 El electrolito

Las razones para la utilización de un electrolito de este tipo son múltiples. En primer lugar, la imposibilidad de realizar una clásica electrólisis acuosa debido a la gran afinidad del aluminio con el oxígeno, obliga a una electrólisis ígnea. Además, el proceso requiere una serie de características que el electrolito debe reunir:

- Buena solubilidad de la alúmina: Mientras mejor se disuelva, mayor facilidad de reacción y menor residuo sólido.
- Temperatura de solidificación baja: Mientras más baja se pueda mantener la temperatura de la reacción, menos energía se empleará en calentar durante el proceso.

- Menor densidad que el aluminio: Una vez que el aluminio se genere, la diferencia de densidad debe ser suficiente para que se deposite por debajo del electrolito, separándose del ánodo carbónico que se sitúa en la parte superior.
- Buena conductividad térmica: Para permitir una temperatura homogénea
- Baja presión de vapor
- Poca solubilidad del aluminio
- Buena estabilidad termodinámica
- Económico

De entre todos los posibles electrolitos, la criolita es el que mejor se acerca a estas condiciones.

### **Aditivos**

Pese a que la opción clara para el electrolito es el uso de criolita, por sí sola no es suficientemente buena y la adición de otros compuestos puede mejorar significativamente sus propiedades. En especial se busca una bajada en la temperatura de solidificación, aunque mejoras en otros aspectos también son necesarias.

Los diferentes elementos que se pueden añadir a la mezcla no tienen únicamente beneficios, y si bien pueden mejorar ciertas características, también pueden empeorar otras. El equilibrio de la química del baño es ciertamente complicado y objeto de numerosos estudios.

En la siguiente tabla se comparan los efectos de los aditivos en el baño:

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

ADITIVO UTILIZADO	SOLUBILIDAD ALUMINA	CONDUCTIVIDAD ELECTRICA	DENSIDAD	VISCOSIDAD
CaF <sub>2</sub>	D	D	A	A
AlF <sub>3</sub>	D	D	D	D
LiF	D	A	D	D
MgF <sub>2</sub>	D	D	A	A ?
NaCl	D	A	D	D
NaF	D	A	A	A
Al <sub>2</sub> O <sub>3</sub>	A	A	D	D
ADITIVO UTILIZADO	TEMPERATURA LIQUIDUS	SOLUBILIDAD DEL ALUMINIO	TENSION SUPERFICIAL	PRESION DE VAPOR
CaF <sub>2</sub>	D	D	A	D
AlF <sub>3</sub>	D	D	D ?	A
LiF	D	D	A	D
MgF <sub>2</sub>	D	D ?	A	D
NaCl	D	D ?	D	?
NaF	A	A	A	D
Al <sub>2</sub> O <sub>3</sub>	I	A	D	A

A: Aumenta    D: Disminuye    I: Indiferente    ?: Escaso nivel de confianza

### Ilustración 3 - Propiedades de los aditivos

#### 2.2.3.3 El ánodo

El ánodo es una parte fundamental en la reacción de electrólisis. Está compuesto por una pasta carbonosa de brea y coque, y aloja en su interior agujas de hierro conectadas a la corriente. Cuando la electrólisis tiene lugar, el carbono de su superficie se consume dando lugar a otro producto; CO<sub>2</sub>.

#### 2.2.3.4 El cátodo

El cátodo es la parte más importante de una cuba de electrólisis. Alojando en su interior las barras de corriente, y al mismo tiempo sirviendo de contenedor para el metal y el baño, resulta el componente más crítico para la vida útil de la cuba. Para mantener su integridad en unas condiciones tan extremas sin fisuras que hagan pasar el metal hacia las barras, el cátodo está formado por la combinación de bloques carbonosos en el interior, y ladrillos refractarios para reducir las pérdidas de energía en el exterior. Es el elemento más caro de una cuba de electrólisis.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado

Fecha: Febrero 2015

## 2.3 Las tecnologías de electrólisis de aluminio

La cuba de electrólisis, tiene una estructura similar en todas sus variantes tecnológicas. Básicamente, se puede considerar una cuba de electrólisis como un gran cajón contenedor. En su parte inferior se encuentra el cátodo, y en su parte superior el ánodo. Entre ellos pasa la corriente atravesando el contenido de la cuba. El baño electrolítico se encuentra en la parte superior, y el aluminio baja al fondo gracias a su mayor densidad. Mediante esta configuración, se logra que cuando la reacción de electrólisis se produce bajo la superficie del ánodo, el metal generado abandone la zona rápidamente por gravedad evitando que pueda reaccionar de nuevo con el oxígeno, regresando a su anterior estado. Una vez al día, cuando hay suficiente metal acumulado, se puede extraer absorbiendo con una máquina especial.

Otros elementos adicionales también son necesarios:

- Captación de gases: equipada con quemadores para no liberar CO y con filtros y recuperadores para el flúor.
- Alimentadores de alúmina: Antiguamente, para alimentar de alúmina las cubas, se picaba lateralmente la costra que cubría el baño y se vertía dentro gran cantidad cada x horas o tras un embalaje. Actualmente, de forma automática se pica puntualmente y se añade la cantidad justa cada poco tiempo.
- Coraza: Cubre el ánodo y ayuda al tapado de la cuba para evitar pérdidas de calor.

Configuración de las cubas en la serie:

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



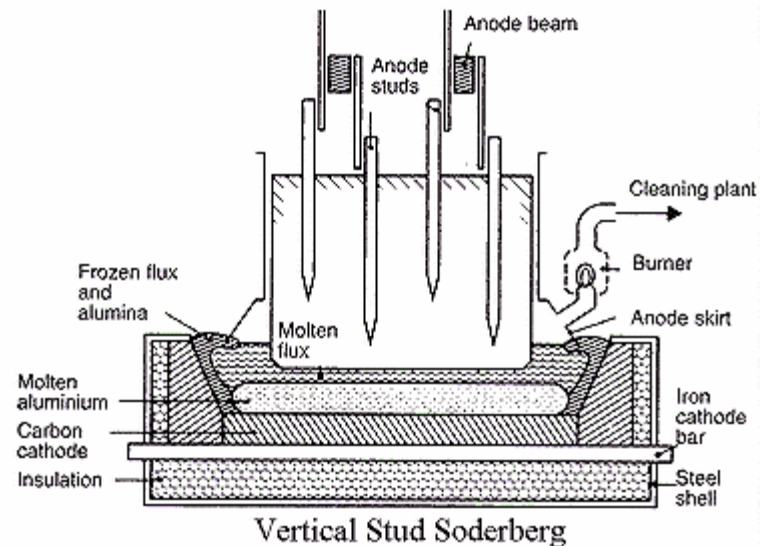
Se llama serie a un grupo de cubas hermanas dentro de la misma fábrica. Su nombre proviene de su configuración, en la que cada cuba está conectada con la siguiente. La intensidad de la serie es muy alta (del orden de 130000A), sin embargo, la diferencia de potencial entre los polos de una cuba es de tan solo unos pocos voltios.

Hecha esta visión general, se pueden distinguir dos tecnologías básicas:

### **2.3.1 Cubas Soderberg**

La tecnología de tipo Soderberg es la más antigua de las dos. En ella, un único ánodo pende encima de la cuba, y a medida que se consume la parte inferior, se rellena por la superior con más pasta de ánodos, que se cuece con la temperatura. Lo más importante para que el paso de la corriente a través del ánodo tenga un buen rendimiento, es que a través de cada aguja interior del ánodo pase la misma corriente. Así se distribuye mejor y no sufre más una zona concreta de forma que se puedan producir defectos. Esto se consigue mediante el correcto apriete de las agujas.

El aporte de alúmina en este tipo de cubas se hace o bien por picado lateral, o bien por picado puntual. El picado puntual consigue un rendimiento mucho mejor de la cuba, cebando solo la cantidad necesaria de alúmina en cada momento. Además supone una gran mejora medioambiental.



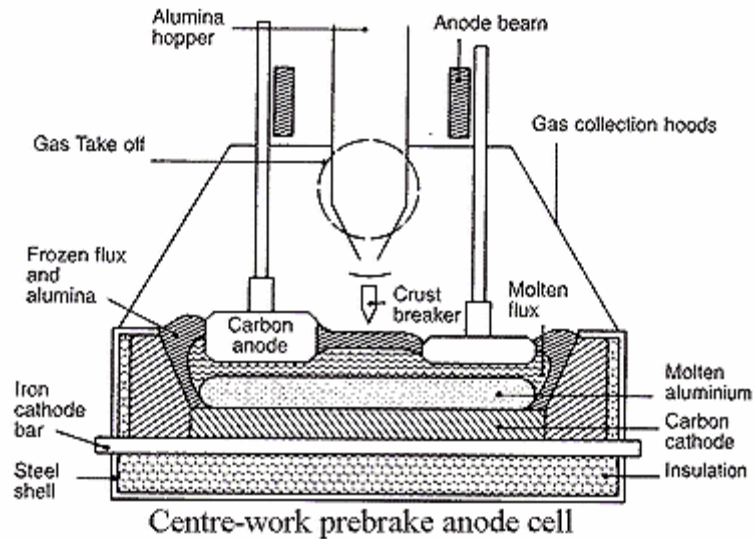
**Ilustración 4 - Cuba Soderberg**

Una desventaja clara, es que el ánodo es insustituible hasta la muerte de la cuba. Por mucho que se degrade, al haber un único y gran ánodo, no se puede levantar sin interrumpir el paso de la corriente en toda la serie, y dejando descubierta la cuba, exponiéndola a una rápida pérdida de calor y a una muy probable muerte. Debido a esto nunca se hace, y si se estropea, es necesario tratar de repararlo en funcionamiento.

Como ventaja está la facilidad con la que el ánodo se alimenta, ya que basta con añadir la pasta y ella misma se solidifica en poco tiempo. No obstante, la fabricación de un ánodo nuevo resulta una tarea bastante laboriosa.

### 2.3.2 Cubas de ánodo precocido

En las cubas de ánodo precocido, cada cuba dispone de varios ánodos individuales en vez de uno grande. Éstos se colocan uno al lado de otro formando una matriz. Gracias a ello, el equilibrado de corriente se realiza mucho más fácil, levantando o bajando cada ánodo hasta lograrlo. Cuando un ánodo se gasta demasiado, se levanta y se sustituye por otro nuevo.



**Ilustración 5 - Cuba de precocido**

Como ventajas se pueden enumerar las siguientes respecto a la tecnología Soderberg:

- Mejor consumo específico de energía
- Reparto anódico de corriente fácil y automático
- Sustitución sencilla de ánodos estropeados o gastados
- Al estar formado el ánodo por varios bloques pequeños, la alimentación es posible realizarla superiormente por la zona central de la celda, lo que garantiza un mejor mezclado de la materia prima.

Desventajas:

- Se necesita un horno de precocido para fabricar los ánodos

- El horno de precocido tiene un alto coste eléctrico
- Pasar de Soderberg a precocido supone una gran inversión

## 3 Descripción del Sistema

### 3.1 La planta de Avilés

El contexto de este trabajo de fin de máster nos sitúa en la planta industrial e ALCOA Inespal S.A. Avilés (España), una fábrica de aluminio primario en la que se produce lingote y tocho a partir de alúmina.



**Ilustración 6 - Entrada de la fábrica de ALCOA de Avilés**

La fábrica de Avilés, con capacidad para 94000 toneladas/año, produce actualmente unas 60000. Su proceso de fabricación no incluye el proceso Bayer, de modo que su materia prima inicial no es la bauxita, sino alúmina procedente de la planta de San Ciprián. Además, también recibe chatarra para reciclaje. Sus instalaciones se componen de dos series de electrólisis, una fundición, una subestación eléctrica, talleres y oficinas. En total trabajan unas 600 personas aproximadamente incluyendo subcontratas.

Desde su inauguración en 1959, la planta opera con tecnología Soderberg y por el momento no se espera ninguna inversión futura para pasar a tecnología de precocido. Desde su inicio sí ha cambiado el método de alimentación de alúmina, pasando de picado lateral a picado puntual en una modernización de la fábrica que mejoró substancialmente el método de trabajo.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Los principales clientes a los que sirve ALCOA desde esta fábrica son la industria de la construcción y la de la automoción, siendo para esta última de gran importancia la pureza del producto. El lingote; de una calidad superior al tocho, se paga a un precio considerablemente superior, pero su producción requiere una operación más perfecta en las series de electrólisis.

### 3.2 El precio de la energía

La electrólisis es un proceso químico que requiere un gran consumo de energía y, por tanto, las fábricas de aluminio son grandes consumidores de electricidad. Alrededor del 60% de los costes de operación de la fábrica de Avilés son debidos a la factura eléctrica. En la actualidad, entre 100 y 120 megavatios son necesarios para alimentar las dos series con energía suficiente para su operación dada la tasa de producción. Teniendo en cuenta el impacto de la energía en la rentabilidad de la fábrica, fácilmente se llega a la conclusión de que éste es un factor clave a optimizar. Una reducción porcentual mínima del consumo, o del precio de la energía se traduce en un ahorro millonario.

El gasto de energía derivado de la operación de cada cuba, depende en gran medida de su estado. Generalmente, una cuba en buenas condiciones requiere significativamente menos voltaje para su funcionamiento. Por otro lado, operar con excesivamente poco voltaje las cubas, puede derivar en otros problemas que a medio plazo pueden provocar pérdidas de rendimiento. La gestión y optimización de la energía en las cubas es un tema comprometido y complejo, pero muy importante en el coste total por tonelada.

Actualmente, la fábrica de Avilés afronta un grave problema debido al incremento constante durante los últimos años de la factura eléctrica en España, que hace peligrar cada día más la viabilidad de la fábrica. A esto se suma la política actual de fijar los precios año a año. Sin un contrato que garantice un precio estable a medio-largo plazo, no es posible realizar inversiones, teniendo en cuenta que en fábricas de este tipo, implican una vida útil y un retorno de inversión a varios años.

### 3.2.1 La interrumpibilidad

Una de las características de las fábricas de aluminio, es su estabilidad en el consumo energético. Al tratarse de un proceso continuo, éste se mantiene constante las 24 horas del día durante todo el año. Ésta cualidad, sumada al gran calibre del mismo, convierten a las fábricas de aluminio en un consumidor excelente para la empresa suministradora.

La situación ideal para la producción de aluminio es que la corriente no se interrumpa en ningún momento. No obstante, si esto ocurre el sistema puede mantenerse durante un tiempo sin sufrir en exceso, y esa es una ventaja que se puede aprovechar. Varias veces al año, durante picos de consumo eléctrico en España, la red puede verse sin suministro suficiente para abastecer a todos los consumidores. Aumentar la potencia generada en las centrales eléctricas no es instantáneo, y durante ese tiempo de transición alguien se tiene que quedar sin suministro. La interrumpibilidad es un servicio que se ofrece a grandes consumidores de energía que consiste en una rebaja del precio de la factura a cambio de la posibilidad de cortarles el suministro sin previo aviso durante un tiempo. Así, se compensa el déficit de energía de la red.

## 3.3 El precio del aluminio

El precio del aluminio lo marca el London Metal Exchange (LME) día a día en función de la oferta y la demanda en el mercado mundial. Su influencia es determinante en la rentabilidad de las empresas productoras ya que implica la cantidad de dinero que ingresa por sus ventas.

Éste valor puede fluctuar significativamente. Hace unos años, la crisis del precio del aluminio hizo caer el precio a unos 1000\$/Tonelada. Actualmente se sitúa ligeramente inferior a los 2000\$/Tonelada y se prevé que se siga incrementando poco a poco durante los siguientes años.

No es un mal momento en el precio del aluminio, y las ganancias han aumentado considerablemente respecto a años anteriores, pero la crisis del precio de la electricidad de ALCOA en España pone en un punto crítico el futuro de la fábrica de Avilés.

### 3.4 Un sistema Multivariable

Operar eficientemente una fábrica de aluminio no es tarea fácil. En un mundo cada vez más globalizado, la competencia de precios obliga a reducir los costes de producción al mínimo y eliminar cualquier fuente de gasto no imprescindible. A su vez la diferenciación del producto por medio de la mejora de la calidad juega un papel decisivo en un mercado en el que las empresas de alta tecnología requieren materiales cada vez mejores.

La consecución de estos objetivos puede lograrse únicamente mediante una operación impecable de las instalaciones. Si el proceso se mantiene en unas condiciones óptimas, los beneficios son innumerables:

- Maximización de la producción
- Menor gasto energético
- Reducción del deterioro de los equipos
- Descenso del número de averías y de los costes de mantenimiento
- Producción más estable y planificable
- Reducción de tiempo de trabajo perdido en solucionar imprevistos, utilizable en desarrollar medidas de mejora a largo plazo
- Mejor calidad de producto

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



- Operación con mayor seguridad para las personas
- Reducción de la contaminación

Se puede imaginar que mantener un control óptimo de las instalaciones en una fábrica de aluminio no debe ser extremadamente complicado. Sin embargo, esto no es así. Pese a que el proceso de la electrólisis tiene ya una larga historia, y ha sido ampliamente estudiado a lo largo de los años, la complejidad va mucho más allá del conocimiento de la reacción química. Alrededor de ella, infinidad de condiciones externas afectan al comportamiento de las cubas de electrólisis:

- Problemas o cortes en el suministro eléctrico
- Envejecimiento gradual de la cuba o de alguno de sus componentes
- Cambios en la calidad de la materia prima recibida (coque, alúmina, brea, etc.)
- Estado de las agujas
- Problemas anódicos
- Pericia de los trabajadores al realizar las operaciones del día a día
- Calidad del cátodo comprado
- Número de arranques de la cuba
- Decisiones tomadas para corregir variaciones
- Historia de la cuba

- Rapidez de actuación ante cambios de comportamiento
- Política de control de cuba aplicada
- Errores humanos y accidentes
- Etc.

El conjunto de todas estas y muchas más características, determinan el estado actual de la cuba e influyen en gran medida su futuro. Mejorar cada día el conocimiento y el control sobre todos los factores posibles garantiza una operación cada vez más eficiente.

### 3.4.1 Proceso continuo personalizado

La producción de aluminio sigue un método de producción continuo, en el que múltiples variables son controladas de forma constante en un proceso ininterrumpido. Éste proceso, se encuentra replicado en cada cuba, y su actividad se desarrolla de forma relativamente independiente.

Tal como hemos visto, multitud de variables afectan de manera heterogénea a la población de cubas, creando una situación en la que cada una reúne unas características únicas en cada momento. Esto motiva una situación en la que cada una de ellas, aunque operadas según el mismo patrón, es tratada como un individuo y llevada de forma personalizada a su situación.

Para entender este fenómeno se suele recurrir al paralelismo que se observa con el caso doctor-pacientes. En términos generales, todos los pacientes (cubas) son personas y la manera de proporcionarles cuidados y mantenerlos sanos responde a unas pautas médicas comunes que el doctor (ingenieros) conoce y sigue para tratarlos. No obstante, de forma más específica los pacientes son muy diferentes entre sí, y un trato personalizado es necesario. El médico tiene muchos datos concretos como la edad, la historia clínica y la condición física de cada paciente, que utilizará para proporcionar un tratamiento mejor en cada caso.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.5 El control de las series de electrólisis

El control del estado de las series, permite que se mantenga la estabilidad en la operación de la fábrica. Una monitorización del proceso persigue la detección de situaciones fuera de lo normal, y facilita la anticipación a los problemas.

Existen ciertas variables que tienen una relación directa con el buen funcionamiento de la fábrica. Gran parte de ellas se deducen de la misma naturaleza del proceso, otras se determinan empíricamente, y otras muchas permanecen desconocidas. La determinación de las variables a monitorizar es una parte clave de todo el sistema de control, ya que constituye la primera fase a llevar a cabo, e influye en todos los resultados aguas abajo.

La elección de las variables de control no debe ser tomada a la ligera. Ignorar la existencia de una variable fuertemente ligada al estado del proceso desemboca en un alto grado de incertidumbre sobre el proceso, no obstante, igual de peligroso puede ser tomar por buena una variable sin relación. La toma de decisiones en base a la misma puede motivar acciones perjudiciales para el proceso sin justificación. Además, su vigilancia y análisis consumirá un valioso tiempo en una tarea inútil.

Por supuesto, de entre todas las variables ligadas al estado del proceso, no todas tienen la misma importancia. Incluso dentro del mismo modelo de negocio, es posible que diferentes fábricas vean como lo que en una otorga una visión clara de algo, en otra no resulta tan útil. Es pues necesario también otorgar un grado de influencia o importancia que ayude a interpretar correctamente el significado de la variable.

Una vez hecho esto, el siguiente paso es definir el valor ideal de cada variable de control; aquel en el cual sabemos que se obtiene el máximo beneficio, o en el que decidimos que debemos situarnos. Además, también es el momento de decidir la tolerancia. ¿Hasta qué punto una variación se considera normal? ¿Cuándo hay que empezar a corregir? Estas decisiones resultan más fáciles de tomar desde la experiencia y el conocimiento del sistema.

Mantener las variables en sus tamaños óptimos de forma automática constituye un reto no en todo caso posible. La heterogeneidad del grupo de cubas, así como la aparición de excepciones y la congruencia de las variables entre si limitan enormemente las posibilidades de control autónomo seguro.

En la fábrica de Avilés, se dispone de un amplio número de variables a controlar. Desde su puesta en marcha, ésta colección se ha ido ampliando, depurando y perfeccionando, adaptándose a las peculiaridades de su caso concreto para ejercer un control más preciso en el proceso. Además, con el paso de los años, la capacidad para registrar datos y analizarlos también ha mejorado, haciendo que hoy en día, la base de datos de electrólisis, almacene al día una enorme cantidad de información de utilidad. Monitorizar toda esa información y decidir las acciones pertinentes a tomar, así como mejorar la manera de hacerlo constituye parte del trabajo de cada día.

### 3.5.1 Investigación y mejora

Visto que hay infinidad de posibles elementos que pueden interferir en la operación de una cuba de electrólisis, una de las cuestiones más importante es cual o cuales están detrás de cada caso.

Las pequeñas variaciones de los parámetros de control no tienen una causa concreta que buscar y corresponden a un funcionamiento normal de la cuba. Mientras se mantengan en cierto rango, no requerirán ninguna medida de investigación. Cuando sea necesario serán corregidas directamente mediante operaciones ya designadas para ese objetivo.

Cuando una variable se sale de los límites de lo que se considera normal, es lógico achacarlo a una situación excepcional, y buscar la causa raíz. Para ello, un primer paso es comprobar si algo ha cambiado en la forma de operar en los últimos días; si responde a una decisión tomada hace poco. Si no es el caso, es el momento de buscar más allá.

Delimitar el tipo de fuente del problema es un buen primer paso para esclarecer la situación. Para ello, se comprueba si el problema afecta a solo una cuba, a todas, o a un grupo.

- Si es un problema generalizado, hay que buscar la causa entre los factores que afecten a la totalidad de las cubas (Por ejemplo, las materias primas).
- Si afecta a un grupo de cubas se debe buscar los puntos en común, aquello en lo que se parecen las cubas afectadas y las diferencia del resto.
- Si es un problema aislado la causa seguramente será algo muy concreto. Se pueden eliminar un montón de posibilidades que de haber ocurrido habrían afectado a mayor población de cubas.

Aunque sea posible combatir las consecuencias del problema sin falta de encontrar la causa, hacerlo nos permite poner medios para que no se vuelva a repetir, o para minimizar las consecuencias la próxima vez. Sin ello, difícilmente se podrá mejorar a largo plazo.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 4 Objetivos básicos iniciales

La descripción del proyecto del proyecto fin de master ofertado para las prácticas por ALCOA, reúne los objetivos generales que se persiguen con el mismo, así como una breve explicación de las necesidades que lo motivan. A continuación se recoge ese texto, que constituirá el punto de partida para todo el posterior trabajo.

### **Descripción:**

En las series de Electrólisis se dispone de gran cantidad de datos para un número elevado de variables. Producción analiza dichos datos “manualmente” para tomar decisiones sobre el proceso.

El proyecto consistirá en implementar una herramienta que permita automatizar el proceso de análisis que actualmente se hace manualmente. Para ello se desarrollará una aplicación que se integre con las herramientas software de apoyo a la gestión de la producción, la cual aglutinará los algoritmos que actualmente usa el departamento de producción.

Los datos a analizar serán extraídos del ordenador de proceso y los resultados obtenidos se representarán mediante gráficas en pantalla e informes en papel.

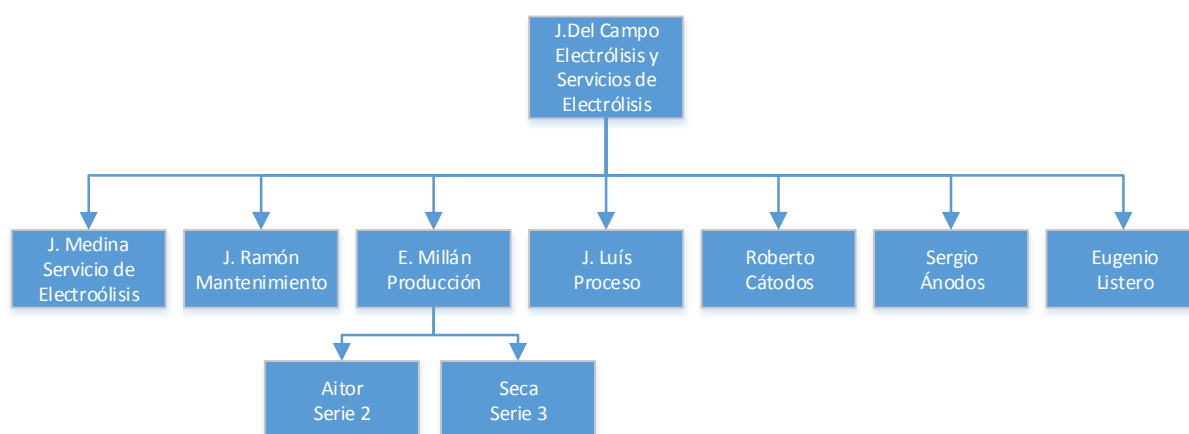
Como puede apreciarse, las indicaciones resultan bastante claras y hacen evidente el objetivo principal, sin embargo, carecen de otros datos más específicos que permitan acotar perfectamente el alcance y las prioridades. Para definir unos objetivos más específicos y verificables a la consecución del proyecto, es necesario realizar una serie de análisis previos y recabar información de las necesidades del cliente (ingenieros del departamento de electrólisis) y de los medios y herramientas disponibles.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 5 Análisis del contexto y situación actual

### 5.1 Organización del trabajo

El departamento de electrólisis se compone de varias áreas con distintas funciones. Todas ellas se encuentran al mismo nivel de jerarquía bajo el mando superior del jefe de electrólisis. Hay un responsable de cada área, y bajo cada uno, la estructura se ramifica.



**Ilustración 7 - Esquema de jerarquía del departamento de electrólisis**

Los responsables de las diferentes secciones de electrólisis trabajan juntos en una oficina común, ya que gran parte de su trabajo está relacionado. De esta forma, pueden intercambiar información fácilmente en cualquier momento. Esto resulta de gran importancia, puesto que pese a que cada uno de los ingenieros de la oficina tiene sus asignaciones, los temas importantes se discuten y deciden en conjunto (previsiones, planes de acción, cambios de modo de operación, etc.).

Cada día, se realiza una reunión diaria en la sala de juntas llamada DMS. En ella se tratan todos los eventos relevantes de la jornada, y se revisa en conjunto la evolución de la serie. También se discuten aquellos temas importantes que requieran la participación del conjunto de ingenieros. Incluso cuando no todos son necesarios, es importante hacerlo durante las reuniones para permitir que la información fluya también hacia los compañeros. En ocasiones, si se requiere, otras personas aparte de los participantes habituales se unen a la reunión.

## **5.2 Herramientas de trabajo**

### **5.2.1 Los equipos y la red**

Cada persona en la oficina dispone de un ordenador de trabajo con Windows 7 para realizar sus tareas. Todos ellos se encuentran conectados en red. Por cuestiones de seguridad, solo pueden ser instalados ciertos programas configurados en una lista. Además, ciertas páginas web están restringidas, y los archivos ejecutables tienen ciertas restricciones. El modo administrador se encuentra desactivado. Todas estas medidas, hacen de los ordenadores de trabajo y de la red, un sitio más seguro, y sobre todo más controlado y centralizado. Cada usuario dispone de un nombre y contraseña (la cual se cambia periódicamente) que le proporciona acceso a diferentes herramientas y carpetas compartidas de la red según su puesto.

Para mantener las comunicaciones entre empleados, se utiliza correo electrónico por medio de Microsoft Outlook. La cuenta de correo es la misma que la cuenta de sesión de Windows, y gracias a una tabla de direcciones es posible encontrar y enviar correos fácilmente a cualquier empleado de ALCOA, nacional o internacional.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



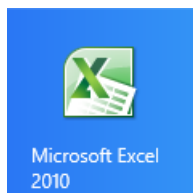
## 5.2.2 El trabajo con los servidores

La información de las series de electrólisis registrada, junto con el resto de información procedente de otras áreas de la planta, se almacena en dos servidores: El AS-400 y el QLC. El AS-400 es el servidor más antiguo, mientras que el QLC se introdujo posteriormente. El QLC no sustituye al AS-400 por completo, y pese a que comparten gran cantidad de funciones y almacenan muchos datos de forma duplicada, ambos aportan características únicas, por lo que operan de forma complementaria. Para aprovechar toda la información que recogen, la red de ordenadores se encuentra conectada a ellos, y mediante distintas herramientas, se pueden extraer los datos requeridos.



**Ilustración 8 - Iconos de APG y P-Screens**

La forma más directa y cómoda de analizar los datos de la serie es mediante las herramientas proporcionadas por ALCOA para tal efecto. A través de un escritorio remoto, es posible acceder a una estación de trabajo desde la que operar el APG Client 2010, y el TrendX Screen Explorer. Estas dos herramientas, conectadas con la base de datos, proporcionan una organización para todos los datos registrados, permitiendo ver y descargar tablas y gráficos tanto de cubas individuales, como de las series completas. Además, dispone de pantallas de operador asociadas a tareas y puestos, proporcionando una forma de modificar las consignas y realizar acciones remotamente.



**Ilustración 9 - Icono de Microsoft Excel**

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Aunque el APG y el TrendX son herramientas muy útiles y completas, la labor de los ingenieros a lo largo de los años ha ido requiriendo tareas a las que no alcanza la versatilidad de las mismas. Nuevas variables derivadas, análisis complejos de conjuntos de cubas filtradas, generación de informes personalizados, etc. Éstas son algunas de las razones por las que gran parte del trabajo que se realiza en las oficinas de ALCOA requiere una herramienta más. Microsoft Excel no es el programa idóneo para la mayor parte de los cometidos, sin embargo, igualmente puede ser conectado a la base de datos, y goza de otras ventajas que lo han hecho imponerse para un gran número de labores:

- Facilidad de manejo, gracias a la cual los archivos compartidos pueden ser entendidos y manejados por un gran número de personas.
- De uso extendido, por lo que la mayoría de gente ya tiene un contacto previo con él.
- Incluido en Microsoft Office, por lo que la licencia de uso ya está pagada y no incurre en ningún gasto extra ni requiere elevar peticiones de software a la dirección. Además, por defecto ya se encuentra instalado en todos los ordenadores de trabajo.
- Alta versatilidad para multitud de cometidos, con funciones útiles para el manejo de datos.
- Permite la programación en Visual Basic de macros para cálculos automáticos.
- No exige inversión adicional y elimina la necesidad de elevar una petición de software nueva.

Debido a estas razones, la aceptación en el departamento del uso de este programa ha sido total, y numerosos archivos de Excel se han preparado para la generación de todo tipo de informes, gráficas, tablas de datos, hojas diarias, etc. Incluso se ha llegado a diseñar un pequeño programa para introducción de datos de colada.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Por desgracia, esta tendencia no intencionada ha tenido ciertas consecuencias negativas:

- La creación de tantos archivos de Excel para tareas específicas ha generado una considerable colección desorganizada y difícil de abordar y entender por alguien no familiarizado con ella. Resulta difícil encontrar algo si se desconoce lo que buscar específicamente.
- Resulta ciertamente ineficiente en muchos aspectos del análisis de datos por su organización en celdas de los mismos.
- Los mismos archivos que almacenan los datos extraídos y derivados son los que los han calculado, lo que expone la funcionalidad de los cálculos a modificaciones de un usuario.
- Suelen almacenarse multitud de copias en diferentes equipos, provocando una multiplicación del espacio de almacenaje ocupado.
- Los archivos de históricos ocupan cada vez más, haciéndose realmente grandes.
- La compartición de información está sujeta a intercambio de archivos actualizados, por lo que es fácil perder trazabilidad.
- La masiva utilización de Excel limita aún más la posibilidad de comenzar a utilizar otras herramientas. Cada vez aumenta más la dependencia.

### 5.3 Análisis de necesidades

El departamento de electrólisis tiene sin duda muchas necesidades, siendo la más importante más personal. Desde hace tiempo cada vez son menos los puestos y mayores las atribuciones de cada uno para con el funcionamiento de la fábrica. La consecuencia de todo esto, es un menor tiempo para dedicar a tareas de análisis y mejora a largo plazo, lo que motiva la necesidad de que el mayor número posible de tareas se realicen de forma rápida y eficiente, y reduzcan el tiempo invertido por el escaso personal.

Para centrar los objetivos más prioritarios a intentar mejorar mediante este proyecto, se analizará qué actividades suponen un excesivo o innecesario trabajo.

#### 5.3.1 Generación de documentación

El aumento de la documentación en las empresas es una tendencia que se acentúa año tras año. Para ejercer labores de dirección fundamentadas en hechos, es necesario disponer de información objetiva sobre las operaciones que se realizan. Esa información se recoge en documentación que debe ser generada por los trabajadores correspondientes. Los beneficios del enriquecimiento progresivo de la misma son evidentes, pero conlleva un aumento del tiempo dedicado a rellenarla, lo que repercute directamente en la eficiencia de los puestos de trabajo.

Teóricamente, para compensar el esfuerzo extra de documentar más y más datos, se deben idear paralelamente nuevas formas de hacerlo más fácil y rápidamente. En la práctica, por desgracia el avance de ambos frentes no siempre es simétrico.

En ALCOA, la cantidad de información que se eleva desde el departamento de electrólisis es ingente. Las previsiones mensuales, trimestrales y anuales se suman a los reportes de fin de mes, trimestre y año correspondientes, a los informes cruzados entre fábricas y a toda la documentación interna. Cada vez más datos son incluidos en estos trasvases de información y hacen que el mes discurra entre entrega y entrega.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Una solución ideal partiría de la creación de un sistema automático (o al menos en gran parte) de generación de documentación estándar tanto para dentro de la fábrica como para con los niveles superiores. Esto permitiría liberar de una gran carga de trabajo a los ingenieros de la planta, permitiéndoles centrarse en investigar posibles acciones de mejora a partir de los datos, y otras labores propias de su puesto.

Paradójicamente, la recopilación de datos y su documentación, que debería facilitar y enriquecer las labores de los ingenieros, son las que lastran su avance debido al tiempo que requieren.

La creación de un sistema de documentación automática para ALCOA en Avilés presenta grandes impedimentos que hacen creer que la posibilidad pertenece a un futuro aún muy lejano. Éstas son algunas de las razones:

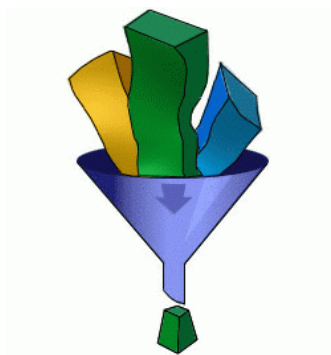
- Gran parte de los datos se generan de forma artesanal, o se retocan para excluir en su cálculo excepciones y acciones puntuales sobre el proceso que distorsionen los resultados haciéndolos inútiles. No hay manera de establecer unos baremos constantes para ello suficientes para realizar una automatización del sistema, por lo que a mayor número de datos, mayor tiempo invertido.
- El estándar de documentación cambia demasiado frecuentemente, y normalmente no lo hace de forma conclusiva, es decir, durante un tiempo hay que mantener ambos tipos de documentación.
- Se trata de un proyecto de gran envergadura que además requeriría cierto mantenimiento y actualizaciones y modificaciones regulares. La fábrica de Avilés no cuenta con un departamento informático que permita su desarrollo ni garantice una sostenibilidad a largo plazo.

- Un proyecto así requeriría una inversión que probablemente no se aceptaría. La política actual de las grandes multinacionales, y entre ellas ALCOA, aboga por la estandarización entre fábricas, y estas medidas solo podrían tomarse desde un proyecto mucho más grande que englobe varias de ellas proveniente de la central.
- Aunque se cambie el estándar de documentación, hay que seguir fabricando documentos antiguos para personas concretas que así lo exigen.

Actualmente la mayoría de documentación se rellena sobre plantillas Excel o en algún tipo de plataforma informática (unicamente aceptan introducción manual).

### 5.3.2 Filtrado de datos

Al analizar los datos de un conjunto de cubas, bien sea buscando patrones, examinando la evolución de una variable, o con cualquier otro objetivo, una dificultad que siempre emerge es la de filtrar la muestra a utilizar. Debido a que la población de cubas es ciertamente heterogénea, muchas de ellas no son comparables entre sí en multitud de aspectos debido a sus grandes diferencias. Así, en muchas ocasiones resulta necesario sacar de la muestra cubas determinadas. Ejemplos: cubas demasiado viejas, arrancadas recientemente, de una de las dos series, con agujas mucho peores/mejores... La selección de unas muestras adecuada al tipo de análisis a realizar es determinante en la fiabilidad de los resultados, y puede resultar la parte que más tiempo consume de todo el proceso de análisis.



Los motivos para la exclusión o inclusión de cubas en una muestra pueden ser de diversa índole. Unos resultan más fáciles de filtrar y se puede hacer de forma automática o semiautomática, mientras que otros, bien por ser la causa subjetiva o bien por la incapacidad o dificultad de establecer un baremo adecuado, requieren hacerlo manualmente. Por lo general el filtrado termina haciéndose casi íntegramente manera manual, ya que el número de cubas lo permite, sin embargo, el número de cubas también es suficientemente alto como para en ocasiones suponer una tarea de gran calibre. En un Excel, que es donde se suele hacer esto, el trasiego de datos resulta además bastante pesado.

La frecuencia con la que se analizan variables de conjuntos de cubas filtrados es relativamente alta. Varias veces por semana surgen dudas que pueden ser resueltas mediante la representación en una gráfica de ciertos datos, y si no es posible generarlas con rapidez, acarrear una pérdida regular de tiempo, o el abandono de algunas comprobaciones.

La necesidad de agilizar el filtrado de datos representa ahora mismo uno de los grandes puntos a tener en cuenta. Una herramienta que facilite estas operaciones resultaría muy beneficiosa.

### **5.3.3 Apoyo gráfico en reuniones y conferencias**

Como ya se ha mencionado con anterioridad, las reuniones del departamento están a la orden del día. Junto con ellas, multitud de otras videoconferencias tienen lugar a lo largo de la semana con empleados de todo el mundo. Para entenderse mutuamente y mostrar la información a exponer de forma clara, el apoyo gráfico es fundamental. Mediante tablas y gráficas bien preparadas es posible mejorar enormemente la efectividad de las conferencias y reducir su duración. Éste es un hecho hartamente conocido por todos los participantes, que se aseguran de disponer de todo el material necesario previamente.



En la fábrica de avilés, se preparan gráficas en Microsoft Excel con éste objetivo constantemente. En muchas ocasiones, si los temas a tratar son muy recurrentes, ya se tienen preparadas plantillas para generar las gráficas. Si no es así, se dedica tiempo a crearlas desde cero. Si se dispusiera de una herramienta para acelerar el proceso se ahorraría tiempo de preparación de reuniones.

En muchas ocasiones, durante la propia reunión surgen dudas o preguntas que pueden ser resueltas fácilmente mediante una representación gráfica de datos de la que no se dispone por no haber sido prevista. Cuando esto ocurre, y las explicaciones no disponen de ese punto de apoyo y se dificulta el entendimiento de lo expuesto. A veces, es posible realizar la consulta en el APG si éste dispone de una gráfica pre programada adecuada. No obstante, no siempre esto es posible, y sería muy beneficioso contar con algún mecanismo para generar rápidamente gráficas más o menos precisas con información proveniente de la base de datos.

#### **5.3.4 Abandono del método de trabajo por hojas Excel**

Éste es sin duda el más grave problema al que ALCOA Avilés debe hacer frente. Cada una de las necesidades expuestas anteriormente, tiene que ver en parte con la problemática utilización masiva de Excel, una herramienta muy versátil, pero muy poco especializada que lastra la eficiencia de las tareas.



La intención de ALCOA como multinacional, es sin duda que no exista necesidad de utilizarlo. Mediante herramientas como el APG, el TrendX y las nuevas plataformas informáticas de subida de datos, se pretende dotar a los empleados de todos los medios necesarios para trabajar sin depender de otros programas y aplicaciones. La realidad en cambio es bien distinta, sobre todo debido a la evidente diferencia entre las diferentes fábricas a lo largo del mundo, cuyas formas de trabajar distan enormemente, siendo las necesidades informáticas globales del grupo ALCOA tremendamente amplias y heterogéneas.

Desde los centros tecnológicos de la compañía, se trabaja día a día en agregar más funcionalidades al software distribuido a las fábricas. Cada vez es mayor la potencia que ofrece, pero por el momento está lejos de cubrir todas las necesidades de cada fábrica. Mientras tanto, las funciones no incluidas en esos programas deben realizarse por otros medios, y por desgracia, la adquisición de un software especializado para cubrir esas necesidades específicas no está contemplada desde la dirección al llevar en dirección contraria a la estandarización.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 6 Definición y justificación de objetivos específicos

En vista de todo lo observado durante la primera fase de análisis de necesidades, se han obtenido varias conclusiones que permiten trazar los objetivos específicos del proyecto.

En primer lugar, como resulta lógico, es imposible tratar de resolver completamente la utilización generalizada de Excel al no poder ofrecer una alternativa de software tan completa. Simplemente coger todas las funcionalidades de cientos de hojas de cálculo e integrarlas en un programa es una labor titánica y no ofrece ventajas significativas. De hecho, el trabajo posterior en ALCOA probablemente continuaría generando contenido en Excel.

La documentación que se genera en ALCOA Avilés es demasiado cambiante como para desarrollar una aplicación que ayude a cumplimentarla. En poco tiempo, quedaría obsoleto, y sin nadie en el departamento capaz de mantenerla actualizada terminaría en desuso. Es por tanto una tarea inviable.

Por clara eliminación, el proyecto se centrará en los dos aspectos restantes en los que el departamento encuentra falta de eficiencia: El filtrado de datos y el apoyo gráfico en reuniones y conferencias. Se creará pues una aplicación de fácil manejo que sirva de apoyo a los ingenieros en sus labores diarias de análisis de datos y en la preparación y exposición de las conferencias. Ésta, permitirá el manejo y cálculo de datos provenientes de las bases de datos (QLC y AS-400), su eficiente filtrado en base a los baremos más comunes, y la representación en tablas y gráficas de los resultados.

Una primera parte, irá enfocada a facilitar y agilizar el análisis de cubas ferrosas que se realiza durante la reunión diaria.

La segunda parte, tratará de ofrecer una herramienta rápida de generación de gráficas a partir de datos de cubas filtradas para análisis preliminares, comprobaciones y asistencia instantánea en reuniones.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 7 Requerimientos y limitaciones

La consecución de este proyecto de fin de master, viene limitado por una serie de restricciones impuestas por el contexto en el que se desarrolla. Identificarlas con precisión desde un principio resulta fundamental para evitar descubrirlas más adelante habiendo trabajado en vano. Además, nos ayudarán a acotar mejor la planificación y estructuración, a prevenir problemas, y en definitiva, a realizar un mejor trabajo.

Las limitaciones pueden provenir de diferentes ámbitos y tener muy distinta importancia. Muchas de ellas podrán ser salvables con mayor o menor dificultad, mientras que otras, exigirán sacrificios en algún ámbito. En cualquier caso, no pueden ser ignoradas, y de una forma u otra es necesario lidiar con ellas.

### 7.1 Equipos y software disponible para el desarrollo

ALCOA, como la gran multinacional que es, dispone de muchos medios para realizar trabajos por todo el mundo. La empresa, no solo invierte en maquinaria para la fabricación de aluminio en sus fábricas, sino también en laboratorios de investigación, oficinas, etc. En definitiva, no es difícil suponer que una parte de ese dinero, se invierta en la informática, tanto para mantener los sistemas, como para generar nuevo software. Aun así, la localización de éstos medios, no es homogénea, y la planta de Avilés, únicamente es una fábrica, y no un centro de desarrollo de software, por lo que las herramientas de las que dispone para tal efecto son muy limitadas.

Únicamente un ordenador en toda la planta cuenta simultáneamente con acceso a las bases de datos y un entorno de desarrollo capaz de crear una aplicación de escritorio como la que se pretende. Dicho ordenador, situado en la sala de control de la fábrica, emplea un sistema operativo Windows Server 2003 y dispone de una licencia de Visual Studio 2005 Profesional. El equipo, que cuenta con privilegios elevados para el control de los servidores, se encuentra conectado a la red interna de ordenadores de la fábrica, aunque por seguridad, no cuenta con acceso a internet.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 7.2 Necesidades estructurales del producto final

Existen una serie de necesidades del producto menos relacionadas con las funcionalidades que aporta, pero de mayor importancia para su éxito. Se trata de aquellas que no se incluyen en la solicitud del cliente y sin embargo son imprescindibles para su aceptación.

### **Compatibilidad:**

Todos los equipos informáticos utilizados por los ingenieros del departamento de electrólisis emplean sistemas operativos Windows 7. Por lo tanto, la aplicación debe ser capaz de funcionar en ellos sin problemas. Asegurarse de ello es prioritario; sin compatibilidad el trabajo no sirve de nada.

### **Portabilidad:**

Las restricciones sobre la instalación de programas, apertura de ejecutables y utilización del modo administrador sobre los trabajadores de la planta está restringida a una lista fija por motivos de seguridad. Es por ello que no resulta fácil el despliegue de una nueva aplicación en los equipos. Sería muy conveniente lograr que el programa no encontrara impedimentos en la seguridad de ALCOA, o al menos que éstos tengan un impacto menor con el fin de facilitar la rápida entrada del producto en el día a día.

Además de prescindir de instalación, no requerir permisos especiales y procurar un bajo peso, no requerir ninguna clase de 'extra' sería la situación ideal. Su utilización debería ser posible por si sola y no depender de paquetes específicos de Windows u otros archivos. De lo contrario, algunos usuarios podrían verse con dificultades no previstas.

### **Facilidad de uso:**

Cualquier herramienta, por muy buena que sea, si su utilización resulta incómoda y pesada, caerá en desuso gradualmente.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

**Sostenibilidad en el tiempo:**

Se trata éste de una condición comprometida. Debido a la falta de personal especializado en el campo de este proyecto, no se espera que pueda ser actualizado frecuentemente. Por ello, se evitará la inclusión de contenido que con seguridad quedará obsoleto pronto, centrandó la atención todo lo posible en aquellos puntos que sean más constantes.

**Sin coste:**

La fábrica de Avilés no está pasando por su mejor momento económico, por lo que el proyecto no puede suponer ningún gasto y debe realizarse únicamente con el material del que se disponga. De hecho, aunque fuera posible solicitar la compra de software (o la descarga de software con licencia gratuita), la burocracia empresarial para otorgar los permisos sería demasiado larga como para ser asumible.

## 8 Alternativas para el desarrollo

Una vez conocido, con sus ventajas e inconvenientes, el contexto en el que se desarrolla el proyecto, llega el momento de tomar las primeras decisiones y perfilar la solución.

La escritura del código requiere dos puntos clave:

- Un ordenador con un entorno de desarrollo instalado.
- Conexión con los servidores de las bases de datos.

Para poder reunir ambos durante la programación, se plantean varias alternativas a tener en consideración, de las cuales se elegirá la más conveniente:

1. **Emplear el ordenador de la sala de control utilizando el software ya instalado** (Visual Studio 2005).

Sin duda, la mejor opción a la hora de comenzar cuanto antes. Cuenta con la ventaja de encontrarse ya listo para su uso. No obstante, la falta de conexión a internet supone un gran hándicap en cuanto a agilidad de programación, puesto que la consulta de guías y foros estaría limitada.

Otras desventajas serían la antigüedad del equipo, las complicaciones y espera de solicitar una cuenta de usuario especial, y la localización; en una sala aislada del resto de empleados (dificulta el intercambio de información) y en un entorno ruidoso. Además, el contraste de la modernidad del software entre el ordenador de la sala de control y los de la oficina generan cierta incertidumbre sobre posibles problemas de compatibilidad.

Por otro lado, como punto fuerte se puede destacar una diferencia principal con respecto a los equipos de oficina: No existen restricciones de uso, y la cuenta de usuario dispondría de privilegios de administrador.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 2. Desarrollo en ordenador personal solicitando acceso a las bases de datos

Una idea cómoda que pronto surge ante las primeras dificultades de la opción 1. Si se proporciona acceso a los servidores a un ordenador externo, se podría programar cómodamente en cualquier entorno de programación que se elija y desde un equipo sin limitar.

Desgraciadamente, se trata de una opción no viable. Proporcionar acceso continuado a la red a un ordenador externo se considera una grave vulneración de la seguridad, y va contra las normas de la fábrica. Si bien en ocasiones es posible garantizar acceso de forma puntual, ese privilegio se encuentra reservado a altos cargos dentro la empresa al realizar visitas.

## 3. Instalación de un entorno de desarrollo en el ordenador de oficina

Por cuestiones económicas, se tendría que tratar de software de licencia gratuita. Aun así, tras una rápida búsqueda en internet, se descubre que no es tan fácil. Existen varios tipos de licencias gratuitas con más o menos restricciones, y no es sencillo determinar cómo se tratan desde el punto de vista de empresa.

Entre las ventajas que puede aportar esta opción, encontramos en primer lugar la de trabajar en el mismo equipo que el target. Evidentemente, poder elegir el software también tiene sus beneficios. Por contrapartida, están las restricciones del equipo.

Finalmente se descarta esta posibilidad debido a las dificultades que conlleva la solicitud de software nuevo. Aunque la petición no suponga desembolso económico es necesario elevar una solicitud a un nivel superior a la fábrica. Con la solicitud, sería necesaria una justificación, y aunque fuera aprobada, los tiempos de espera serían inasumibles.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### **Elección final:**

Tras eliminar las opciones imposibles de llevar a cabo, la única salida posible parece trabajar con el ordenador de la sala de control. Tal vez no una elección perfecta, pero válida al fin y al cabo. Se solicita una cuenta de usuario y se obtiene acceso.

Para una mejora de las condiciones de trabajo, se obtiene acceso al equipo de la sala de control desde el ordenador de oficina a través de la red por medio de un escritorio remoto. Esto beneficia enormemente el trabajo. Gracias a ello, la comunicación con los ingenieros será mucho más fluida, y el acceso a internet no requerirá cambiar de estancia.

El software de programación será forzosamente Visual Studio 2005, y se utilizará Windows Forms para crear el programa. Será la primera vez que utilice Windows Forms, y también la primera que desarrolle una aplicación de este tipo.

## **8.1 Windows Forms**

Windows Forms es una librería gráfica de clases incluida como parte de Microsoft .NET Framework que proporciona una plataforma para la creación de aplicaciones de escritorio para Windows. Incluye clases que permiten la utilización de los controles nativos de Windows, como botones, cuadros de texto, etc.

Para la programación usando esos servicios que ofrece la framework de Microsoft, es necesario emplear como lenguaje C++/CLI. Éste lenguaje, permite la utilización de una sintaxis similar a C++ para la utilización del CLR de Windows. A este efecto, incluye una serie de modificaciones que diferencian su sintaxis, gramática, palabras clave y atributos del inicial C++ en múltiples puntos.

C++/CLI es la estandarización de la renovación de managed C++, ahora obsoleto.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



**Aplicación de escritorio para  
visualización de datos de análisis del  
proceso electrolítico del aluminio**

**Planificación y Organización**

**David González Aguado**

# Índice

## Tabla de contenido

APLICACIÓN DE ESCRITORIO PARA VISUALIZACIÓN DE DATOS DE ANÁLISIS DEL PROCESO ELECTROLÍTICO DEL ALUMINIO PLANIFICACIÓN Y ORGANIZACIÓN DAVID GONZÁLEZ AGUADO .....	0
ÍNDICE .....	1
1 INTRODUCCIÓN .....	2
1.1 Identificación del Proyecto .....	2
1.2 Descripción del documento .....	2
2 PLANIFICACIÓN .....	3
3 ORGANIZACIÓN .....	6
3.1 Cuaderno de laboratorio .....	6
3.2 Pautas de programación .....	6
3.3 Salvado de datos .....	7

# 1 Introducción

## 1.1 Identificación del Proyecto

Título: Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio

Director: Ricardo Mayo Bayón

Autor: David González Aguado

Fecha: Febrero de 2015

Empresa: ALCOA Inespal S.A. España (Avilés)

Tutores de empresa: Jose Luís García García, Leonor Otero

## 1.2 Descripción del documento

Este documento expone la planificación y la organización con las que se ha llevado a cabo el proyecto. Se definen las fases de ejecución y su orden, y se explican los métodos de trabajo y pautas seguidas durante el proceso.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado
---

Fecha: Febrero 2015
---------------------

## 2 Planificación

La planificación de un proyecto de este tipo sin experiencia previa se antoja más complicada de lo habitual. Desconocer los problemas y desafíos más probables evitan la anticipación y dan lugar a una preparación más pobre. Aun así, es necesario abordar el proyecto de una forma ordenada y calculada para reducir al mínimo posible los problemas que puedan surgir.

Se identifican claramente varias fases o áreas en las que trabajar para dar lugar a la solución. Entre ellas, se encuentran relaciones y dependen unas de otras, por lo que el orden en el que se llevan a cabo no carece de importancia. Algunas de las fases; las que resultan más independientes, pueden ser llevadas de forma individual en cualquier momento. Otras requieren seguir una secuencia entre ellas, o ser llevadas en paralelo.

Puesto que los conocimientos previos de las herramientas empleadas no abarcan la totalidad de medios a utilizar, una parte importante del desarrollo del proyecto la constituyen tareas de aprendizaje y búsqueda de información.

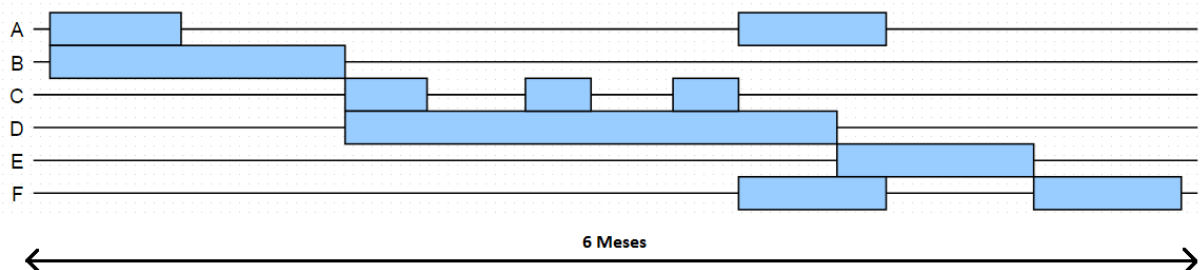
A continuación se describen brevemente las diferentes áreas de trabajo en las que se puede dividir el proyecto:

- A. **Compatibilidad:** En ella se engloba toda la búsqueda de información para saber si es posible alcanzarla, así como el trabajo posterior para conseguirla.
- B. **Aprendizaje y pruebas:** Comprende la investigación sobre el funcionamiento del proceso del aluminio, los sistemas de Alcoa, Visual Studio, Windows forms y .Net con el sistema operativo, así como todas las pruebas prácticas realizadas aparte del proyecto.
- C. **Diseño de página:** En ella se decide la información a mostrar mediante el programa en base a su utilidad, y la manera de presentarla para que cumpla su cometido lo mejor posible.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

- D. **Programación:** Se trata de la fase más grande del conjunto. Con un nombre tal vez algo escaso para lo que representa, abarca un gran grueso de contenido que podría subdividirse en varias partes de acuerdo a su tema. Sin embargo, de cara a la planificación, resulta apropiado reunir las en un único grupo. Extracción desde la base de datos, representación, creación de algoritmos, formato, etc., se entremezclan en armonía a lo largo de toda la fase de programación para dar lugar a un único producto.
- E. **Fase de pruebas:** Testeo del programa en busca de fallos y oportunidades de mejora apoyado por los ingenieros del departamento de electrólisis de ALCOA.
- F. **Solución de bugs:** Se nutre de la información extraída durante la fase de pruebas para dar solventa los problemas encontrados.

La organización en el tiempo de estas fases a lo largo de los meses que han durado las prácticas en ALCOA Avilés se puede observar en la siguiente gráfica orientativa. En ella, se aprecia cómo a medida que el proyecto avanza, los esfuerzos se centran en diferentes áreas.



### Ilustración 1 - Planificación

En un principio, los esfuerzos se centran en la búsqueda de información. Ésta va dirigida en dos direcciones:

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

- **Compatibilidad:** Para entender bien la forma en que funciona .Net, y saber las dificultades que implicará lograr la compatibilidad. Es necesario comprobar esto último cuanto antes para por lo menos saber si es posible.
- **Información sobre Windows Forms:** Para saber programar utilizando esta librería, se consultan guías y tutoriales con los que se hacen pruebas y ejercicios simples.

Una vez dada por finalizada esa primera etapa de aprendizaje, se comienza con las tareas de programación. Secuencialmente se diseñan y programan las tres pantallas de las que se compone el programa hasta su finalización.

Hacia la fase final, se retoma de nuevo el tema de la compatibilidad para general una primera versión que pueda ser testeada en los ordenadores de la oficina. Así, será más fácil identificar los errores, que ya han comenzado a buscarse.

Cuando se abandona ALCOA, la fase de pruebas aún no ha finalizado. Se necesita un tiempo mayor de uso para encontrar el resto de fallos y puntos de mejora mediante la utilización en el día a día del programa. Tras un tiempo prudencial, se vuelve para finalizar el trabajo.

Durante todo este proceso, se mantiene una constante comunicación con los ingenieros del departamento, que ofrecen el feedback a partir del progreso del programa.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3 Organización

A lo largo de la elaboración de un proyecto, el contenido del mismo crece y evoluciona a lo largo de varias y distintas fases. Para que este proceso sea efectivo y se mantenga vinculado en todo momento a los objetivos, debe estar marcado por una serie de pautas de trabajo y organización.

#### 3.1 Cuaderno de laboratorio

El cuaderno de laboratorio es una herramienta de apoyo para el día a día. En él, se apunta cada día la información y el trabajo de la jornada, datos relevantes, comentarios, y cualquier otra cosa que pueda ser de utilidad. Con el paso del tiempo, el registro de información permite volver atrás y realizar comprobaciones y consultas sobre el trabajo previo que de otra forma quizá se hubieran olvidado o perdido.

Para un beneficio máximo de ésta herramienta, es recomendable que los registros en el cuaderno sean regulares y mantengan un nivel de detalle suficiente.

Para este proyecto, el cuaderno de laboratorio utilizado es de formato en papel, en parte por las limitaciones de instalación de programas específicos y en parte por comodidad al trabajar en varios equipos.

#### 3.2 Pautas de programación

Seguir buenas prácticas de programación permite que el código producido sea mucho más legible y entendible, y frente a posibles modificaciones futuras, resulta de gran interés. Simplemente mediante el paso del tiempo, el propio programador puede olvidar el funcionamiento de las entrañas de su programa, pero el problema se agrava cuando ni siquiera es la misma persona la que debe modificar el código fuente.

- En primer lugar, y como punto más importante, se comentará todo el código de la forma más completa posible. Las explicaciones sobre la utilidad de los bloques de código son fundamentales.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

- Siempre que sea posible, se nombrarán las variables con palabras clave que indiquen su función. De este modo es mucho más difícil abstraerse de su propósito final.
- Se utilizarán sufijos con el nombre para indicar el tipo de variable de que se trata.

### 3.3 Salvado de datos

La pérdida de información es uno de los peligros que puede poner en peligro la conclusión en plazo de un proyecto. Reducir su riesgo sin embargo no resulta demasiado difícil.

Para garantizar una mayor seguridad del trabajo guardado, es de vital importancia realizar frecuentes copias de seguridad. Diariamente si es posible, se guardarán los archivos actualizados te forma triple:

- En el ordenador de trabajo
- En los servidores de Alcoa
- En un usb

Adicionalmente, y a medida que el programa evolucione, se salvarán también las versiones anteriores. De esta manera, si se descubre algún error importante, es posible volver atrás en el desarrollo o comparar entre versiones para detectar el problema.

Para no confundir archivos ni versiones, cada guardado tendrá siempre el mismo formato de nombre; la palabra indicativa del archivo, seguido de su número de versión.



**Aplicación de escritorio para  
visualización de datos de análisis del  
proceso electrolítico del aluminio**

**Elaboración**

**David González Aguado**

# Índice

<b>APLICACIÓN DE ESCRITORIO PARA VISUALIZACIÓN DE DATOS DE ANÁLISIS DEL PROCESO ELECTROLÍTICO DEL ALUMINIO PLANIFICACIÓN Y ORGANIZACIÓN DAVID GONZÁLEZ AGUADO .....</b>	<b>0</b>
<b>ÍNDICE.....</b>	<b>1</b>
<b>ÍNDICE DE ILUSTRACIONES.....</b>	<b>3</b>
<b>1 INTRODUCCIÓN .....</b>	<b>5</b>
1.1 Identificación del proyecto.....	5
1.2 Descripción del documento.....	5
<b>2 ÁREAS DESARROLLADAS.....</b>	<b>6</b>
2.1 Compatibilidad.....	6
2.1.1 Primeras pruebas: .....	6
2.1.2 Inventario .....	8
2.1.3 .Net Framework.....	10
2.1.4 Versiones y compatibilidad.....	12
2.1.5 SxS Error .....	13
2.1.6 SxS Trace.....	13
2.2 Adquisición de datos.....	16
2.2.1 Procedencia de la información.....	16
2.2.2 Conexión y extracción de datos .....	17
2.3 Representación de gráficas.....	20
2.4 Diseño de página y formato .....	22
<b>3 BLOQUES DEL PROGRAMA.....</b>	<b>24</b>
3.1 Menú.....	24

---

3.1.1	Gestión de ventanas.....	25
3.2	Ferrosas.....	25
3.2.1	Contexto.....	27
3.2.2	Tabla general.....	29
3.2.3	Histograma y gráfico circular.....	33
3.2.4	Tabla de últimos embalajes.....	37
3.2.5	Tabla de histórico.....	39
3.2.6	Tabla de histórico de embalajes.....	40
3.2.7	Gráfico de evolución de metales.....	42
3.3	Análisis de grupos.....	45
3.3.1	Opciones de filtrado.....	47
3.3.2	Variables.....	49
3.3.3	Algoritmo y gestión de datos.....	51
3.4	Análisis específico.....	52
3.4.1	Variables.....	54
3.4.2	Algoritmo y gestión de datos.....	56

# Índice de ilustraciones

Ilustración 1 - Ejemplo de error SxS .....	7
Ilustración 2 - Diagrama de un CLI.....	11
Ilustración 3 - Diagrama de estructura de .Net .....	11
Ilustración 4 - Detalles del error SxS .....	15
Ilustración 5 - Asistente de conexiones de datos.....	18
Ilustración 6 - Ejemplo de tabla de datos .....	19
Ilustración 7 - Logotipo de Zedgraph .....	20
Ilustración 8 - Ejemplo de gráfica de zedgraph.....	21
Ilustración 9 - Ventana de menú .....	24
Ilustración 10 - Ventana de ferrosas.....	26
Ilustración 11 - Tabla general de ventana de ferrosas.....	29
Ilustración 12 - Tabla de datos de la tabla general.....	31
Ilustración 13 - Control de fecha de página de ferrosas .....	32
Ilustración 14 - Control de baremo de hierro de página de ferrosas .....	33
Ilustración 15 - Controles de cuba extra de página de ferrosas .....	33
Ilustración 16 - Histograma del Fe de página de ferrosas.....	34
Ilustración 17 - Gráfico circular de pureza de página de ferrosas .....	35
Ilustración 18 - Tabla de datos para histograma y grafico circular de página de pureza	36
Ilustración 19 - Tabla de últimos embalajes .....	37
Ilustración 20 - Selector de cuba de página de ferrosas .....	37
Ilustración 21 - Tablas de datos de últimos embalajes de página de ferrosas .....	39
Ilustración 22 - Tabla de histórico de página de ferrosas .....	39
Ilustración 23 - Tabla de datos de histórico de página de ferrosas .....	40
Ilustración 24 - Tabla de histórico de embalajes de página de ferrosas.....	40
Ilustración 25 - Tabla de datos de histórico de embalajes de tabla de ferrosas .....	42
Ilustración 26 - Gráfico de evolución de metales de página de ferrosas .....	43

---

Ilustración 27 - Control selector de metales de página de ferrosas.....	43
Ilustración 28 - Página de análisis de grupos.....	46
Ilustración 29 - Paneles de configuración de grupos de página de grupos .....	47
Ilustración 30 - Selector de variable de página de grupos.....	49
Ilustración 31 - Página de análisis específico .....	52
Ilustración 32 - Selector de variable de página análisis específico .....	54

# 1 Introducción

## 1.1 Identificación del proyecto

Título: Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio

Director: Ricardo Mayo Bayón

Autor: David González Aguado

Fecha: Febrero de 2015

Empresa: ALCOA Inespal S.A. España (Avilés)

Tutores de empresa: Jose Luís García García, Leonor Otero

## 1.2 Descripción del documento

Este documento recoge la información acerca de la elaboración del proyecto, comprendiendo el diseño de páginas, los razonamientos de las decisiones y el funcionamiento general a nivel técnico entre otras cosas. Se encuentra dividido en dos grandes apartados: áreas desarrolladas, y bloques de programa.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 2 Áreas desarrolladas

El desarrollo de un proyecto es un proceso largo y variado en el que se tocan variedad de temas para lograr un objetivo. Éste en particular no constituye una excepción, y las áreas que de las que se compone deben ser analizadas y explicadas por separado para garantizar una mayor claridad.

Estas áreas de las que se habla, no guardan un paralelismo con las fases introducidas en la planificación, que atienden más a motivos temporales. Si bien es cierto que en algunos casos como el de la compatibilidad si se puede encontrar una correspondencia, no es la norma. En muchos otros, distintas tareas de diversa naturaleza forman parte de una misma fase del desarrollo, entrelazando sus tareas simbióticamente, sin por ello poder ser tratadas de forma similar.

Existen también áreas de trabajo que no es posible situar en el tiempo, ya que afectan directa o indirectamente al proyecto de una forma global.

Las áreas en las que se ha dividido la ejecución de éste proyecto han sido elegidas por resultar apropiadas para la completa comprensión del mismo. No obstante, la clasificación podría haber sido diferente resultando igualmente efectiva.

### 2.1 Compatibilidad

Desde el momento en el que las herramientas de trabajo quedaron fijadas, la compatibilidad ha sido un gran motivo de preocupación para la realización de este proyecto. Si no es posible hacer funcionar el programa en los ordenadores de las oficinas, todo el trabajo que se realice será en vano.

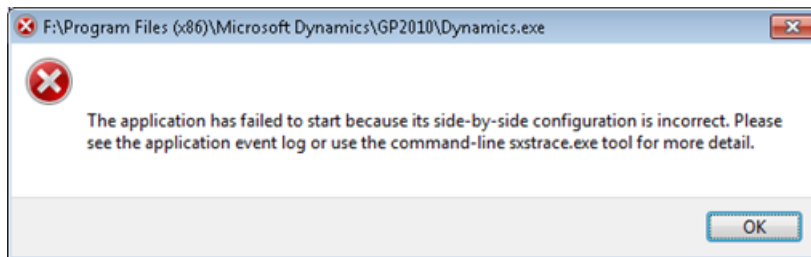
#### 2.1.1 Primeras pruebas:

Las primeras pruebas realizadas en el marco de la compatibilidad, pretenden comprobar si realmente constituye un problema, o si por el contrario, el funcionamiento es normal en cualquier máquina desde un principio.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Se crea un programa sencillo de Windows Forms similar al histórico “Hola Mundo”, y tras ser compilado y enlazado, se procede a ser portado a un ordenador de la oficina, y así verificar los resultados.

Lamentablemente, el programa no llega a ejecutarse, confirmando las sospechas de que lograr la compatibilidad no resultará tan fácil. En lugar de ello, un mensaje de error se muestra en pantalla, advirtiendo de una forma muy genérica de problemas denominados “side to side” (SxS).



#### Ilustración 1 - Ejemplo de error SxS

Puesto que la información aportada por el mensaje de error es insuficiente, y el concepto de side to side desconocido, en ese momento comienza un proceso de documentación para comprender y acotar el problema.

Tras una búsqueda rápida en google, se advierte que el concepto side to side guarda relación con .Net, y la manera en que distintos programas y aplicaciones utilizan los recursos en Windows.

Para poder analizar el problema y contrastarlo con nuestra situación concreta, se comienza realizando un inventario de los recursos software con los que cuenta cada ordenador.



## 2.1.2 Inventario

Se elaboran dos listas separadas, una para el equipo de desarrollo, y otra para el target, es decir, el que va a ejecutar la aplicación. En ellas, se recogen todos los recursos instalados que guarden relación con .Net.

### **Equipo A (Desarrollo):**

#### **SO:**

- Windows Server 2003 Service Pack 2 Standard edition 5.2.3790 build 3790

#### **.Net builds:**

- .Net Compact Framework 1.0 Service Pack 3 Developer
- .Net Compact Framework 2.0
- .Net Compact Framework 3.5
- .Net Framework 2.0 Service Pack 2
- .Net Framework 3.0 Service Pack 2
- .Net Framework 3.5 Service Pack 1
- .Net Framework 4 Client profile
- .Net Framework 4 Extended

#### **Entorno de desarrollo:**

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

- Visual Studio 2005 Profesional Edition ENU 8.0.50727.762

### **Equipo B (Target):**

#### **SO:**

- Windows 7 64bits Service Pack 1

#### **.Net builds:**

- .Net Framework 4 Client profile 4.0.30319
- .Net Framework 4 Extended 4.0.30319

Como se puede observar, el equipo A cuenta con una mayor variedad de versiones de .Net que el B, aunque por el momento; sin entender la utilidad de esa amplia gama, no es una información que aporte mucho valor.

Gracias a una consulta a la Wikipedia, se comprueba que la versión de Visual Studio de 2005 que incluye el equipo A, se apoya en la framework .Net 2.0.

Acabada esta fase de inventario, se pasa al estudio del funcionamiento de .Net, con el que se espera obtener respuestas que ayuden a solucionar el problema. Gracias a la vasta información recogida en portales como Microsoft Developer Network, Wikipedia o Stackoverflow, ésta tarea se antoja mucho más fácil, aunque en ocasiones abrumadora.

### 2.1.3 .Net Framework

Un framework, es una infraestructura digital que proporciona un entorno con una serie de herramientas genéricas y reutilizables que pueden ser utilizadas como base para el desarrollo de software. Los frameworks pueden incluir programas de apoyo, compiladores, librerías, APIs, etc. que proporcionan funcionalidades estándar de bajo nivel. Con todo ello se pretende facilitar la labor del desarrollador, que puede invertir más tiempo en la programación de las funciones específicas del proyecto.

.Net es el framework creado por Microsoft que principalmente es utilizado en Windows. Está formado básicamente por el Common Language Runtime (CLR) y la Framework Class Library (FCL). A éstos se añaden otras librerías de clase que se han ido agregando con cada nueva versión de .Net, como pueden ser WindowsForms, ADO.Net o WPF.

#### **Common Language Runtime:**

El CLR, es la implementación de Microsoft en .Net de una CLI, o infraestructura de lenguaje común (Common Language Infrastructure). Su función es la de crear una plataforma capaz de soportar el desarrollo y la ejecución de código generado en múltiples lenguajes de alto nivel, aportando además funciones de manejo de excepciones, seguridad, etc. Se trata del motor de ejecución de .Net, y sobre él se ejecutan todos los programas de ésta framework.

Los programas desarrollados para .Net, deben ser compilados a CIL; un lenguaje intermedio de bajo nivel. Durante la ejecución, ese código se traduce mediante un compilador en tiempo real a lenguaje máquina.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

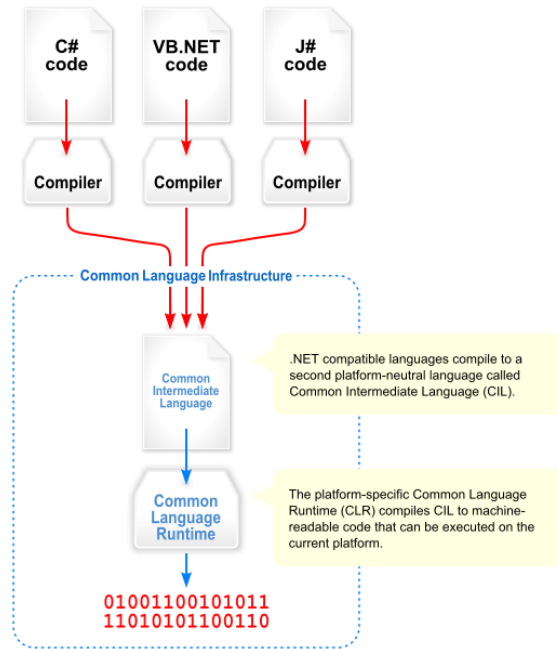


Ilustración 2 - Diagrama de un CLI

**Framework Class Library:**

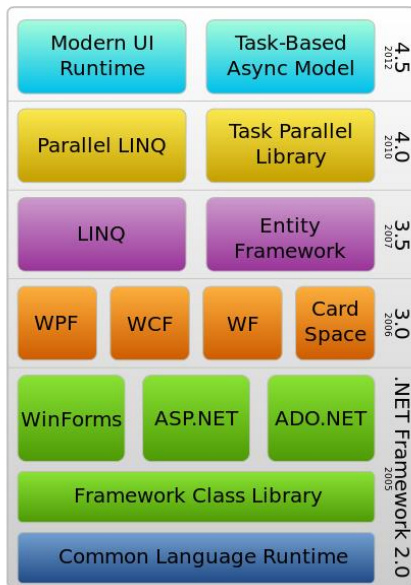


Ilustración 3 - Diagrama de estructura de .Net

<p>Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado</p>	<p>Fecha: Febrero 2015</p>
--	----------------------------

.Net incluye gran cantidad de librerías estándar que implementan funciones comunes. Su código está escrito en CIL, por lo que se encuentran disponibles para todos los lenguajes de programación admitidos por la common language infrastructure. Se pueden clasificar en dos categorías no excluyentes:

- **FCL:** Las librerías del FCL contienen las funciones más básicas e imprescindibles.
- **BCL:** Comprende las librerías del FCL y todas las demás que forman parte de .Net, como WindowsForms, WPF, ADO.Net, etc. Con cada nueva versión, ésta categoría aumenta su tamaño.

Una de las principales ventajas que ofrecen las librerías de .Net es la eficiencia que ofrecen. Almacenadas en código CIL en archivos .dll, ofrecen sus funciones a cualquier programa que las requiera e invoque. De ésta forma, no solo ayudan al desarrollo de programas, sino que reducen la duplicidad de código sirviendo como recurso de acceso común.

#### 2.1.4 Versiones y compatibilidad

Desde la versión 2.0 de .Net en la que trabaja Visual Studio 2005 hasta la 4.0 de la que dispone el equipo target, hay un salto de 3 versiones. Durante el paso de 2.0 a 3.5, pese a los cambios sufridos por el framework, la versión de CLR se ha mantenido constante. No obstante, con la llegada de .Net 4.0, éste ha sido sustituido, lo que podría indicar una posible fuente de problemas, ya que supone importantes modificaciones en el núcleo de ejecución.

Contrariamente a lo que se pensaba, según la información consultada, 4.0 es 100% compatible con el contenido generado para versiones anteriores, y solo podría existir conflicto en el caso inverso.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Como posible fuente del problema, está la posibilidad de que la opción del panel de control que autoriza la compatibilidad con .Net 3.5 (CLR 2.0) se encuentre desmarcada. La comprobación de este factor es necesaria, sin embargo requiere privilegios de administrador. Con ayuda de la administradora de equipos y redes de Alcoa, se comprueba que la casilla no estaba desactivada, y por lo tanto, no era la fuente del error.

### 2.1.5 SxS Error

Los errores Side by Side, son generados cuando existe un problema con el dll al que intenta acceder un programa. El problema más evidente sería que no existiera en el equipo, y por tanto, no sea posible encontrarlo.

Como ya se ha visto previamente, los archivos .dll son compartidos por muchos programas, y esto, en ocasiones genera otro tipo de complicaciones como sustitución de uno de ellos por una versión distinta, borrado por desinstalación de otro programa, etc. En los sistemas modernos, existe una administración automática de las versiones de librerías que reducen los problemas de este tipo. Si en el archivo del tipo manifest del programa se detalla la versión a cargar, se buscará esa específicamente, mientras que si no existe ese dato, se cargará la versión por defecto.

### 2.1.6 SxS Trace

Una vez comprendido el funcionamiento general de .Net y los conceptos básicos de Side by Side y los dll en Windows, se trata de encontrar el origen del error específico generado por la ejecución del programa. Para ello, tal como se aconseja en numerosas fuentes de internet (consultar bibliografía), se emplea la herramienta sxstrace, accesible a través de la consola de comandos de Windows (cmd).

La herramienta sxstrace, sigue el rastro de los mecanismos side by side para identificar el dll que genera conflicto, y escribe un report con los la información en un documento .txt. Para realizar estas operaciones, es necesario contar con privilegios de administrador, ya que de lo contrario, el archivo de texto no se generará, y no se mostrará ningún mensaje que indique la causa.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

---

Tras varios intentos fallidos, el posterior descubrimiento de la necesidad de iniciar como administrador, y la obtención temporal de privilegios para éste objetivo, se obtiene el documento con la información detallada del error:

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

```

=====
Begin Activation Context Generation.
Input Parameter:
  Flags = 0
  ProcessorArchitecture = wow32
  CultureFallbacks = en-US; en
  ManifestPath = C:\Users\v-gonzad1\Desktop\menu\debug\menu.exe
  AssemblyDirectory = C:\Users\v-gonzad1\Desktop\menu\debug\
  Application Config File =
-----
INFO: Parsing Manifest File C:\Users\v-gonzad1\Desktop\menu\debug\menu.exe.
INFO: Manifest Definition Identity is (null).
INFO: Reference: Microsoft.VC80.CRT,processorArchitecture="x86",publicKeyToken="1fc8b3b9a1e18e3b",type="win32",version="8.0.50727.6195"
INFO: Resolving reference Microsoft.VC80.CRT,processorArchitecture="x86",publicKeyToken="1fc8b3b9a1e18e3b",type="win32",version="8.0.50727.6195".
INFO: Resolving reference for ProcessorArchitecture WOW64.
INFO: Resolving reference for culture Neutral.
INFO: Applying Binding Policy.
INFO: No publisher policy found.
INFO: No binding policy redirect found.
INFO: Begin assembly probing.
INFO: Did not find the assembly in WinSxS.
INFO: Attempt to probe manifest at C:\Windows\assembly\GAC_32\Microsoft.VC80.CRT\
8.0.50727.6195_1fc8b3b9a1e18e3b\Microsoft.VC80.CRT.DLL.
INFO: Did not find manifest for culture Neutral.
INFO: End assembly probing.
INFO: Resolving reference for ProcessorArchitecture x86.
INFO: Resolving reference for culture Neutral.
INFO: Applying Binding Policy.
INFO: Find publisher policy at C:\Windows\winsxs\manifests\
x86-policy_8.0.microsoft.vc80.crt_1fc8b3b9a1e18e3b_8.0.50727.4940_none_516d712b0f495a45.manifest
INFO: No binding policy redirect found.
INFO: Begin assembly probing.
INFO: Did not find the assembly in WinSxS.
INFO: Attempt to probe manifest at C:\Windows\assembly\GAC_32\Microsoft.VC80.CRT\
8.0.50727.6195_1fc8b3b9a1e18e3b\Microsoft.VC80.CRT.DLL.
INFO: Attempt to probe manifest at C:\Users\v-gonzad1\Desktop\menu\debug\Microsoft.VC80.CRT.DLL.
INFO: Attempt to probe manifest at C:\Users\v-gonzad1\Desktop\menu\debug\Microsoft.VC80.CRT.MANIFEST.
INFO: Attempt to probe manifest at C:\Users\v-gonzad1\Desktop\menu\debug\Microsoft.VC80.CRT.DLL.
INFO: Attempt to probe manifest at C:\Users\v-gonzad1\Desktop\menu\debug\Microsoft.VC80.CRT.MANIFEST.
INFO: Did not find manifest for culture Neutral.
INFO: End assembly probing.
ERROR: Cannot resolve reference Microsoft.VC80.CRT,processorArchitecture="x86",publicKeyToken="1fc8b3b9a1e18e3b",type="win32",
version="8.0.50727.6195"
ERROR: Activation context generation failed.
End Activation Context Generation.

```

#### Ilustración 4 - Detalles del error SxS

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado

Fecha: Febrero 2015



Como puede observarse en la imagen, el error proviene de los archivos del assembly VC80.CRT. Éstos están incluidos en la distribución de Microsoft Visual C++ Redistributable.

La solución llegado a este punto sería asegurar que la versión correcta de los dll necesarios se encuentra disponible en todos los ordenadores en los que se pretende desplegar el programa mediante la instalación de la distribución de Visual C++ adecuada en todos ellos. Dado que la instalación de software está tan limitada en los ordenadores de ALCOA, sería preferible optar por otra alternativa. Además, requerir que el usuario instale aparte otros extras para su funcionamiento puede disuadir del uso del programa.

Finalmente, la opción más conveniente es descargar los archivos específicos necesarios en su versión adecuada e incluirlos junto con el programa en la misma carpeta, de forma que siempre se encuentren disponibles. No se trata de la opción más eficiente desde el punto de vista de desarrollo de software, pero es la más conveniente de cara al producto final.

## 2.2 Adquisición de datos

La adquisición de datos externos desde el programa permite tomar los datos de las series de electrólisis de los servidores, que supone la materia prima a partir de la cual se trabajará. Durante este capítulo se explica cómo este proceso se hace posible.

### 2.2.1 Procedencia de la información

Como ya se ha comentado anteriormente, Alcoa aloja dos grupos de servidores: el AS400 y el QLC, siendo éste último el más moderno.

Ambos servidores contienen una gran cantidad de datos de los procesos. A ellos, llega de diversas fuentes, como pueden ser PLCs, PDAs, introducción manual, etc. Mucha de ella es compartida, gracias a que el AS400 vuelca una gran parte sobre el QLC, sin embargo aún existe información solo accesible desde uno de ellos.

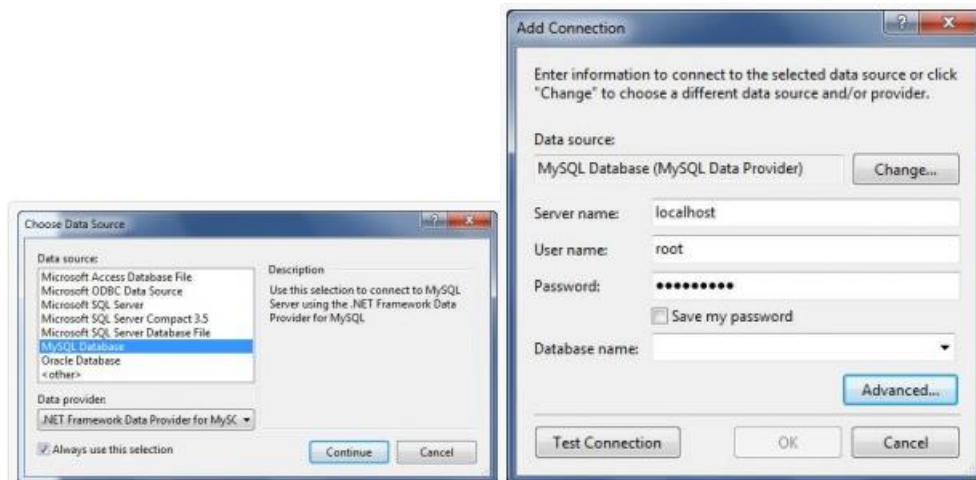
Para poder explotar toda la información disponible en la fábrica, una conexión con ambas fuentes de información se antoja la mejor decisión. No obstante, finalmente esto no fue posible.

Tras conseguir satisfactoriamente la conexión con el QLC, se encontraron grandes dificultades para detectar el AS400. Puesto que ya se habían recibido advertencias de su mayor complicación, se continuó buscando una solución durante un tiempo, hasta que se conoció el verdadero problema. Aunque los ordenadores de las oficinas gozan de acceso a ambas fuentes de datos, el equipo de la sala de control en el que se desarrolla el programa se encuentra aislado del AS400, haciendo evidentemente imposible su detección.

En vista de los acontecimientos, únicamente existirá conexión con el QLC desde el programa. Afortunadamente, contiene la inmensa mayoría de los datos que podrían resultar de interés para la aplicación.

### **2.2.2 Conexión y extracción de datos**

La conexión con la base de datos se programa de manera bastante sencilla desde Visual Studio mediante su asistente. Siguiendo los pasos indicados y marcando las opciones convenientes la conexión se configura de manera automática, autogenerándose el código para ello.

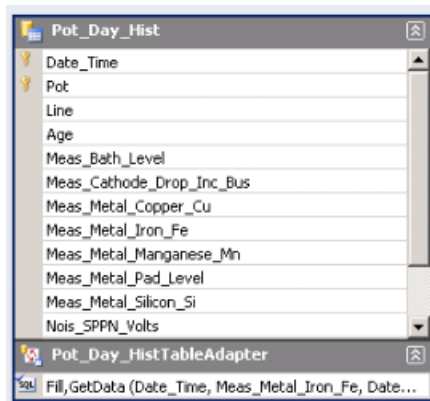


**Ilustración 5 - Asistente de conexiones de datos**

Entre las opciones, encontramos como las más importantes el tipo de fuente de datos, el usuario, y la contraseña. Estos dos últimos son fijos, es decir, no existen perfiles para usuarios sino que son los únicos datos de acceso y nunca se cambian. Por esta razón, se ha decidido que la información estará contenida en el código fuente y no se le requerirá al usuario por pantalla. Siempre será la misma, nunca se cambia, y solo incurre en una pérdida de tiempo cada vez que se acceda, sobre todo si no se recuerda y es necesario consultarla.

Una vez completada esta primera parte, queda configurar las variables a extraer y los métodos utilizados para ello.

Valiéndose de las dos herramientas que proporciona Visual Studio para ello, no resulta muy difícil seleccionar las variables requeridas de entre las disponibles en la base de datos, y configurar una tabla con ellas. Para cada tabla de datos que se quiera rellenar en la aplicación, se crea su correspondiente tabla para la adquisición de datos con las variables a extraer.



### Ilustración 6 - Ejemplo de tabla de datos

Junto con la tabla, se crea un table adapter asociado, que determina la manera en que se extraen los datos de esas variables. Mediante su modificación, se termina de establecer todo el mecanismo de extracción de datos.

Es posible agregar variables pertenecientes a otras tablas de la base de datos, marcar filtros en función de datos fijos o variables, modificar el nombre o el valor de las variables al entregarlas al programa, etc. Un gran número de posibilidades que se definen a medias entre un interfaz visual y la modificación del query en SQL para generar el comportamiento deseado en cada tabla de datos. Pueden consultarse todas las queries en el documento de código.

Para aprender a usar las herramientas de Visual Studio de conexión y extracción de datos de una fuente de datos, han servido de gran ayuda los documentos y tutoriales de los que dispone la web de Microsoft <https://msdn.microsoft.com>.

Los conocimientos necesarios de SQL se han obtenido mediante la observación del código autogenerated, las pruebas realizadas, y la lectura de diferentes hilos en los foros de <http://stackoverflow.com>.

## 2.3 Representación de gráficas

La representación de gráficas es una parte muy importante en el análisis de datos, y una herramienta muy común en casi todas las empresas. Para los objetivos que trata de cumplir este programa, son indispensables.

Hoy en día, las versiones modernas de Windows Forms, ofrecen una gran variedad de posibilidades para generar y personalizar distintos tipos de gráficas para cualquier fin. En cambio, la versión de Visual Studio de 2005 mediante la que se desarrolla este proyecto, aún no incluye dichas funciones. Es por ello que es necesaria la inclusión de librerías complementarias que suplan esa carencia.

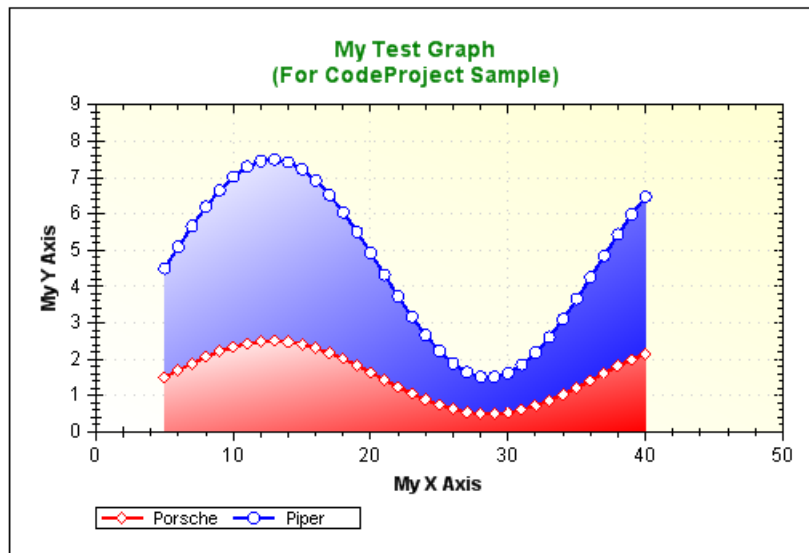
La búsqueda de librerías de gráficas para Windows Forms es una tarea complicada. La solución no solo debe adaptarse a las necesidades, sino que además debe ser específica para el CLR de .Net 1.0 o 2.0, ya que si se apoya en uno superior, podría dar lugar a errores. Esto es un problema, dado que el software antiguo es mucho más difícil de encontrar.

Tras barajar varias opciones posibles, la elección final fue ZedGraph. Zedgraph es una librería de clases que permite dibujar gráficas 2D de tipo línea, barra y tarta. Su uso es relativamente sencillo, ofrece las funcionalidades necesarias para este proyecto, y dispone de amplia documentación en su página web (<http://zedgraph.sourceforge.net/samples.html>).



**Ilustración 7 - Logotipo de Zedgraph**

Zedgraph añade el control gráfica a los ya existentes en Windows Forms. Se trata de un control altamente versátil que cuenta con numerosas opciones. En la web <http://www.codeproject.com/Articles/5431/A-flexible-charting-library-for-NET> se puede consultar un amplio tutorial para aprender a manejarlas, aunque dispone de otras muchas no nombradas en él.



**Ilustración 8 - Ejemplo de gráfica de zedgraph**

Mediante el uso de sus métodos y la modificación de sus propiedades, los gráficos se rellenan de información y varían su formato para conseguir el efecto deseado, consiguiendo unos resultados como el del ejemplo.

Algunas de las posibilidades que ofrece y han sido aprovechadas en el desarrollo del programa son las siguientes:

- Modificación de texto y formato en títulos, ejes, leyenda, etc.
- Colores de las distintas zonas del control, así como de las líneas/barras de la gráfica.

- Escalado y zoom de la zona de representación.
- Modificación de grosores, marcas y numeración en ejes y líneas.
- Agregación de múltiples series de datos como objetos modificables y personalizables.

## 2.4 Diseño de página y formato

Siempre que la aplicación de un programa sirve a la observación del usuario, una gran parte del éxito radica en el formato. La forma en que se muestran las cosas afecta sustancialmente a la efectividad con la que se perciben y al interés que genera. Por esta razón, cuidar el formato muchas veces es más importante que ofrecer un mayor detalle.

Al diseñar cada elemento y sus características en las tres ventanas del programa, se mantiene especial cuidado en multitud de objetivos relacionados con el formato, dejando al azar el menor número posible de cosas:

- En las listas desplegadas, siempre que no sean numeradas se elige el orden en el que se sitúan los elementos en función de su importancia o frecuencia en que se suele consultar.
- Se trata de no sobrecargar de texto con explicaciones, haciendo intuitiva la identificación de las funciones de todos los elementos.
- Se emplean colores para distinguir y separar por similitud, y tamaño y grosor de letra para denotar importancia.
- Las magnitudes de las variables se presentan convertidas siempre a la unidad que se maneja comúnmente en el día a día en ALCOA.

- Se impide que pueda existir duda acerca de la correlación entre las opciones marcadas y la información mostrada, bien actualizando en tiempo real, o bien mediante un mensaje en pantalla.
- El orden de las variables en las tablas se elige cuidadosamente por función y utilidad.
- Se evita sobrecargar mostrando datos no relevantes u opciones no útiles. Más completo no siempre es mejor.



### 3 Bloques del programa

En este capítulo del documento, se explica cómo se han desarrollado las diferentes partes del programa. De forma jerárquica según los bloques en que se encuentra dividido, se expondrán las razones de cada decisión tomada en el diseño del programa, y se explicará el modo en que se han llevado a cabo.

#### 3.1 Menú

El menú es el primer contacto que tiene el usuario con el programa al ejecutarse. Su función, es ser el nexo de unión entre las distintas aplicaciones que ofrece, y guiar a ellas al usuario. En un programa como éste, su diseño es bastante sencillo.

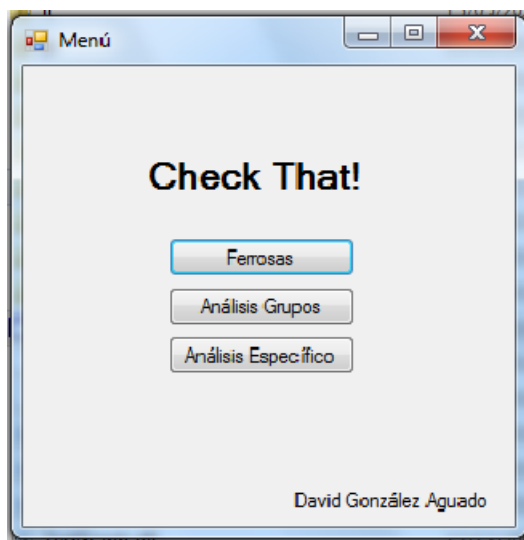


Ilustración 9 - Ventana de menú

Con un tamaño de ventana relativamente pequeño (de acuerdo a la cantidad de componentes), la ventana de menú presenta el nombre del programa en letras grandes como recibimiento. En la barra superior, señala que nos encontramos en el menú.

Tres botones se sitúan en la parte central de la ventana, justo debajo del título. Cada uno, tiene de nombre una de las tres utilidades del programa, y es el acceso a ella. Mediante la pulsación de un botón con el ratón, se activa el evento que dispara la generación de la ventana seleccionada, que queda maximizada y seleccionada automáticamente.

### 3.1.1 Gestión de ventanas

Para poder abrir simultáneamente varias ventanas distintas, el menú no se cierra al abrirse ninguna de ellas; simplemente se queda en segundo plano. Ésta decisión sin embargo, permite que abrir un infinito número de pestañas, y no solo una de cada, lo que podría resultar demasiado lioso.

Para solucionar el problema se opta por el siguiente método:

Tras inicializar la Form1 (menú), se generan las otras tres forms, que no se hacen visibles hasta la activación del evento click de su botón correspondiente. Pulsaciones sucesivas únicamente reafirmarán su visibilidad, sin crear nuevas entidades.

Adicionalmente, se modifican los eventos de cierre de las formas 2, 3 y 4 para sustituirlo por la modificación de su propiedad “visible” a cero. Así, la forma desaparece al pulsar la equis, pero realmente no se cierra, y sigue aguardando a una nueva invocación mediante los botones del menú, por lo que su estado se guarda hasta salir por completo del programa a través de la equis del menú.

## 3.2 Ferrosas

La página de cubas ferrosas tiene como objetivo informar en profundidad sobre el estado de las cubas de la serie centrándose principalmente en el hierro. Con las herramientas que en ella se incluyen, se pretende facilitar la labor del departamento de electrólisis en sus reuniones diarias de monitorización de cubas ferrosas y pureza.

La parte izquierda de la ventana, está dedicada a un análisis global de las series, mientras que la derecha, está centrada en el análisis de cubas concretas seleccionadas mediante el control de la parte superior derecha de la ventana.

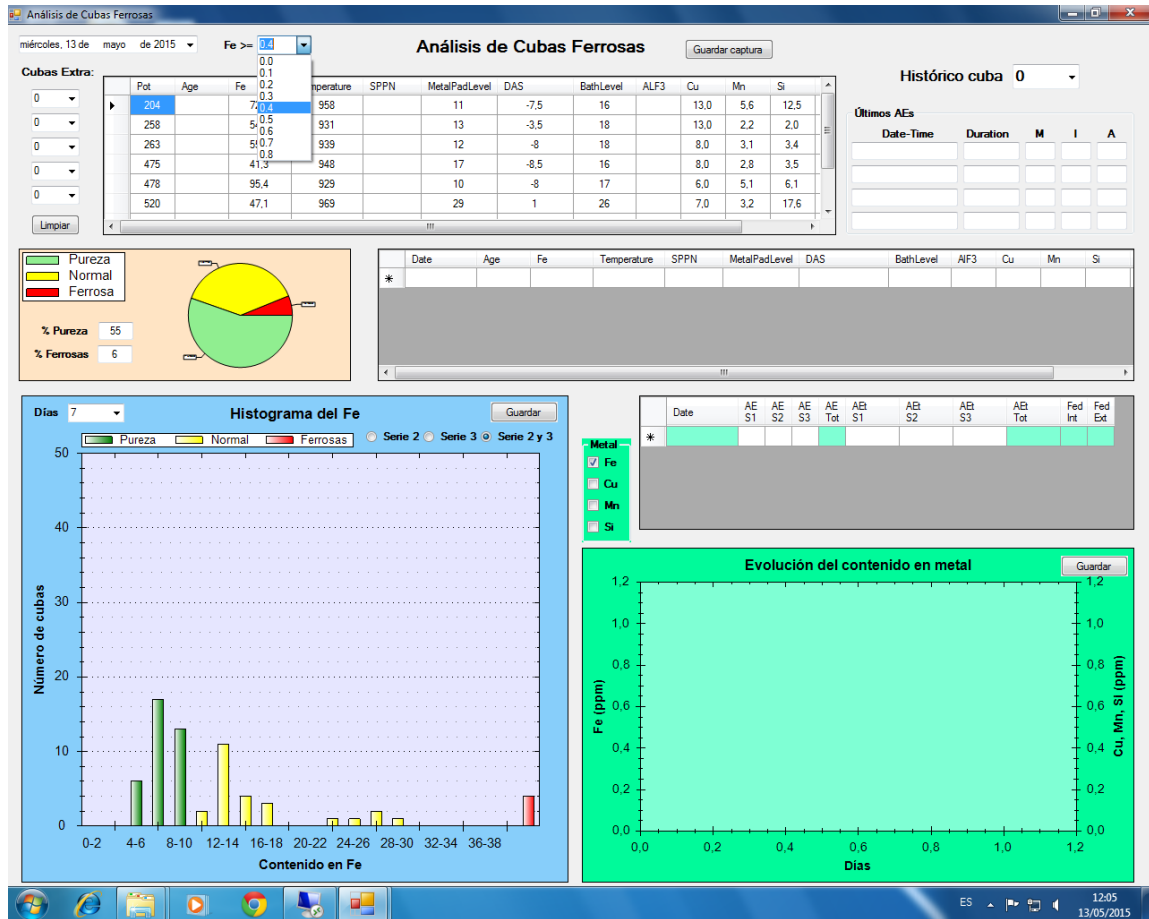


Ilustración 10 - Ventana de ferrosas

<p>Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado</p>	<p>Fecha: Febrero 2015</p>
--	----------------------------

### 3.2.1 Contexto

#### 3.2.1.1 El contenido en hierro

La importancia del contenido en hierro de una cuba de electrolisis es muy alta. En primer lugar, afecta directamente a la pureza del producto final, limitando la cantidad de aluminio de alta pureza (mayor precio) que la fábrica es capaz de generar. Además, un alto contenido puede significar un peligroso descenso en la integridad de la cuba. Por ello, mantener los valores de hierro bajo supervisión, y trabajar diariamente para reducirlos lo máximo posible es fundamental.

El contenido en hierro de una cuba viene dado por un gran número de factores que lo incrementan en mayor o menor medida.

- Materias primas: El hierro presente en alúmina y coque supone un alto porcentaje del contenido total. Afecta por igual a la totalidad de las cubas y no suele variar excesivamente con el tiempo, aunque puede ocurrir.
- Herramientas: Pérdida de material de las herramientas de hierro que los trabajadores sumergen en la cuba. Causa generalizada y poco importante en el total de hierro.
- Caída de piezas: En ocasiones, una tapa de un quemador, fragmentos de coraza u otras piezas pueden caer a la cuba. Si éstos no pueden ser rápidamente recuperados, elevarán el contenido en hierro durante una semana o dos.
- Desgaste de agujas: Las agujas del ánodo pierden material muy poco a poco y se van desgastando, pero este fenómeno puede verse altamente incrementado por un mal ajuste de agujas que las sobrecargue de forma desigual.
- Problemas anódicos: Causan desprendimientos de coque y un consumo acelerado de ánodo, disolviendo más del hierro contenido en éste.

- Barras catódicas: Las barras catódicas, se encuentran protegidas por unos ladrillos de carbono muy resistentes a la temperatura. Con el tiempo e influenciados sobre todo por la operación de la cuba, éstos se van deteriorando y se pueden agrietar o corroer. Si el metal se abre camino hasta las barras catódicas, el nivel de hierro aumenta considerablemente. Es la causa menos deseable y también la más peligrosa, ya que puede desembocar en la perforación de la cuba y su consiguiente destrucción.

Con todas estas fuentes de hierro interviniendo, y otras posiblemente desconocidas, descubrir la causa detrás de un incremento de hierro para actuar en consecuencia no siempre es fácil, pero mediante la experiencia y la observación de otras variables relacionadas, las posibilidades aumentan.

### 3.2.1.2 Cubas de pureza y cubas ferrosas

De cara al producto final, la pureza del aluminio cumple un papel fundamental en los beneficios. Mientras mayor sea la pureza global de la suma del aluminio que se extrae de todas las cubas, más aluminio de primera calidad (más caro) se podrá vender y más chatarra se podrá fundir en los hornos.

El aluminio salido de las series de electrólisis puede clasificarse en tres categorías:

- Aluminio de pureza: Su contenido en hierro es menor de 0.1. Aluminio de alta calidad que puede ser utilizado para fabricación de lingote o para compensar un aluminio peor proveniente de cubas ferrosas o de hornos de reciclado.
- Aluminio normal: Entre 0.1 y 0.4 de hierro. Utilizado para fabricación de tochos.
- Aluminio ferroso: Por encima de 0.4Fe. Debe ser mezclado con aluminio de mejor calidad para poder ser utilizado para hacer tochos.

Una alta cantidad de cubas ferrosas produce problemas para fundición, ya que su metal debe ser diluido para fabricar tochos.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

En cambio, el exceso de pureza puede generar muchas ganancias. Se pueden producir muchos más lingotes, que cuentan con una prima por calidad, y también se puede diluir con metal de los hornos de reciclaje para aumentar producción.

### 3.2.2 Tabla general

La tabla general constituye un listado detallado de las cubas con problemas de hierro del día. En él, las cubas que superen el baremo de las consideradas “ferrosas” se ordenan junto con sus datos más relevantes. Sobre ella pues, no solo se puede obtener una visión de la cantidad de cubas problemáticas en ese momento, sino también realizar un análisis preliminar de la situación de cada una y comprobar la gravedad de la situación.

Las cubas pueden ser ordenadas en función de cualquiera de las variables de mayor a menor o viceversa.

Pot	Age	Fe	Temperature	SPPN	MetalPadLevel	DAS	BathLevel	ALF3	Cu	Mn	Si
204		72,3	958		11	-7,5	16		13,0	5,6	12,5
258		54,4	931		13	-3,5	18		13,0	2,2	2,0
263		55,4	939		12	-8	18		8,0	3,1	3,4
475		41,3	948		17	-8,5	16		8,0	2,8	3,5
478		95,4	929		10	-8	17		6,0	5,1	6,1
520		47,1	969		29	1	26		7,0	3,2	17,6

Ilustración 11 - Tabla general de ventana de ferrosas

#### 3.2.2.1 Variables

Las variables contenidas en la tabla general, están meticulosamente escogidas de forma que abarquen toda la información útil para entender la situación específica de cada cuba, excluyendo cualquier otro no relevante para su análisis como cuba ferrosa.

- Pot: Número de cuba.
- Age: Edad de la cuba.

- Fe: Contenido en hierro. Es la principal variable a observar, y la que determina la pertenencia a la lista. Valores muy altos implican una mayor gravedad.
- Temperatura: Dado que la perforación de una cuba viene favorecida por las altas temperaturas, es uno de los valores más importantes de monitorizar.
- SPPN: Medida de la estabilidad de la cuba. Valores anormalmente bajos indican que la electrólisis no se está produciendo bien. Valores muy altos tampoco son deseables y señalan una situación anómala como una tensión insuficiente o un problema anódico.
- MetalPadLevel: Medida de la altura de metal. Influencia en gran medida la estabilidad y la inercia térmica.
- DAS y BathLevel: Medidas del nivel de baño de la cuba. Importante de conocer si se pretende añadir o retirar baño.
- ALF3: Contenido de fluoruro de aluminio. Influencia en gran medida la temperatura de la cuba. A mayor cantidad, menor temperatura.
- Cu: Contenido en cobre. Algunas cubas tienen incrustadas en sus paredes pletinas de cobre como testigo. Si la pared se corroe hasta llegar a ellas, el cobre se mezcla con el metal para indicar el problema mediante una medida alta de contenido en cobre.
- Mn: Contenido en manganeso. El manganeso está presente en las juntas que unen los ladrillos de carbono del cátodo.
- Si: Contenido en silicio. Los ladrillos refractarios de las paredes de la cuba contienen silicio. Si la medida aumenta, indica que se están consumiendo por falta de talud.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

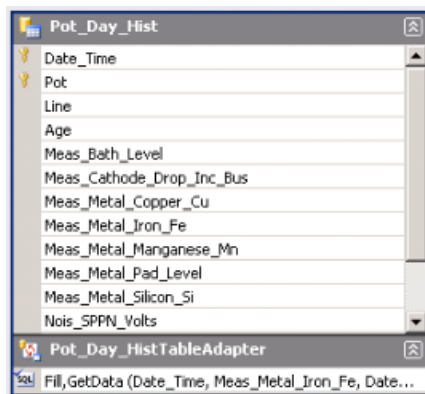
- Resistencia catódica: Medida mensual del estado del fondo de cuba. Valores más altos implican un fondo en peor estado, con más suciedad, planchones, etc.

El orden de estas variables no es azar. Éstas se disponen en función de su importancia para favorecer una rápida consulta.

Una casilla en blanco significa que no existe dato para esa variable ese día. Podría haberse presentado el último dato disponible, sin embargo no sería una buena práctica. En los casos como los de las cubas ferrosas, en los que las variables pueden cambiar muy rápidamente, presentar datos antiguos al mismo nivel que actuales puede suponer un gran error, y aún más si no se sabe cuáles de ellos lo son.

### 3.2.2.2 Funcionamiento

La tabla general toma sus datos de la librería PotDayHist del QLC. Esta librería, contiene los datos diarios históricos de una gran cantidad de variables, calculados como la media del día en caso de que existan varias entradas.



**Ilustración 12 - Tabla de datos de la tabla general**

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



Para acceder a la información deseada almacenada en la librería PotDayHist, se crea un data binding entre el control datagridview en el que se visualizarán los datos, y la tabla Pot\_Day\_Hist configurada en el editor de accesos a fuentes de datos. Mediante el query programado en SQL para esa tabla, se accede específicamente a los datos requeridos en la base de datos y éstos son cargados en el control datagridview.

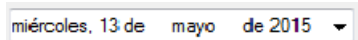
El query, configurado en el TableAdapter, es una cadena de órdenes para extraer datos a la base de datos que incluye una serie de información como: una lista de las variables requeridas con su situación correspondiente, las condiciones que filtran la totalidad de los datos para seleccionar los deseados, los modificadores que se aplican a los datos obtenidos para variar los valores o los nombres, etc. En un query, respetando la sintaxis del lenguaje SQL, debe figurar todo lo necesario para obtener exactamente y nada más que lo que se necesite.

El query de la tabla general puede ser observado en el documento de código de este proyecto.

### **Controles:**

Para poder extraer los datos de la base de datos, se necesita proporcionar una serie de valores de configuración que reflejen las necesidades del usuario. A través de su selección en los controles disponibles a tal efecto, esta información se recoge y se manda la petición de datos.

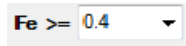
Fecha: La fecha se selecciona mediante un control datetimepicker preseleccionado por defecto al día actual. Puede seleccionarse cualquier día desde que se comenzaron a recoger datos diarios.



### **Ilustración 13 - Control de fecha de página de ferrosas**

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Baremo de hierro: Fija el contenido en hierro a partir del cual una cuba aparece representada en la gráfica. Por defecto se sitúa en 0.4 (límite típico de cubas ferrosas), pero puede modificarse para abrir o cerrar más el espectro.



**Ilustración 14 - Control de baremo de hierro de página de ferrosas**

Controles de cubas extra: Se proporcionan cinco selectores de cuba extra para añadir cubas que no cumplan la condición del hierro pero que por otros motivos (como un rápido incremento del hierro los últimos días) se consideren dignas de interés. Pueden ser vaciados simultáneamente todos ellos mediante el botón limpiar para mayor comodidad.



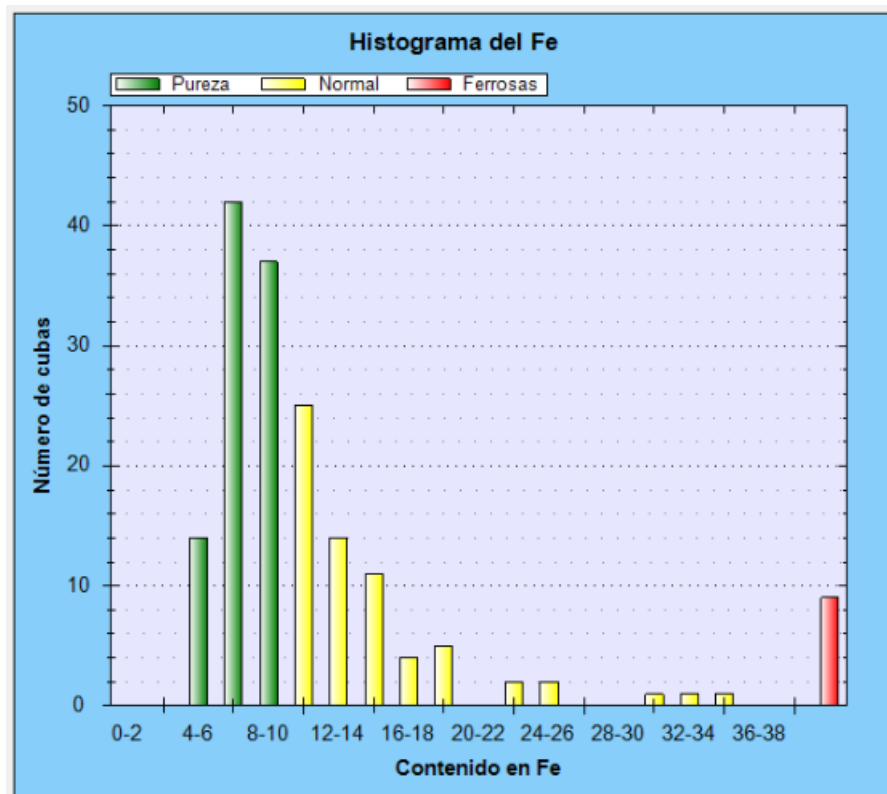
**Ilustración 15 - Controles de cuba extra de página de ferrosas**

### 3.2.3 Histograma y gráfico circular

La función de estos dos diagramas, es proporcionar una visión general del estado de la serie con respecto a las ferrosas.

Histograma:

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



**Ilustración 16 - Histograma del Fe de página de ferrosas**

El histograma del hierro, muestra en el eje x intervalos de contenido en hierro, y en el eje y el número de cubas de la serie o series que pertenecen a ese intervalo. Típicamente la distribución se asemeja a una gaussiana con algunas cubas ferrosas muy alejadas.

El intervalo final del eje x abarca todos los valores superiores, ya que es bastante común que una vez disparado el hierro, varíe mucho entre cubas y resulta más práctico e intuitivo de este modo de representación.

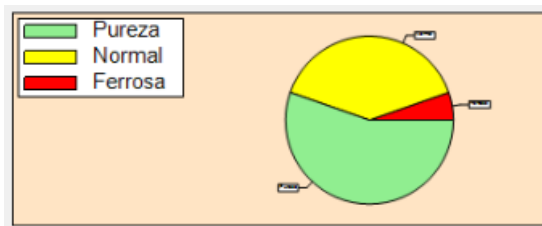
Este tipo de gráfico del hierro, ya era utilizado por los trabajadores de Alcoa anteriormente. El programa APG ofrece una versión más rudimentaria que presenta algunos problemas, pero aun así siempre resultó de bastante utilidad. Ésta versión incorpora varias mejoras y lo incluye en el entorno de las demás herramientas de análisis de ferrosas, lo que mejora sustancialmente su aplicación.

Para resaltar los límites de pureza y ferrosas, las barras se colorean en tres colores. A la parte izquierda (pureza) en verde, a la parte central amarillo, y a la parte derecha (ferrosas) rojo. De este modo, un vistazo rápido revela mucha más información.

La información aportada por esta gráfica para un solo día, no suele ser demasiado útil para, como se quiere, comprobar el estado general de la serie, dado que las fluctuaciones normales desvían aleatoriamente los datos. Una media de varios días suele ser preferible. Para ello, mediante un selector es posible elegir entre los valores actuales, o la media de la última semana (opción por defecto) o los últimos quince días para la representación.

Adicionalmente, también es posible limitar la representación a una de las dos series por separado mediante los botones conmutables de la parte superior.

Gráfico circular:

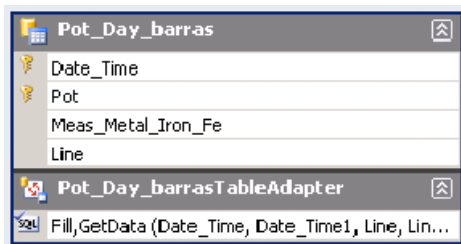


**Ilustración 17 - Gráfico circular de pureza de página de ferrosas**

Se trata de un gráfico sencillo del porcentaje de pureza y ferrosas. Sus opciones van ligadas a las del histograma y su única función es comparar la importancia de cada categoría con las demás. Además, muestra los valores numéricos exactos en dos cajas de texto.

### 3.2.3.1 Funcionamiento

La manera en que los datos pasan desde la base de datos hasta las gráficas no dista mucho de cómo lo hacen en el caso de la tabla principal. Del mismo modo, las opciones marcadas en los controles se transmiten a la petición de datos a través del query, y los datos se registran en un control datagridview, que en este caso está oculto. Por supuesto, la tabla y el tableadapter es distinto, pero funciona de igual manera.



**Ilustración 18 - Tabla de datos para histograma y grafico circular de página de pureza**

Una vez allí, se puede acceder a cada elemento por su posición en la talla, y operar para transformarlos en los datos derivados que se requieren para las gráficas. Mediante la creación de arrays de contadores y la comparación de los valores de contenido en hierro para incrementarlos, se obtienen de forma ordenada los valores y para cada punto del eje x, que se introducen en la gráfica gracias al uso de los métodos del control de zedgraph.

En los días en que no existe dato de hierro (frecuente), se utiliza el último dato disponible para esa cuba, que nunca será más antiguo de siete días.

Para modificar los aspectos visuales de la gráfica basta con cambiar las propiedades de sus elementos.

### 3.2.4 Tabla de últimos embalajes

Últimos AEs				
Date-Time	Duration	M	I	A
20/04/2015 23:36	00:03:30	24	13	15
20/04/2015 11:34	00:02:50	26	9	16
19/04/2015 21:00	00:01:50	24	9	15
19/04/2015 21:00	00:03:20	24	8	17

**Ilustración 19 - Tabla de últimos embalajes**

Se dice que se produce un embalaje, cuando se interrumpe la electrólisis, y la electricidad que fluye a través de la cuba y que antes se empleaba en ella, comienza a sobrecalentar el baño. Durante un embalaje, la resistencia y el voltaje aumentan considerablemente.

Los embalajes son unos eventos de gran interés para el análisis de cubas específicas, ya que influyen en gran medida en su integridad y varían enormemente muchas de sus variables como temperatura o estabilidad. Conocer si ha habido embalajes en los últimos días y como han sido puede ayudar a entender mejor la situación actual de una cuba.

En la tabla de los últimos embalajes, se representa la información de los cuatro últimos eventos de este tipo para la cuba seleccionada en el selector de cuba. Únicamente cuatro debido a que no se recogen más datos en la base de datos.

Histórico cuba 0 ▾

**Ilustración 20 - Selector de cuba de página de ferrosas**

Las variables que se representan son las siguientes:

- Fecha y hora del embalaje

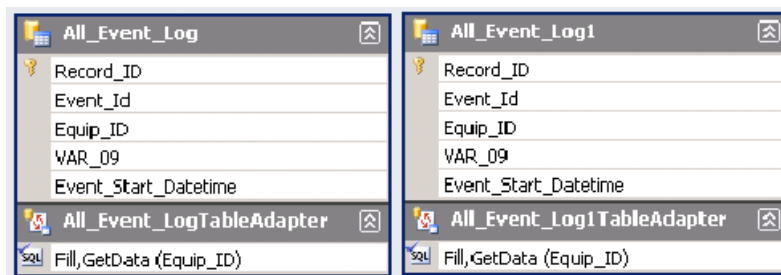
Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

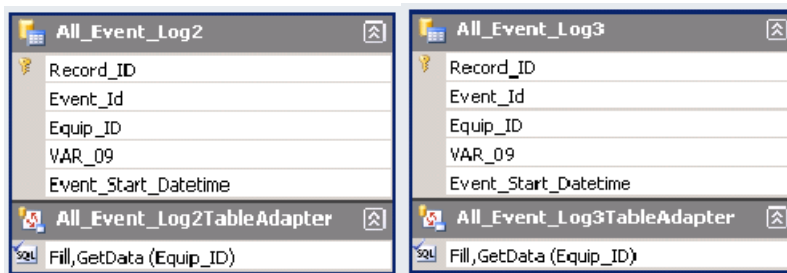
- Duración del embalaje: mientras mayor duración, más agresivo habrá sido el embalaje.
- Fuerzas máxima, inicial y media del embalaje: Por lo general, las cubas en buen estado presentan embalajes con mayor fuerza.

La principal ventaja que supone para los ingenieros de Alcoa esta tabla, es que hasta este momento no disponían de una manera tan rápida de acceder a esta información. Para ello era necesario consultar una lista de últimos embalajes ordenada por orden cronológico que incluía datos de todas las cubas. Ahora es posible seleccionar una y obtener directamente los datos buscados.

### 3.2.4.1 Funcionamiento

Como se puede comprobar, pese a que se nombra como tabla de últimos embalajes, ésta no está construida sobre un control del tipo datagridview, sino que es un conjunto de cajas de texto separadas. La razón de esto es la nefasta organización de estos datos en el QLC, que impide extraer la información mediante un único query de forma sencilla. Para sobreponerse a esta dificultad, se ha extraído mediante cuatro de ellos, uno para cada uno de los últimos embalajes. Cada una de las casillas se encuentra ligada a la variable correspondiente de la tabla definida en el configurador de fuentes de datos.





**Ilustración 21 - Tablas de datos de últimos embalajes de página de ferrosas**

### 3.2.5 Tabla de histórico

Con una distribución de elementos similar a la tabla general de cubas ferrosas, la tabla de histórico cumple una función muy distinta. Conteniendo únicamente datos de la cuba seleccionada en el control “histórico cuba”, ordena cronológicamente los valores de las variables treinta días hacia atrás desde la fecha marcada. Con ello, se puede observar la progresión a lo largo del último mes de todas ellas y estudiar la tendencia, los cambios bruscos, etc.

	Date	Age	Fe	Temperature	SPPN	MetalPadLevel	DAS	BathLevel	AIF3	Cu	Mn	Si
	13/05/2015		72.31	958		11	-7.5	16		13	5.6	12,540
	12/05/2015	1774	64.79	954	185,843338	11	-6.5	15	8.95752	12,000...	5.1000...	11.5
	11/05/2015	1773	60.2100029	969	140.1787	11	-4.5	17		12,000...	4.9	11.53
	10/05/2015	1772	56.42	956	131.5177	11				12,000...	4.7	8.41
	09/05/2015	1771	44.9	950	136.1019	11				11	3.4	6.19

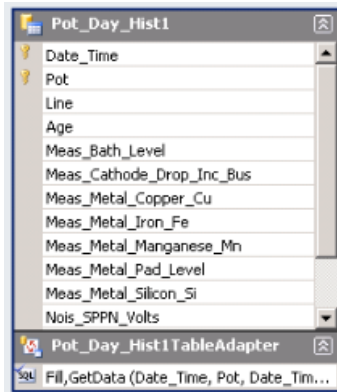
**Ilustración 22 - Tabla de histórico de página de ferrosas**

#### 3.2.5.1 Funcionamiento

Funcionamiento extremadamente parecido al de la tabla general. Las variables a extraer de la base de datos son las mismas, y el filtrado cambia ligeramente para limitar a únicamente la cuba del selector e incluir treinta días en vez de uno.



Probablemente podría haberse programado la extracción de datos utilizando la misma tabla que la general y añadiendo otro tableadapter distinto, sin embargo, se ha construido otra tabla similar en su lugar para una mejor claridad en la pantalla de tablas de fuentes de datos.



**Ilustración 23 - Tabla de datos de histórico de página de ferrosas**

### 3.2.6 Tabla de histórico de embalajes

La tabla de histórico de embalajes, agrupa por días los datos acumulados de los embalajes de la cuba seleccionada y los presenta ordenados cronológicamente desde la fecha señalada hasta treinta días atrás. Además, incluye también información sobre los fallos de alimentación.

Date	AE S1	AE S2	AE S3	AE Tot	AEt S1	AEt S2	AEt S3	AEt Tot	Fed Int	Fed Ext
13/05/2015										
12/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	7	2
11/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	2	0
10/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	0	0
09/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	0	0

**Ilustración 24 - Tabla de histórico de embalajes de página de ferrosas**

La función de la tabla de histórico de embalajes, aunque similar a la de la tabla de últimos embalajes, no es redundante, sino complementaria, ya que cada una de ellas, bebe de unos datos insuficientes para saciar todas las necesidades de información sobre embalajes por sí misma.

Una de las mayores debilidades que acarrea la tabla de últimos AEs, es su limitación en cuanto a cantidad de datos. Únicamente cuatro embalajes puede resultar suficiente en muchas ocasiones, sin embargo en otras no lo es. Desgraciadamente, los datos de embalajes registrados como eventos en la base de datos solo disponen de una pila de cuatro espacios. No es muy extraño que en ocasiones, un embalaje produzca más de un evento al bajar momentáneamente de la tensión límite, ocupando pues, dos o más huecos.

En la librería de la que se extraen los datos para la tabla de histórico de embalajes, la información no se recoge por eventos sino de forma acumulativa cada turno y cada día. Como principal ventaja que este método nos brinda, está la capacidad de retroceder en el tiempo sin límite mientras haya datos. Además, se pueden hacer interpretaciones más globales. Por otro lado, esta forma de registro abstrae totalmente la visión de los embalajes como eventos individuales, y no permite obtener datos de la fuerza o duración de cada uno. Para ello, es necesaria la otra tabla.

### 3.2.6.1 Variables

Las variables que aporta la tabla son básicamente tres: número de embalajes, duración de la suma de embalajes, y número de problemas de alimentación. Estas tres variables se encuentran desglosadas para agregar un mayor detalle.

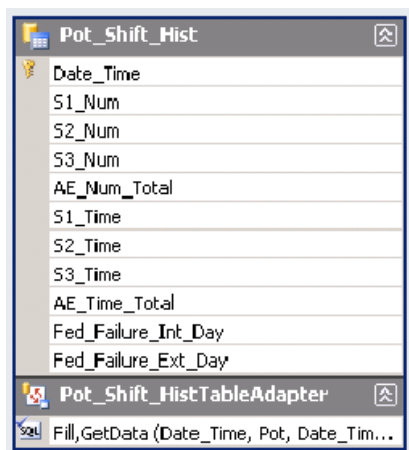
Tanto para el número como la duración de la suma de embalajes, existen tres casillas numeradas como S1, S2 y S3. Cada una de ellas representa un turno de seis horas, y permiten localizar posibles focos en los embalajes. El total se recoge en las casillas Tot marcadas en verde para una un fijado más rápido de la vista.

Los fallos de alimentación se dividen en dos columnas, fallos de alimentación interior y exterior. Esto nos permite conocer no solo el número, sino también si existe simetría o no.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.2.6.2 Funcionamiento

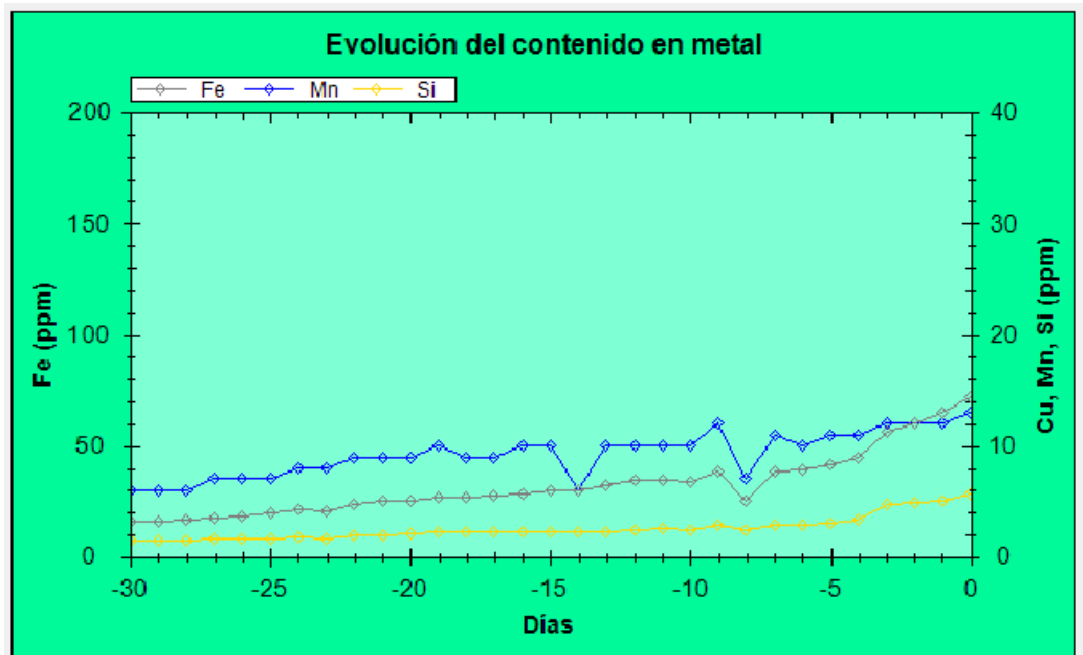
La tabla trae sus datos de la librería Pot\_Shift\_Hist del QLC. Filtrando para la cuba y la fecha señaladas, proporciona las variables de duración y número de embalajes de cada turno y los el número de fallos de alimentación del día interiores y exteriores. Para presentar las duraciones en la tabla, el formato de representación ha tenido que ser cambiado, y para obtener los totales, se efectúa la suma de los turnos. Ambas operaciones se realizan en el query en SQL.



**Ilustración 25 - Tabla de datos de histórico de embalajes de tabla de ferrosas**

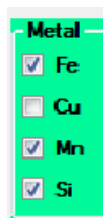
### 3.2.7 Gráfico de evolución de metales

El gráfico de evolución del contenido en metal es un gráfico de tipo línea cuyo objetivo es proporcionar una visión clara del avance le la concentración de metal en el baño a lo largo del tiempo, para una cuba concreta durante los últimos treinta días desde la fecha marcada.



**Ilustración 26 - Gráfico de evolución de metales de página de ferrosas**

Mediante la proyección de líneas de colores representando cada uno de los cuatro metales relevantes (hierro, cobre, manganeso y silicio), se pueden comprobar no solo las variaciones a lo largo del tiempo, sino también el grado de relación entre ellas. Cada una de las líneas puede activarse y desactivarse a placer mediante su checkbox correspondiente.



**Ilustración 27 - Control selector de metales de página de ferrosas**

Para favorecer una más intuitiva toma de referencias, los ejes mantienen unas escalas fijas, que en caso de no ser adecuadas siempre pueden modificarse mediante un zoom con la rueda del ratón. Sobre el eje y izquierdo se reflejará el hierro, mientras que los demás metales lo harán en el y derecho debido a sus valores mucho menores.

### 3.2.7.1 Solución a la falta de datos

Con una gráfica de tipo línea y una serie de datos sobre el contenido en metal que no siempre están disponibles con una periodicidad diaria, resulta de vital importancia no mostrar al usuario una visión distorsionada de la realidad. Además, cualquier decisión que se tome debe ser aplicable según los métodos de toma de datos de zedgraph.

Si se introduce directamente la serie de datos con sus correspondientes casillas en blanco, en la representación la línea queda dividida en trozos, lo que no resulta nada cómodo para el usuario.

Las posiciones en blanco no pueden ser eliminadas puesto que ocupan su lugar en el tiempo sobre el eje x como una serie, y hacerlo desplazaría puestos a la izquierda a los datos posteriores. De esto se deduce, que la solución pasa por otorgar valores razonables a los puntos vacíos de forma que se cierre la línea. Para ello se dispone de dos posibilidades:

1. Interpolar entre los días con dato anterior y posterior para otorgar un valor intermedio a los puntos entre ellos.
2. Aplicar exactamente el valor del día anterior a cada punto sin dato.

La opción elegida finalmente ha sido la 2, principalmente por dos motivos. El primero es que no se puede saber si realmente los valores entre los días conocidos son intermedios, especialmente en cubas ferrosas, que pueden experimentar grandes variaciones en muy poco espacio de tiempo. Además, la interpolación añadiría un problema más grave que la propia falta de dato, y es la pérdida de la certeza de que ese dato no existe. Es imprescindible que el usuario sepa cuando un dato ha sido medido y cuando no para poder otorgarle mayor o menor importancia. Utilizando el dato anterior para cada punto sin dato mantiene esa certeza, que queda patente en la perfecta horizontalidad de la línea.

### 3.2.7.2 Funcionamiento

Con los datos procedentes de la tabla de histórico, se crean las series de datos, que se introducen en la gráfica para representar en la línea. En este caso, al estar ya disponibles los datos necesarios en otra de las tablas de la página no es necesario traerlos de nuevo de la base de datos.

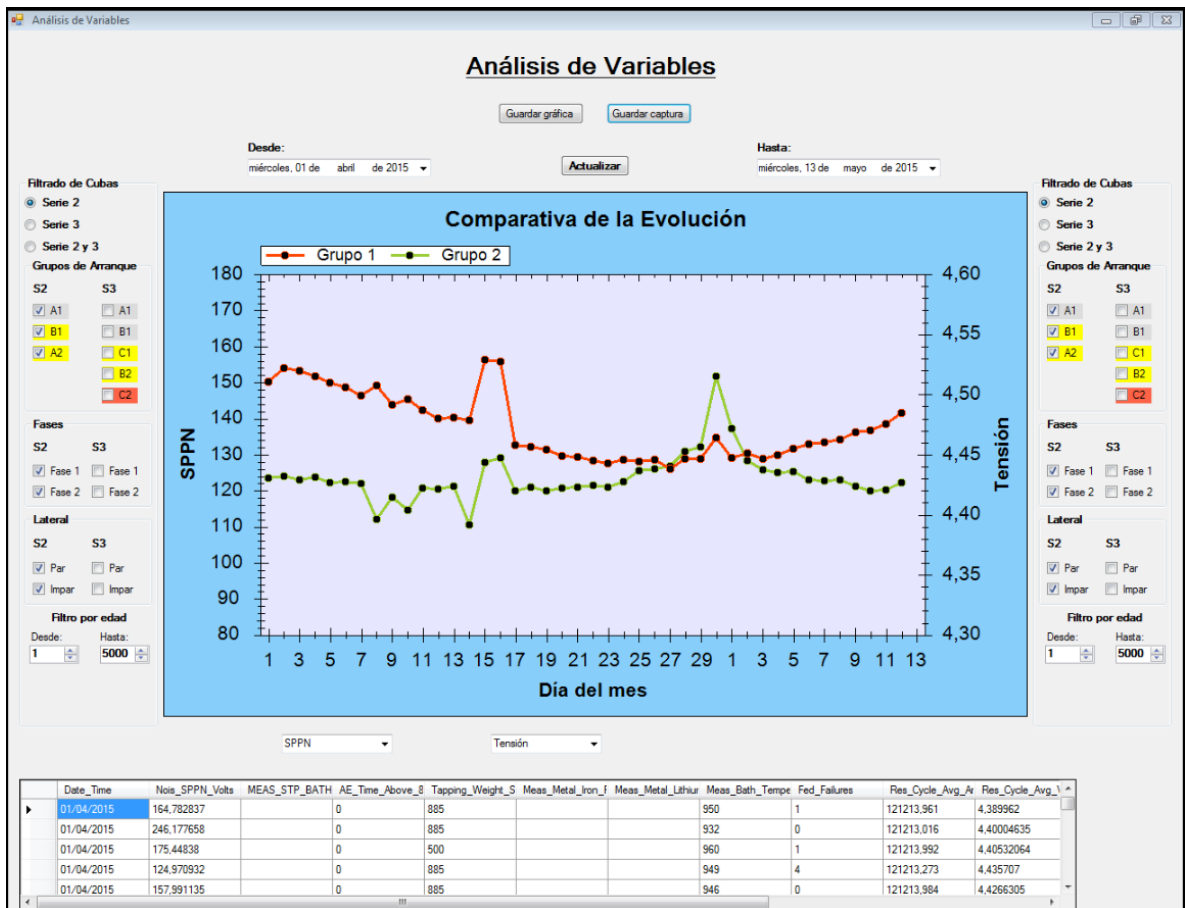
## 3.3 Análisis de grupos

La serie de electrólisis de Alcoa es un entorno de trabajo en permanente cambio. Pese a que teóricamente todo funciona del mismo modo, constantemente surgen nuevos factores que hacen heterogéneo el campo de cubas. El afán descubrir esas fuentes de cambios o analizar su influencia, lleva constantemente a comparar agrupaciones de cubas entre sí.

Las muestras empleadas para las comprobaciones, pueden variar según el objeto de interés, pero normalmente se forman a partir de ciertas condiciones que podemos considerar las más típicas, y que separan grupos tradicionalmente comparados.

Cada semana, es fácil que al menos una vez sea necesario generar algún tipo de gráfica para despejar dudas sobre alguna situación o posible suceso. Hasta ahora, eso implicaba un tedioso trabajo en Excel que pasa por obtener los datos del servidor, y filtrar las cubas casi siempre manualmente.

Esta ventana de análisis de grupos pretende agilizar la generación de gráficas para comprobaciones del día a día por medio de la ayuda al filtrado de datos en función de las condiciones más comunes. Aunque para un análisis muy concreto haga falta una selección más minuciosa de las cubas que forman la muestra, mediante esta herramienta se puede hacer un análisis preliminar para saber si aproximadamente los resultados apoyan la teoría o no.



### Ilustración 28 - Página de análisis de grupos

Como se puede observar en la imagen, una gran gráfica central ocupa la mayor parte de la pantalla. A ambos lados, dos paneles idénticos permiten la selección filtrada de los grupos de cubas a representar.

Mediante los controles de fecha se puede modificar la amplitud del eje x, que representa el tiempo. Su graduación muestra el día del mes. En los selectores de variable se elige para cada grupo la variable del eje y para cada grupo.

Gracias a que existe un selector de variable para cada grupo, las opciones de esta ventana aumentan. Seleccionando el mismo grupo de cubas para el conjunto 1 y 2 y marcando variables distintas, se pueden comprobar correlaciones en la evolución de dos variables.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado

Fecha: Febrero 2015

Hacia la parte inferior de la pantalla, se sitúa una tabla que inicialmente debería estar oculta. Es la tabla a la que se llevan los datos desde la base de datos, y de donde se toman para transformarlos y posteriormente representarlos en la gráfica. Se decidió hacerla visible a petición de los ingenieros, con el fin de poder comprobar numéricamente detalles observados en la gráfica si los resultados resultaban extraños. No se trata de una tabla ideal para la consulta, pero inicialmente no era su objetivo.

### 3.3.1 Opciones de filtrado

Las opciones para el filtrado de cubas se configuran mediante los paneles a izquierda y derecha de la gráfica. Divididos en varias categorías, permiten la acumulación de condiciones para selecciones bastante restrictivas de una forma rápida.

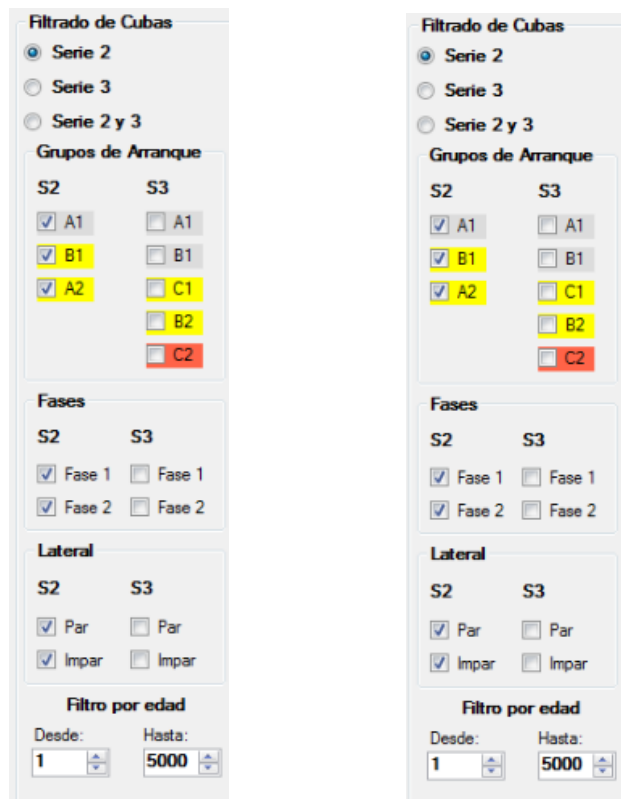


Ilustración 29 - Paneles de configuración de grupos de página de grupos



Las casillas de la izquierda corresponden a la serie 2, y las de la derecha a la serie 3. Los botones de serie 2, serie 3, y serie 2 y 3 marcarán automáticamente todas las casillas correspondientes a su pulsación, desmarcando al mismo tiempo aquellas que no procedan. Además, cada una de las checkbox puede ser marcada y desmarcada individualmente. Gracias a la conexión entre botones y casillas, éstos se ajustarán de acuerdo a las modificaciones. Esto es posible gracias a la programación de los eventos.

Las categorías por las que se pueden filtrar los grupos de cubas son las siguientes: Grupos de arranque de agujas, fase, y lateral. Adicionalmente, también se puede añadir una restricción por edad de cuba.

Grupos de arranque: Los grupos de arranque de cuba son conjuntos de cubas cuyas agujas son arrancadas por el mismo equipo de agujistas. Además, los grupos pueden tener diferente color, que indica el estado de las agujas con las que trabajan las cubas. Gris=buenas, amarillas=regular, rojas=malas. Las cubas pertenecientes al mismo grupo de arranque se sitúan de forma consecutiva en la serie para una fácil organización del trabajo, por lo que unas cubas siempre dispondrán de mejores agujas que otras.

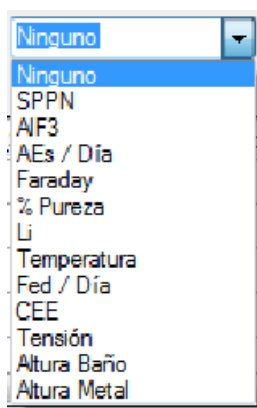
Fases y laterales: Aunque parezca condiciones basadas en la situación de las cubas no pueden resultar relevantes, la práctica demuestra lo contrario. La implantación de nuevos sistemas o medidas a las series, suele realizarse de forma progresiva, según estos segmentos físicos, por lo que para la evaluación de los resultados suele adaptarse las muestras a ellos. Además, la temperatura de la serie no es la misma en el exterior que en el interior, y los grupos de operarios son distintos.

Filtro por edad: Ajustando una edad mínima se pueden excluir del análisis las cubas demasiado jóvenes, cuya operación aún no está normalizada y no resulta comparable. Del mismo modo, se pueden retirar aquellas cubas demasiado castigadas por el tiempo en las que la variable a comparar se pueda ver influenciada en mayor medida por la edad que por lo que se quiere medir.

Todos los filtros aplicados acumulan sus restricciones, de forma que por ejemplo, partiendo de una serie 2 completa se desactiva la fase 2 y las cubas pares, únicamente las cubas impares y además de fase 1 pertenecerán al grupo.

### 3.3.2 Variables

El análisis de grupos admite una gran variedad de variables que representar en la gráfica. Pueden ser seleccionadas mediante los selectores de variable, inicialmente fijados en Ninguno.



**Ilustración 30 - Selector de variable de página de grupos**

- SPPN: Medida de la inestabilidad.
  
- ALF3: Contenido en fluoruro de aluminio; aditivo regulador de la temperatura de cuba.
  
- Aes/Día: Número de embalajes producidos al día. Es un número que puede variar mucho de un día a otro para la misma cuba, por lo que se recomiendan grupos grandes para comparar.

- Faraday: Rendimiento energético de la cuba. Calculado a partir de la colada programada, que contrariamente a lo esperado, es más precisa que la llamada colada real.
- % pureza: Porcentaje de cubas de pureza.
- Li: Cantidad de litio.
- Temperatura
- Fed/Día: Número de fallos de alimentación al día. Al igual que en el caso de los embalajes, no se recomienda crear grupos demasiado pequeños.
- CEE: Consumo específico de energía por tonelada de metal. Calculado también según la colada programada.
- Tensión
- Altura de baño
- Altura de metal

Siguiendo la idea de la gráfica de contenido en metal de la página de ferrosas, se establecieron ejes fijos para cada variable también en esta gráfica designados manualmente según los valores posibles de cada una. Al igual que en la otra gráfica, mediante el zoom se pueden variar.

### 3.3.3 Algoritmo y gestión de datos

El código utilizado para la organización y transformación de los datos a representar en la gráfica de análisis de grupos, es el más largo complicado de este programa. En él, no solo se filtran los datos, sino que también se organizan y se completan cuando no existen, se construyen a partir de ellos variables derivadas, y se preparan para la representación.

Todo comienza con el trasvase de datos desde la base de datos a la tabla auxiliar. En ella, se presentan todos aquellos necesarios para las posibles selecciones de grupos y variables entre las fechas designadas por el usuario, ordenados por día.

Posteriormente, y tras crear un array vacío en el que contener los datos finales a representar, se comienza a evaluar una a una las filas de la tabla. Para cada una, se comprueba que la cuba cumpla todas las condiciones de filtrado. Si es así, sus datos se utilizarán para alimentar el array, sumándose al valor almacenado en la posición del día al que corresponda e incrementando el contador de valores para más adelante dividir y hacer la media. En los casos en que la variable a representar sea derivada, previamente se realizarán los cálculos correspondientes.

Cuando no existe dato para una de las variables requeridas, se retrocede hasta encontrar el día anterior más próximo que disponga de él.

Como se puede comprobar, se trata de una larga, elaborada, e incluso liosa lista de instrucciones para una salida de datos relativamente sencilla. En ocasiones, a gran cantidad de consultas de variables puede comprometer la velocidad de proceso, especialmente si las variables a representar no disponen de demasiados datos almacenados en la base de datos, ya que la búsqueda de cada valor no registrado entre los días anteriores supone instrucciones extra. Es posible que no sea el algoritmo más eficiente para esta tarea, sin embargo si resulta altamente preciso y adecuado a las necesidades.

Para una comprensión detallada del algoritmo y el código que lo ejecuta, consulte el documento “Código”, y lea los numerosos comentarios.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.4 Análisis específico

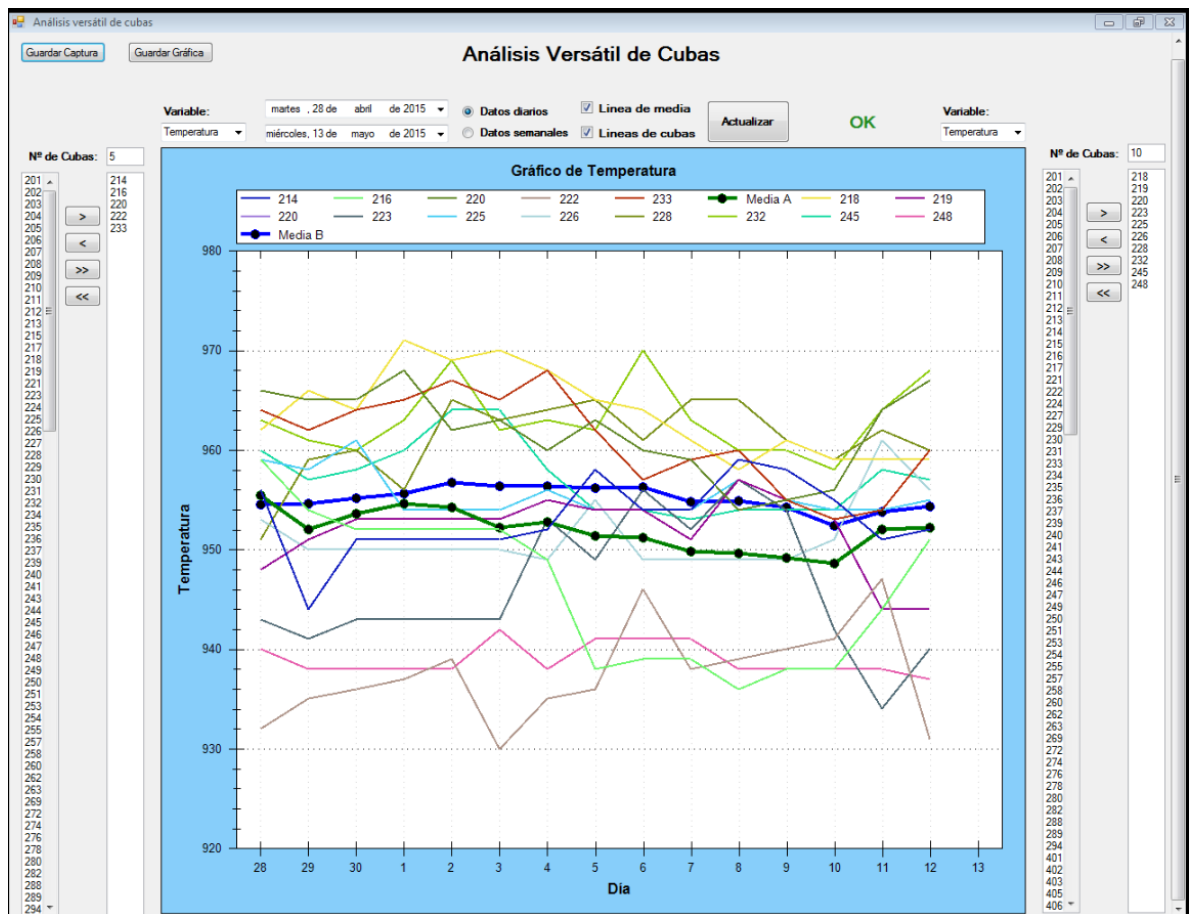


Ilustración 31 - Página de análisis específico

Cuando la ventana de análisis de grupos se queda corta en cuanto a precisión de filtrado, es necesario el uso de otro tipo de herramienta que permita configurar los grupos de cubas manualmente. Este método, sin duda más lento y tedioso, sí que resulta más versátil en cuanto a posibilidades.

La organización de la ventana, similar a la presentada por la ventana de análisis de grupos, sitúa también una gran gráfica central ocupando la mayor parte del espacio. Sustituyendo los paneles de configuración de grupos mediante condiciones de filtrado, se encuentran listas para la inclusión manual de cubas. Unos botones y un indicador de número de cubas asisten a esta labor. En la lista de cubas posibles, solo se presentan aquellas que tengan voltaje para no inducir a error.

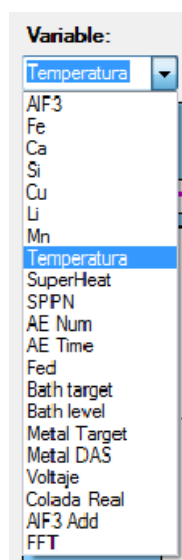
Como mejora añadida frente a la página de análisis de grupos, está la inclusión de varias nuevas opciones:

- Posibilidad de selección de análisis semanal en contraposición al diario (Con la opción marcada, las fechas se ajustan a semanas naturales automáticamente).
- Opción de representación de las cubas individuales y la línea global de media. Ya que en este caso las cubas son elegidas específicamente, puede tener sentido visualizar su aporte individual al grupo al que pertenecen.
- Activación o desactivación de la leyenda. Si no resulta relevante conocer cuál es cada cuba, y solo es importante comprobar la desviación del valor medio de las cubas en general, se puede ganar más espacio de representación.

Otra diferencia importante con la gráfica de grupos es la escala. En este caso se ha definido automática, y se ha podido comprobar que al menos para este tipo de aplicaciones resulta más beneficioso que la fija. La página de análisis de grupos podría verse beneficiada si se cambiara también su escala a automática.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.4.1 Variables



**Ilustración 32 - Selector de variable de página análisis específico**

Las variables disponibles para la representación son muy similares a las ofrecidas en la ventana de análisis de grupos. Algunas nuevas han sido añadidas a petición posterior de los ingenieros a última hora, por lo que no estarán disponibles en ambas páginas.

- ALF3: Cantidad de fluoruro de aluminio
- Fe: contenido en hierro
- Ca: contenido en calcio
- Si: contenido en silicio
- Cu: contenido en cobre
- Li: contenido en litio

- Mn: contenido en manganeso
  
- Temperatura
  
- Superheat: Diferencia entre la temperatura de cuba y la temperatura de liquidus.
  
- SPPN: Medida de la inestabilidad
  
- AE Num: Número de embalajes al día
  
- AE Time: Duración total de embalajes diarios
  
- Fed: Número de fallos de alimentación producidos al día
  
- Bath target: Objetivo de altura de baño.
  
- Bath level: Altura de baño.
  
- Metal target: Objetivo de altura de metal
  
- Metal DAS
  
- Voltaje
  
- Colada real
  
- AlF3 Add: Cantidad de fluoruro de aluminio añadida a la cuba
  
- FFT: Otra medida de inestabilidad



Una diferencia apreciable, es que debido a la utilización de un algoritmo diferente para el procesado de los datos, ha resultado imposible proporcionar los datos de CEE y Faraday calculados a partir de la colada programada. En vez de eso, se ofrecen los datos con la colada real.

### 3.4.2 Algoritmo y gestión de datos

Habiendo sido ya diseñado un algoritmo para el procesado de los datos previo a la representación en gráfica para la ventana de análisis de grupos, y siempre buscando una solución mejor, se decide orientar este segundo algoritmo de forma distinta. El nuevo algoritmo, que no solo pretende incorporar nuevas características como la representación de las cubas individuales, sino también reducir el tiempo de procesado, parte de una situación diferente.

Al igual que la gráfica de grupos, la gráfica de análisis específico toma los datos de una tabla auxiliar que en este caso sí que permanece oculta (puesto que la gráfica permite observar las cubas de forma individual, deja de ser necesario). La diferencia, es la ordenación que proporciona el query de la información depositada en la tabla: por número de cuba ascendente, y de forma secundaria por fecha.

Con esta nueva ordenación, el nuevo algoritmo es capaz de recorrer todas las filas en un número mucho menor de instrucciones. El funcionamiento a grandes rasgos es de la siguiente manera:

Para cada cuba, partiendo de un array vacío, se rellena cada posición de día según corresponda con el actual, el dato del día anterior (si no hay actual) o se deja en blanco (sin dato actual ni anterior). Si la opción de línea media está activada, se va alimentando el array correspondiente de forma paralela mientras se lleva la cuenta del número de datos mediante un array contadores.

Una vez finalizada una cuba, se representa si procede. Si la opción está marcada, se realizan los cálculos para convertir los datos en semanales, y de nuevo, si la visualización está activada, se representa. Además se va rellenando el array de media semanal.

---

Una vez finalizada una cuba, se pasa a la siguiente, hasta terminar, cuando se representan las líneas de media semanal o diaria si está marcada la opción.

Si mediante esta explicación no se comprende del todo la forma de operar, se recomienda consultar el documento "Código" y leer los comentarios.

# **Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio**

## **Conclusiones y posibles mejoras**

**David González Aguado**

# Índice

CHECK THAT! PROGRAMA DE APOYO AL ANÁLISIS DE DATOS DE ALCOA CONCLUSIONES Y POSIBLES MEJORAS     DAVID GONZÁLEZ AGUADO .....	0
ÍNDICE .....	1
1    INTRODUCCIÓN .....	2
1.1    Identificación del Proyecto.....	2
1.2    Descripción del documento.....	2
2    CONCLUSIONES .....	3
3    OPCIONES DE MEJORA .....	5
4    BIBLIOGRAFÍA.....	6

# 1 Introducción

## 1.1 Identificación del Proyecto

Título: Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio

Director: Ricardo Mayo Bayón

Autor: David González Aguado

Fecha: Febrero de 2015

Empresa: ALCOA Inespal S.A. España (Avilés)

Tutores de empresa: Jose Luís García García, Leonor Otero

## 1.2 Descripción del documento

Este documento está dedicado a la valoración final del proyecto. En él se exponen las reflexiones y conclusiones que se extraen del mismo una vez finalizado el trabajo, así como una de posibles mejoras que se podrían implementar para completarlo. Finalmente se incluye la bibliografía utilizada durante el desarrollo del proyecto.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado
---

Fecha: Febrero 2015
---------------------

## 2 Conclusiones

Tras la finalización del proyecto, se puede decir que se han cumplido satisfactoriamente los objetivos, adaptándose adecuadamente el programa desarrollado a los métodos de trabajo en Alcoa, y generando un producto de utilidad apreciable.

Las limitaciones a las que se ha visto expuesto el proyecto desde un principio, han sido superadas, evitadas o asumidas de la mejor forma posible, tratando de tomar en todo momento las mejores decisiones al alcance.

En cuanto a la cantidad de contenido que abarca el programa, ciertamente podría ser superior, pero nunca ha sido la prioridad. Tratar de ofrecer una herramienta verdaderamente útil y que sea usada en el día a día sin caer en el olvido sí lo ha sido, y en este sentido creo que se puede estar satisfecho. Cualidades como fiabilidad de los datos, sencillez y rapidez de uso, portabilidad, claridad en las opciones y utilidad real del contenido, se han cuidado para que no ofrezca barreras al usuario y resulte cómodo.

Respecto al código fuente se pueden resaltar varias conclusiones:

- En primer lugar, la limpieza y organización de código que se pretendía en un principio no se ha alcanzado. Sobre todo debido a que gran parte de las herramientas utilizadas supusieron una novedad, y durante el aprendizaje simultáneo da lugar a este tipo de problemas.
- En segundo lugar, la documentación mediante comentarios es abundante y detallada, por lo que no sería excesivamente complicado realizar modificaciones en un futuro.

Las otras dos conclusiones que se pueden extraer del proyecto, tienen que ver menos con el programa en sí, y más con las condiciones externas.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

---

Una de ellas, es lo difícil que resulta obtener exactamente el feedback buscado, ya que los puntos de vista de la persona externa y la que trabaja a diario con las herramientas de las que se discute pueden ser muy distintos. Ninguno se suele fijar en las mismas cosas.

Por último, se ha descubierto la importancia de la costumbre a la hora de pensar en los resultados. Variables que se pueden expresar en diferentes unidades, tienen mucho más o menos valor para el observador en función de la experiencia tratando con ellas, y eso es algo que se tarda mucho en cambiar. Similar a pensar en pesetas o en euros para que los valores cobren significado aunque se sepa perfectamente la relación entre ellos.

### 3 Opciones de mejora

Durante la fase final del desarrollo de un proyecto, gracias al feedback aportado por el usuario, siempre se descubren nuevas posibilidades de mejora para el producto. El manejo continuado del mismo, saca a la luz sus carencias y resalta aquellos puntos en los que se podría ampliar o mejorar. Este caso no ha resultado diferente.

De la lista de sugerencias de mejora y cambios recibida de los trabajadores de Alcoa, junto con la crítica personal, se han podido destilar varias opciones de mejora:

- Posibilidad de guardado de los grupos de cubas configurados manualmente en la página de análisis específico, de forma que se pueda repetir el análisis a las mismas cubas varios días sin tener que volver a seleccionar cada una.
- Permitir la modificación desde el programa de las cubas que comprende cada grupo de arranque de agujas en la ventana de análisis de grupos, ya que podría cambiar a medio-largo plazo y sería mucho más cómodo de cambiar que modificar el código.
- Posibilidad de exportar los datos numéricos a Excel para, en el caso de no resultar suficiente el análisis en el programa, poder continuar rápidamente fuera de él.
- Ampliación de contenido, tal vez con temática de arranques de cubas.

Por la cantidad de trabajo que implican y la falta de tiempo en el punto en que se propusieron estas mejoras, no ha sido posible comenzar su desarrollo, aunque algunas de ellas claramente resultarían muy beneficiosas.



## 4 Bibliografía

<https://msdn.microsoft.com/> Web de Microsoft para desarrolladores. Contiene gran cantidad de manuales y tutoriales que han resultado muy útiles en el aprendizaje tanto de Windows Forms, como del proceso de extracción y enlace de datos en Visual Studio. Los foros de ésta página también resultaron muy útiles en temas referentes a la compatibilidad y los problemas SxS.

<http://stackoverflow.com/> Foro de preguntas y respuestas con un amplio contenido de temática informática. En él se han podido consultar y resolver multitud de dudas y problemas específicos que un manual no recoge.

**Aplicación de escritorio para  
visualización de datos de análisis del  
proceso electrolítico del aluminio**

**Manual de Usuario**

**David González Aguado**

# Índice

<b>APLICACIÓN DE ESCRITORIO PARA VISUALIZACIÓN DE DATOS DE ANÁLISIS DEL PROCESO ELECTROLÍTICO DEL ALUMINIO      MANUAL DE USUARIO DAVID GONZÁLEZ AGUADO .....</b>		<b>0</b>
<b>ÍNDICE.....</b>		<b>1</b>
<b>ÍNDICE DE ILUSTRACIONES.....</b>		<b>3</b>
<b>1    INTRODUCCIÓN .....</b>		<b>5</b>
1.1    Identificación del proyecto.....		5
1.2    Descripción del documento.....		5
<b>2    INSTALACIÓN Y PUESTA EN MARCHA .....</b>		<b>6</b>
<b>3    MANEJO.....</b>		<b>7</b>
3.1    Ferrosas .....		9
3.1.1    Composición de la pantalla .....		10
3.1.2    Controles disponibles.....		15
3.1.3    Salvado de capturas.....		19
3.2    Análisis de Grupos.....		20
3.2.1    Composición de la pantalla .....		21
3.2.2    Controles disponibles.....		23
3.2.3    Generación de la gráfica.....		27
3.2.4    Salvado de capturas.....		28
3.2.5    Instrucciones generales .....		28
3.3    Análisis Específico .....		29
3.3.1    Configuración de ventana.....		30
3.3.2    Controles disponibles.....		31

---

3.3.3	Generación de la gráfica.....	36
3.3.4	Salvado de capturas.....	36
3.3.5	Instrucciones generales .....	37

## Índice de ilustraciones

Ilustración 1 - Menú de programa .....	7
Ilustración 2 - Ventana de ferrosas .....	9
Ilustración 3 - Tabla general de ferrosas .....	10
Ilustración 4 - Gráfico circular de pureza.....	11
Ilustración 5 - Histograma de pureza.....	12
Ilustración 6 - Tabla de últimos embalajes .....	13
Ilustración 7 - Tabla histórica de cuba ferrosa.....	14
Ilustración 8 - Tabla histórica de embalajes y alimentación .....	14
Ilustración 9 - Gráfica de contenido en metales.....	15
Ilustración 10 - Control de fecha en ventana de ferrosas.....	16
Ilustración 11 - Control de baremo de hierro .....	16
Ilustración 12 - Selector de cuba extra de tabla general de ferrosas .....	17
Ilustración 13 - Botón de limpiado de selectores de cuba extra .....	17
Ilustración 14 - Selector de amplitud de media para histograma de pureza.....	18
Ilustración 15 - Selector de serie para histograma de pureza.....	18
Ilustración 16 - Selector de cuba ferrosa.....	19
Ilustración 17 - Casillas de representación de metales.....	19
Ilustración 18 - Botones de guardado en ventana de ferrosas .....	20
Ilustración 19 - Ventana de análisis de grupos .....	20
Ilustración 20 - Gráfica de análisis de grupos .....	21
Ilustración 21 - Paneles de control de ventana de grupos.....	22
Ilustración 22 - Controles superiores de la ventana de grupos .....	22
Ilustración 23 - Selectores de variables de grupos .....	23
Ilustración 24 - Tabla auxiliar de grupos .....	23
Ilustración 25 - Controles de fecha de ventana de grupos.....	23
Ilustración 26 - Selector de variable de grupos expandido.....	24

---

Ilustración 27 - Paneles de control de ventana de grupos (parcial).....	26
Ilustración 28 - Diagrama de AND lógico .....	26
Ilustración 29 - Conmutador de selección de serie de ventana de grupos .....	27
Ilustración 30 - Filtro por edad.....	27
Ilustración 31 - Botón de actualizado de gráfica de grupos .....	27
Ilustración 32 - Botones de guardado de ventana de grupos.....	28
Ilustración 33 - Ventana de análisis específico.....	30
Ilustración 34 - Controles de fecha de análisis específico.....	31
Ilustración 35 - Selector de variable de análisis específico expandido .....	32
Ilustración 36 - Selector de tipo de dato de análisis específico.....	34
Ilustración 37 - Casillas de representación de líneas de gráfico de análisis específico ..	34
Ilustración 38 - Gestores de grupos de cubas (cortado).....	35
Ilustración 39 - Botón de actualizado de gráfica de análisis específico .....	36
Ilustración 40 - Botones de guardado de ventana de análisis específico .....	36

# 1 Introducción

## 1.1 Identificación del proyecto

Título: Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio

Director: Ricardo Mayo Bayón

Autor: David González Aguado

Fecha: Febrero de 2015

Empresa: ALCOA Inespal S.A. España (Avilés)

Tutores de empresa: Jose Luís García García, Leonor Otero

## 1.2 Descripción del documento

En este documento se presenta el manual de instrucciones del programa Check That!, comprendiendo el proceso de puesta en funcionamiento y el manejo de sus herramientas. Se explican también todas las opciones disponibles y sus usos.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 2 Instalación y puesta en marcha

No es necesaria una instalación para el funcionamiento del producto Check That! Simplemente copie y pegue la carpeta con los archivos en su disco duro sin borrar o cambiar de ubicación ninguno de ellos.

Antes de iniciar el programa haga las siguientes comprobaciones:

1. Asegúrese de que utiliza un sistema operativo Windows
2. Verifique que su equipo se encuentra conectado a la red de ALCOA y dispone de servicios de acceso a las bases de datos.

Si ambas condiciones se cumplen, proceda a arrancar el programa haciendo doble click sobre el archivo ejecutable llamado "menu".

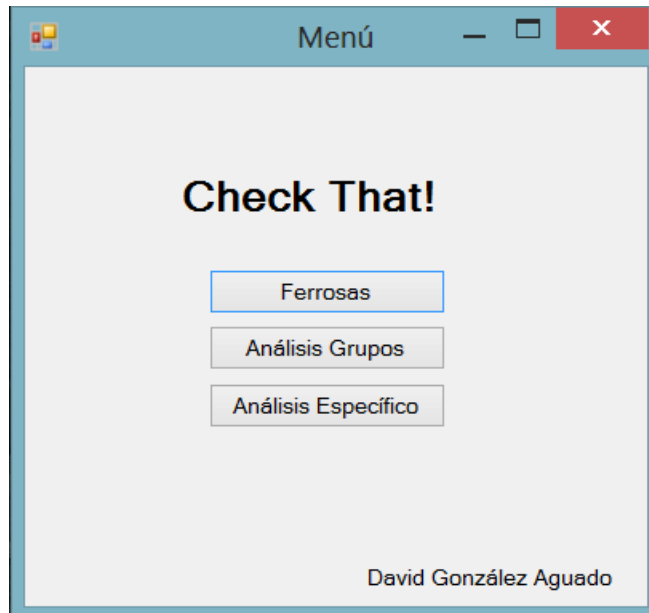
Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



### 3 Manejo

Una vez inicie el programa a través del archivo “menu”, se mostrará una ventana de pequeño tamaño en pantalla. En ella, se muestra el nombre y autor del programa, y se proporciona acceso a las diferentes funcionalidades mediante tres botones:

1. Ferrosas
2. Análisis de Grupos
3. Análisis Específico



**Ilustración 1 - Menú de programa**

Mediante la pulsación de cada botón, se abrirá una nueva ventana, que contendrá las herramientas correspondientes a esa área.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### **Gestión de ventanas:**

La apertura de cada nueva ventana no cerrará el menú, por lo que es posible mantener abiertas de forma simultanea varias de ellas. No es posible por el contrario abrir la misma múltiples veces.

El cierre de las ventanas secundarias es de carácter individual, por lo tanto no afecta a la totalidad del programa. Para cerrar una ventana pulse el símbolo “x” en la parte superior derecha de la misma.

Cuando una ventana secundaria se cierre, se guardará su estado actual hasta la finalización del programa. Por lo tanto, si mediante el botón correspondiente se vuelve a abrir la misma pestaña, la información previamente cargada en ella se mantendrá.

Para cerrar el programa completamente, pulse la “x” de la ventana principal o de menú. Esto desencadenará la finalización de todas las ventanas abiertas.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.1 Ferrosas

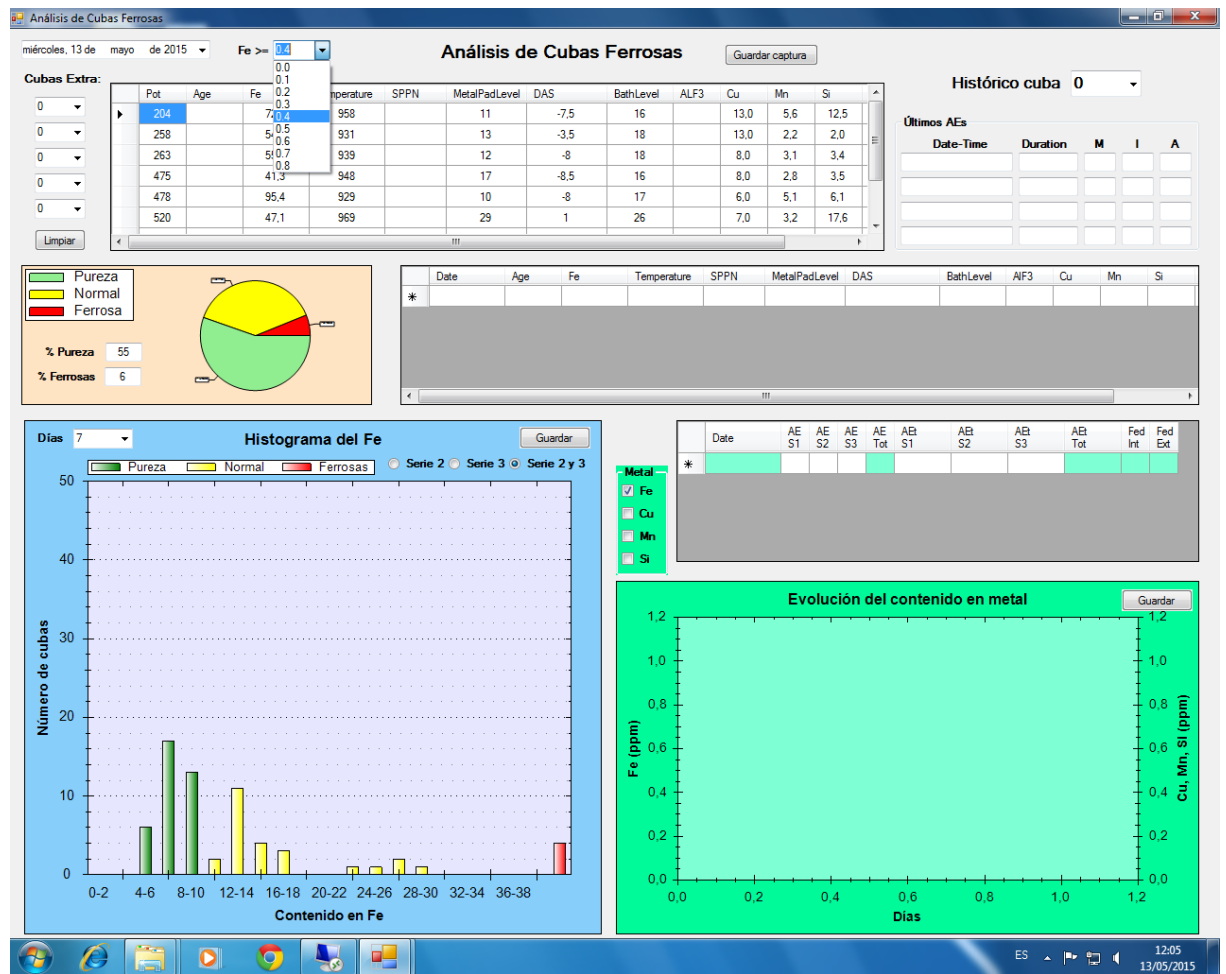


Ilustración 2 - Ventana de ferrosas

La ventana de cubas ferrosas es una página de información que recopila todos los datos relevantes para el análisis y el seguimiento de las cubas ferrosas. A través de ella, es posible estudiar tanto la información actual como el histórico.

### 3.1.1 Composición de la pantalla

#### 3.1.1.1 Tabla General

Pot	Age	Fe	Temperature	SPPN	MetalPadLevel	DAS	BathLevel	ALF3	Cu	Mn	Si
204		72,3	958		11	-7,5	16		13,0	5,6	12,5
258		54,4	931		13	-3,5	18		13,0	2,2	2,0
263		55,4	939		12	-8	18		8,0	3,1	3,4
475		41,3	948		17	-8,5	16		8,0	2,8	3,5
478		95,4	929		10	-8	17		6,0	5,1	6,1
520		47,1	969		29	1	26		7,0	3,2	17,6

#### Ilustración 3 - Tabla general de ferrosas

Situada en la esquina superior izquierda de la pantalla, muestra por pantalla las cubas clasificadas como ferrosas ( $Fe > 0.4$ ) en una lista con toda la información relevante para el día seleccionado:

- Número de cuba
- Edad
- Contenido en hierro
- Temperatura
- SPPN (medida de estabilidad de cuba)
- Nivel de metal
- DAS
- Nivel de baño

- Contenido de ALF3
- Contenido en cobre, manganeso y silicio
- Caída catódica

### 3.1.1.2 Gráficas de serie

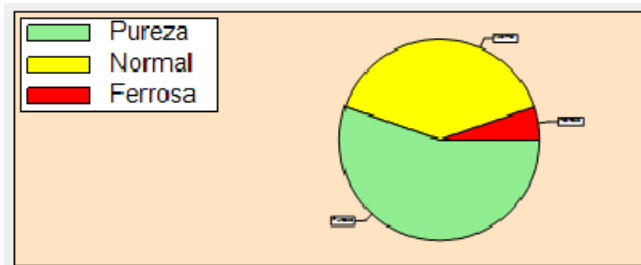
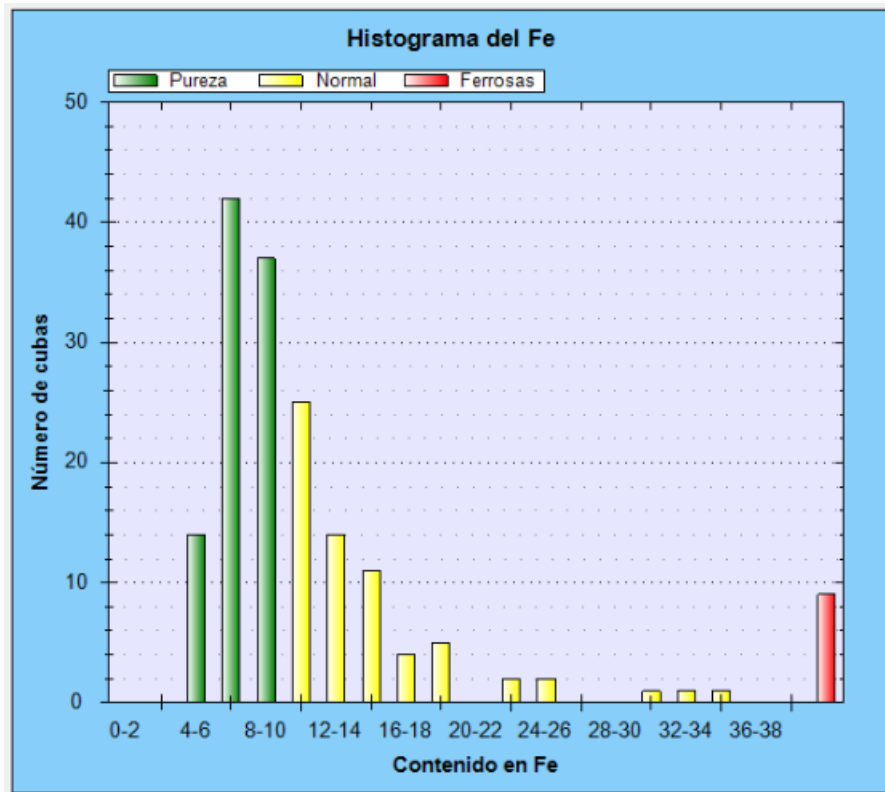


Ilustración 4 - Gráfico circular de pureza



**Ilustración 5 - Histograma de pureza**

A la parte inferior de la tabla general, se observan dos gráficas; una circular, y un histograma:

- En el gráfico circular se puede apreciar de forma clara la proporción de cubas ferrosas, normales y de alta pureza. Unos controles numéricos muestran el valor exacto.
- En el histograma se muestra la distribución de cubas por tramo de pureza para una vista más detallada.

En ambas gráficas se sigue la misma leyenda de colores, siendo rojo=cuba ferrosa, amarillo=cuba normal y verde=cuba de pureza, tal como se indica en el propio programa.

Por defecto, la información representada por las gráficas, corresponde a la media de los últimos siete días respecto a la fecha actual para la suma de las dos series.

### 3.1.1.3 Herramientas de información detallada de cuba ferrosa

A la parte derecha de la pantalla, se pueden distinguir tres tablas y una gráfica, reservados para el análisis concreto de una cuba ferrosa seleccionada. Por defecto, al arranque se encontrarán vacíos.

Últimos AEs				
Date-Time	Duration	M	I	A
20/04/2015 23:36	00:03:30	24	13	15
20/04/2015 11:34	00:02:50	26	9	16
19/04/2015 21:00	00:01:50	24	9	15
19/04/2015 21:00	00:03:20	24	8	17

**Ilustración 6 - Tabla de últimos embalajes**

En primer lugar, y bajo el selector de cuba, se encuentra la información sobre los últimos cuatro embalajes. Los datos que incluye son los siguientes:

- Fecha y hora
- Duración
- Fuerzas inicial, máxima y media del embalaje

Date	Age	Fe	Temperature	SPPN	MetalPadLevel	DAS	BathLevel	AIF3	Cu	Mn	Si
13/05/2015		72,31	958		11	-7,5	16		13	5,6	12,540
12/05/2015	1774	64,79	954	185,843338	11	-6,5	15	8,95752	12,000...	5,1000...	11,5
11/05/2015	1773	60,2100029	969	140,1787	11	-4,5	17		12,000...	4,9	11,53
10/05/2015	1772	56,42	956	131,5177	11				12,000...	4,7	8,41
09/05/2015	1771	44,9	950	136,1019	11				11	3,4	6,19

### Ilustración 7 - Tabla histórica de cuba ferrosa

Justo debajo, una tabla que contiene los mismos campos que la tabla general presenta los datos de la cuba para cada uno de los últimos treinta días desde la fecha seleccionada.

Date	AE S1	AE S2	AE S3	AE Tot	AEt S1	AEt S2	AEt S3	AEt Tot	Fed Int	Fed Ext
13/05/2015				0	00:00:00	00:00:00	00:00:00	00:00:00	7	2
12/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	2	0
11/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	0	0
10/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	0	0
09/05/2015	0	0	0	0	00:00:00	00:00:00	00:00:00	00:00:00	0	0

### Ilustración 8 - Tabla histórica de embalajes y alimentación

La última tabla, dedicada a los embalajes y fallos de alimentación, representa la información por días hasta un mes atrás. Para cada día, los campos mostrados son los siguientes:

- Número de embalajes total y en cada turno (mañana, tarde y noche)
- Duración de embalajes total y por turno
- Número de fallos de alimentación interiores y exteriores



Finalmente se encuentra una gráfica de tipo línea, que ofrece la evolución de la concentración de hierro, cobre, manganeso, y silicio a lo largo de los últimos días. Cada uno de ellos puede ser ocultado o mostrado individualmente. Por defecto, únicamente el hierro aparecerá en la gráfica.

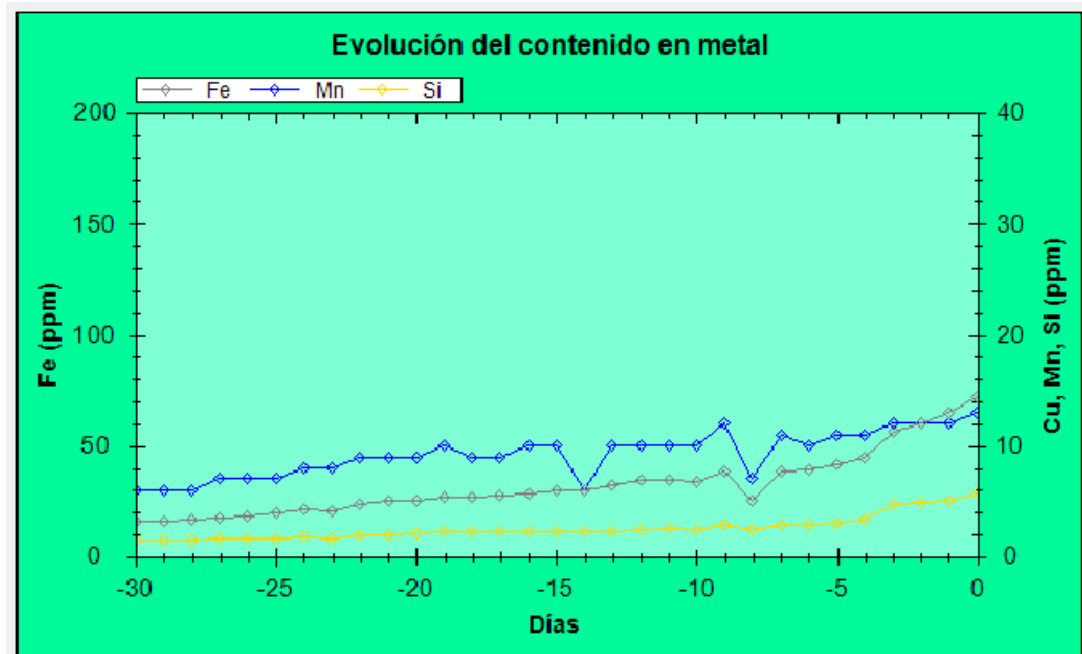


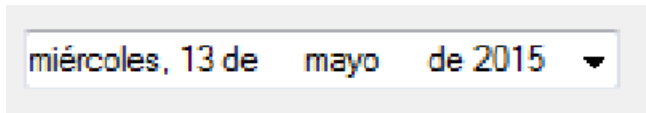
Ilustración 9 - Gráfica de contenido en metales

### 3.1.2 Controles disponibles

La interacción con los controles de la página, resulta fundamental para poder seleccionar, encontrar y representar aquella información que se esté buscando.

#### 3.1.2.1 Control de fecha

El control de fecha es de entre todos el más importante, ya que afecta a la totalidad de la información mostrada, centrándola en un punto concreto del calendario.

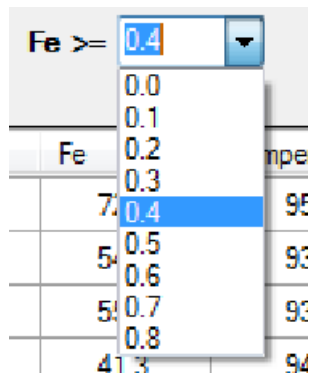


**Ilustración 10 - Control de fecha en ventana de ferrosas**

Para modificar la fecha, haga click en el control de fecha y seleccione un día válido en el calendario. Cualquier día del cual se disponga de datos históricos podrá ver sus datos representados en ésta ventana del programa.

### 3.1.2.2 Baremo de hierro

Para que la tabla general muestre una cuba, ésta debe superar el nivel de hierro marcado por el baremo. Puesto que se denominan cubas ferrosas a aquellas por encima de 0.4, éste es el valor por defecto. No obstante, es posible modificarlo para imponer una condición más o menos restrictiva.

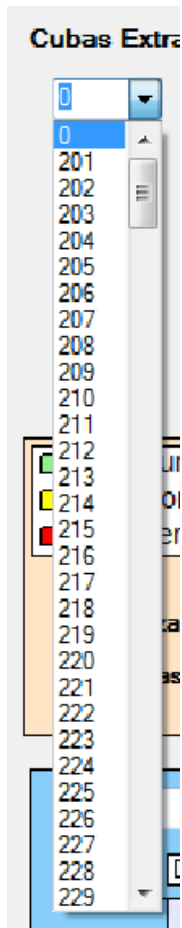


**Ilustración 11 - Control de baremo de hierro**

### 3.1.2.3 Selectores de cubas adicionales

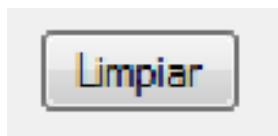
Si una cuba no supera el baremo del hierro pero se considera relevante su estudio, puede agregarse a la tabla general. Para incluir cubas específicas a la tabla general, puede hacerse mediante los selectores de cubas adicionales a la izquierda de la misma.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------



**Ilustración 12 - Selector de cuba extra de tabla general de ferrosas**

Para descartar la selección, marque la cuba 0. Si desea descartar todas las selecciones, pulse el botón limpiar que se sitúa bajo los controles de cubas adicionales.



**Ilustración 13 - Botón de limpiado de selectores de cuba extra**

### 3.1.2.4 Controles de los gráficos

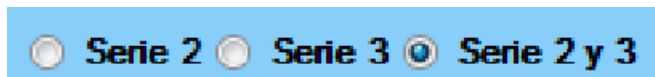
Los gráficos, que por defecto toman la media de los datos de la última semana desde el día seleccionado para las dos series, pueden ser modificados parcialmente.

Mediante el control situado en la parte superior izquierda del histograma, se ajusta el número de días atrás que se emplean para elaborar el dato de la media. Sus posibles valores están restringidos a 7 (predeterminado), 15, 30 y actual (solo utiliza los valores de la fecha seleccionada).



**Ilustración 14 - Selector de amplitud de media para histograma de pureza**

Seleccionando los botones circulares situados también en el interior del marco del histograma, se elige mostrar los datos de una de las dos series, o los conjuntos (predeterminado).



**Ilustración 15 - Selector de serie para histograma de pureza**

### 3.1.2.5 Selector de cuba

La información recogida en las tres tablas y la gráfica situadas a la derecha, siempre está referida a una cuba concreta. La selección de la misma se realiza por medio del selector de cuba, situado en la esquina superior derecha de la ventana.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

Histórico cuba 204 ▾

Ilustración 16 - Selector de cuba ferrosa

### 3.1.2.6 Checks de metales mostrados

Estas casillas tienen como función seleccionar qué metales son representados en la gráfica de evolución del contenido en metal. Pueden ser marcadas para buscar correlaciones o desmarcadas para aliviar el espacio visual a placer.

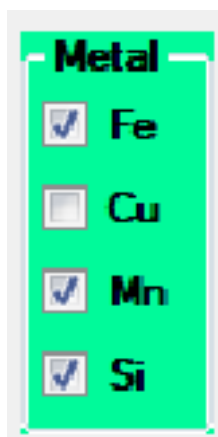


Ilustración 17 - Casillas de representación de metales

### 3.1.3 Salvado de capturas

Guardar y compartir la información es una característica fundamental de cualquier herramienta de análisis o visualización de datos. Check That! ofrece dos opciones para esto:

- Puede guardar los gráficos de forma individual mediante los botones de guardar de su esquina superior derecha. Un diálogo se abrirá, y podrá elegir un nombre y guardarlos en formato bmp.

- Puede guardar la captura completa de la ventana con el botón guardar captura situado junto al título. El método es similar al de los gráficos.

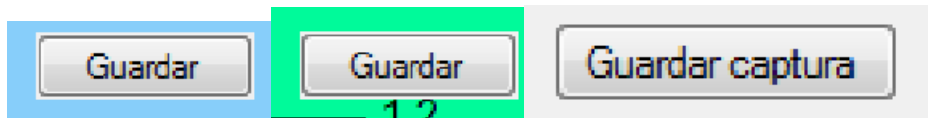


Ilustración 18 - Botones de guardado en ventana de ferrosas

## 3.2 Análisis de Grupos

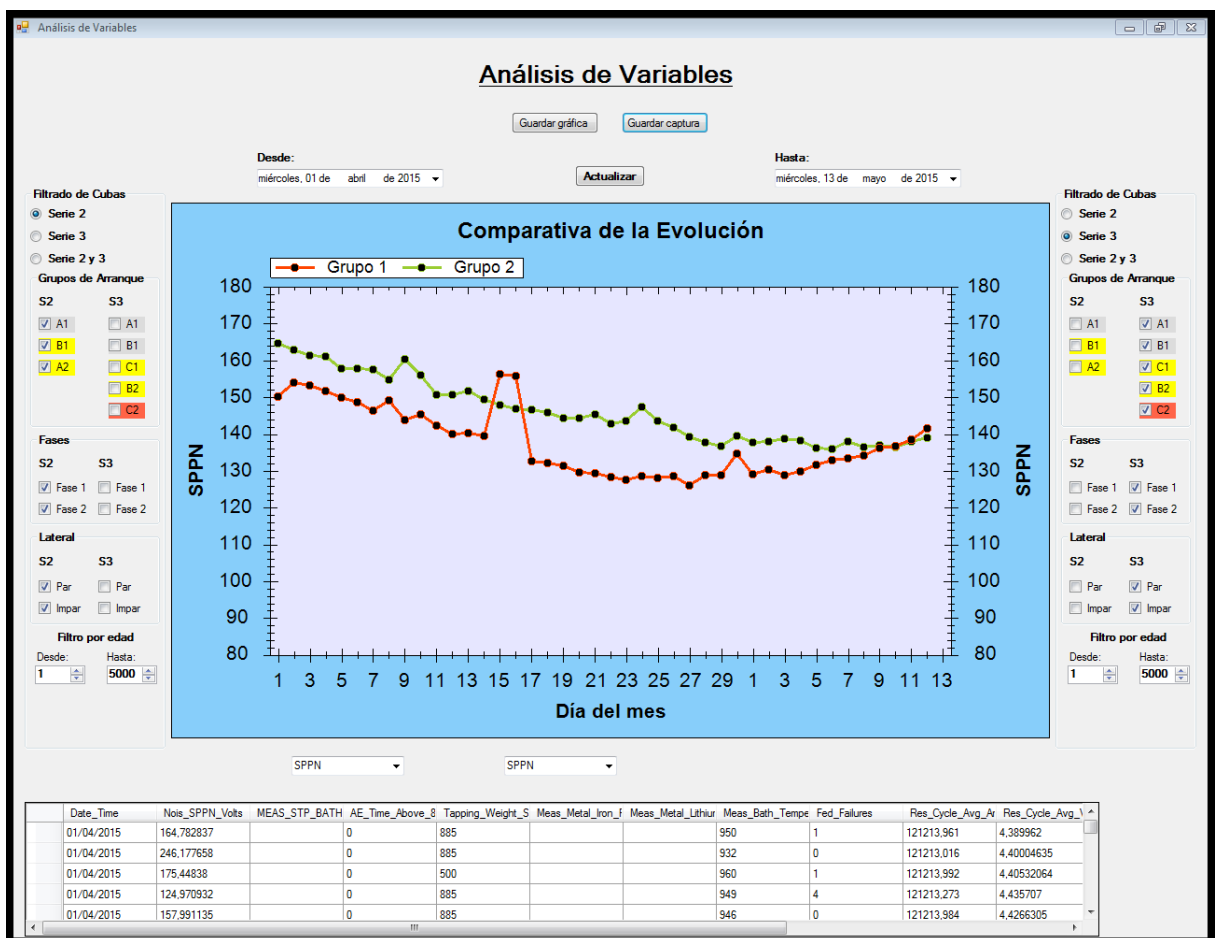


Ilustración 19 - Ventana de análisis de grupos

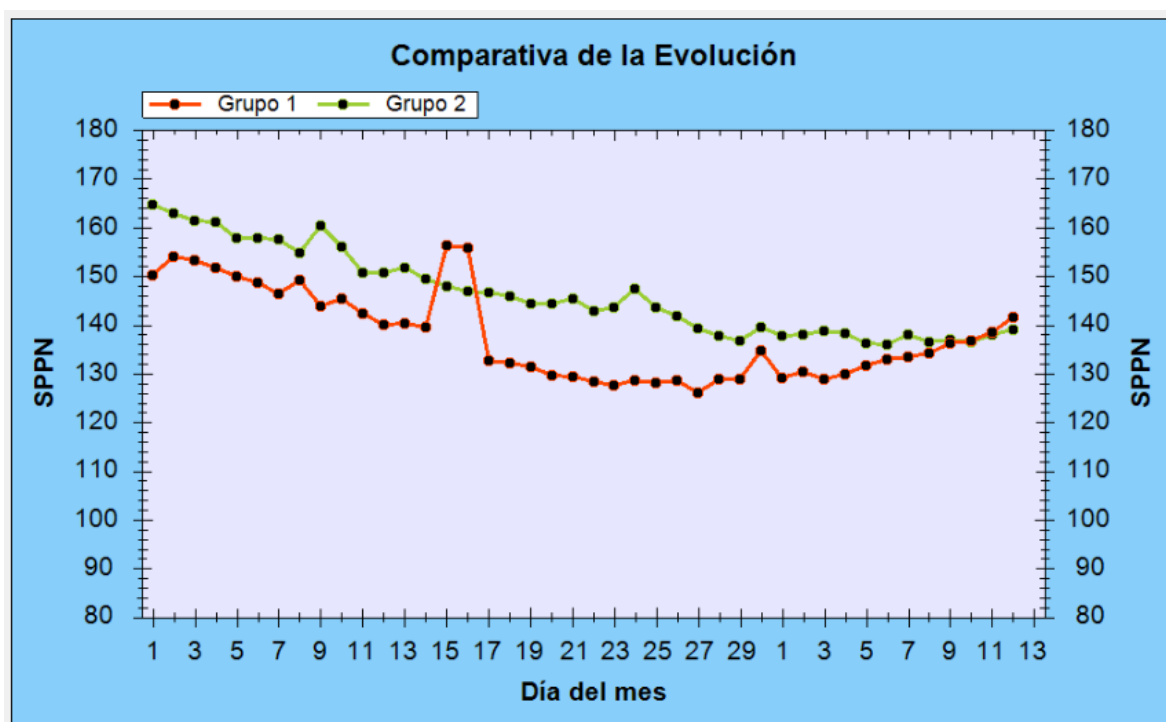
Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado

Fecha: Febrero 2015

La pestaña de análisis de grupos, titulada en el programa “Análisis de variables”, permite la creación de gráficos personalizados de grupos de cubas filtrados según condiciones comunes. Éstos, pueden ser empleados tanto como para comparar una variable entre dos grupos de cubas, como para, utilizando el mismo grupo, visualizar dos variables distintas.

### 3.2.1 Composición de la pantalla

La composición de la ventana de Análisis de grupos distribuye todos sus elementos alrededor de un único elemento central; la ventana de gráfico. En ella, a partir de la selección configurada, se representará la gráfica de tipo línea correspondiente.



**Ilustración 20 - Gráfica de análisis de grupos**

A izquierda y derecha se encuentran dos paneles de control idénticos con los que configurar los grupos de cubas A y B. La manera de hacerlo se explica en profundidad en la siguiente sección del manual de usuario.

**Ilustración 21 - Paneles de control de ventana de grupos**

En la parte superior, justo debajo del título, las opciones de salvado de información, los controles que acotan el intervalo de días de la gráfica, y el botón de actualizar coronan la representación gráfica.

**Ilustración 22 - Controles superiores de la ventana de grupos**

Debajo de ella, dos selectores permiten elegir las variables a representar.



### Ilustración 23 - Selectores de variables de grupos

Finalmente, a la parte inferior se muestra una tabla de datos. Esta tabla, que inicialmente no iba a ser mostrada en pantalla, contiene todos los datos extraídos de la base de datos, y a partir de ella se genera la gráfica. Pese a que su densidad la hace poco efectiva para consultas, resulta de utilidad para encontrar explicación a desviaciones puntuales. Fue incluida a petición de los ingenieros.

Date_Time	Nois_SPPN_Volts	MEAS_STP_BATH	AE_Time_Above_8	Tapping_Weight_S	Meas_Metal_Iron_F	Meas_Metal_Lithur	Meas_Bath_Tempe	Fed_Failures	Res_Cycle_Avg_Ar	Res_Cycle_Avg_V
01/04/2015	164.782837		0	885			950	1	121213.961	4.389962
01/04/2015	246.177658		0	885			932	0	121213.016	4.40004635
01/04/2015	175.44838		0	500			960	1	121213.992	4.40532064
01/04/2015	124.970932		0	885			949	4	121213.273	4.435707
01/04/2015	157.991135		0	885			946	0	121213.984	4.4266305

### Ilustración 24 - Tabla auxiliar de grupos

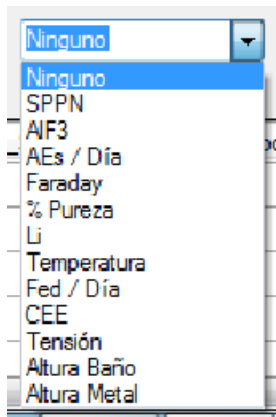
## 3.2.2 Controles disponibles

### 3.2.2.1 Controles de fecha

### Ilustración 25 - Controles de fecha de ventana de grupos

Su función es acotar el intervalo temporal reflejado en el eje x de la gráfica mediante la selección los puntos inicial y final. El día del mes se mostrará en la graduación.

### 3.2.2.2 Selectores de variable



**Ilustración 26 - Selector de variable de grupos expandido**

Hay dos selectores de variables, uno a derecha y otro a izquierda. Cada uno corresponde con un grupo de cubas y sus opciones son exactamente iguales. La variable seleccionada se representará en el eje y a un lado o al otro según corresponda.

Las variables disponibles mediante los selectores son las siguientes:

- Ninguno
- SPPN
- AIF3
- Aes /Día
- Faraday
- % de pureza
- Li

- Temperatura
  
- Fed / Día
  
- CEE
  
- Tensión
  
- Altura de baño
  
- Altura de metal

Los datos del Faraday y CEE son calculados a partir de la colada programada, por lo que garantizan la mayor exactitud posible.

### 3.2.2.3 Casillas de filtrado de cubas

A izquierda y derecha de la gráfica central, se encuentran las opciones para la configuración de los grupos de cubas 1 y 2 respectivamente. Mediante la selección de los checkbox se controla qué cubas pertenecen al grupo y cuáles no.

#### **Mecánicas de inclusión y exclusión y marcado de casillas:**

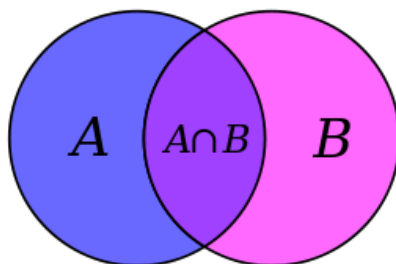
Cada uno de los dos grupos de controles idénticos dispone de 16 casillas, un conmutador de tres posiciones y un filtro de edad con dos controles numéricos.

Las 16 casillas, distribuidas en dos series y tres categorías, constituyen una forma sencilla de realizar filtrados sencillos y comunes de cubas. Cada una de las categorías (Grupos de arranque, fases y lateral) engloba todo el conjunto de cubas de la serie, dividiéndola en segmentos que pueden ser agregados o descartados.

Grupos de Arranque	
<b>S2</b>	<b>S3</b>
<input checked="" type="checkbox"/> A1	<input type="checkbox"/> A1
<input checked="" type="checkbox"/> B1	<input type="checkbox"/> B1
<input checked="" type="checkbox"/> A2	<input type="checkbox"/> C1
	<input type="checkbox"/> B2
	<input type="checkbox"/> C2
Fases	
<b>S2</b>	<b>S3</b>
<input checked="" type="checkbox"/> Fase 1	<input type="checkbox"/> Fase 1
<input checked="" type="checkbox"/> Fase 2	<input type="checkbox"/> Fase 2
Lateral	
<b>S2</b>	<b>S3</b>
<input checked="" type="checkbox"/> Par	<input type="checkbox"/> Par
<input checked="" type="checkbox"/> Impar	<input type="checkbox"/> Impar

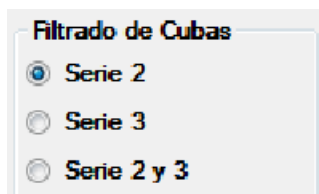
**Ilustración 27 - Paneles de control de ventana de grupos (parcial)**

Entre las categorías, el filtrado se realiza mediante condiciones AND, lo que significa que las restricciones son acumulativas. Por ejemplo: Si marcamos en la columna de la serie 1 todas las casillas de grupos de agujas, la de la fase 1, y la de par, estaremos agrupando únicamente las cubas pares de fase 1, independientemente de su grupo de agujas.



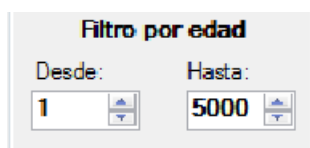
**Ilustración 28 - Diagrama de AND lógico**

Los conmutadores, inicialmente preseleccionados a Serie 2 para el grupo 1, y a serie 3 para el grupo 2, cumplen una función de preselección inicial de las casillas. Al marcar una de las tres opciones, todas las casillas de la serie o series seleccionadas se marcan, dejando limpias las demás.



**Ilustración 29 - Conmutador de selección de serie de ventana de grupos**

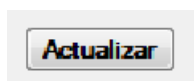
El filtro por edad limita la inclusión de aquellas cubas fuera de rango bien por ser demasiado jóvenes o bien por demasiado viejas según el rango establecido. Por defecto los valores permiten cualquier edad, pero pueden ser modificados para excluir cubas potencialmente dañinas para la homogeneidad de la muestra.



**Ilustración 30 - Filtro por edad**

### 3.2.3 Generación de la gráfica

Una vez configuradas las opciones mediante los controles disponibles a tal efecto, es posible generar la gráfica a través de la pulsación del botón actualizar:



**Ilustración 31 - Botón de actualizado de gráfica de grupos**

Desde el momento en que se pulsa este botón hasta que la gráfica se muestra en pantalla, puede transcurrir un tiempo más o menos largo en función del volumen de datos y la variable a representar. Por favor, espere con paciencia y no pulse repetidas veces, pues podría saturar el programa.

Si tras representar la gráfica se modifican los controles, un texto en rojo le hará saber que la gráfica no se encuentra actualizada, y que por tanto no se corresponde con las opciones marcadas. Una vez que se pulse de nuevo el botón de actualizar, el mensaje desaparecerá.

### 3.2.4 Salvado de capturas



**Ilustración 32 - Botones de guardado de ventana de grupos**

Para guardar una captura de la ventana, pulse el botón de guardar captura situado debajo del título. Un diálogo se abrirá y le permitirá asignar un nombre y ubicación para el archivo. Para salvar únicamente la gráfica, presione guardar gráfica.

### 3.2.5 Instrucciones generales

Para utilizar de forma efectiva las funcionalidades de la ventana de análisis de grupos siga las siguientes instrucciones para los dos grupos de cubas entendiendo el funcionamiento de los controles y las mecánicas de inclusión y exclusión:

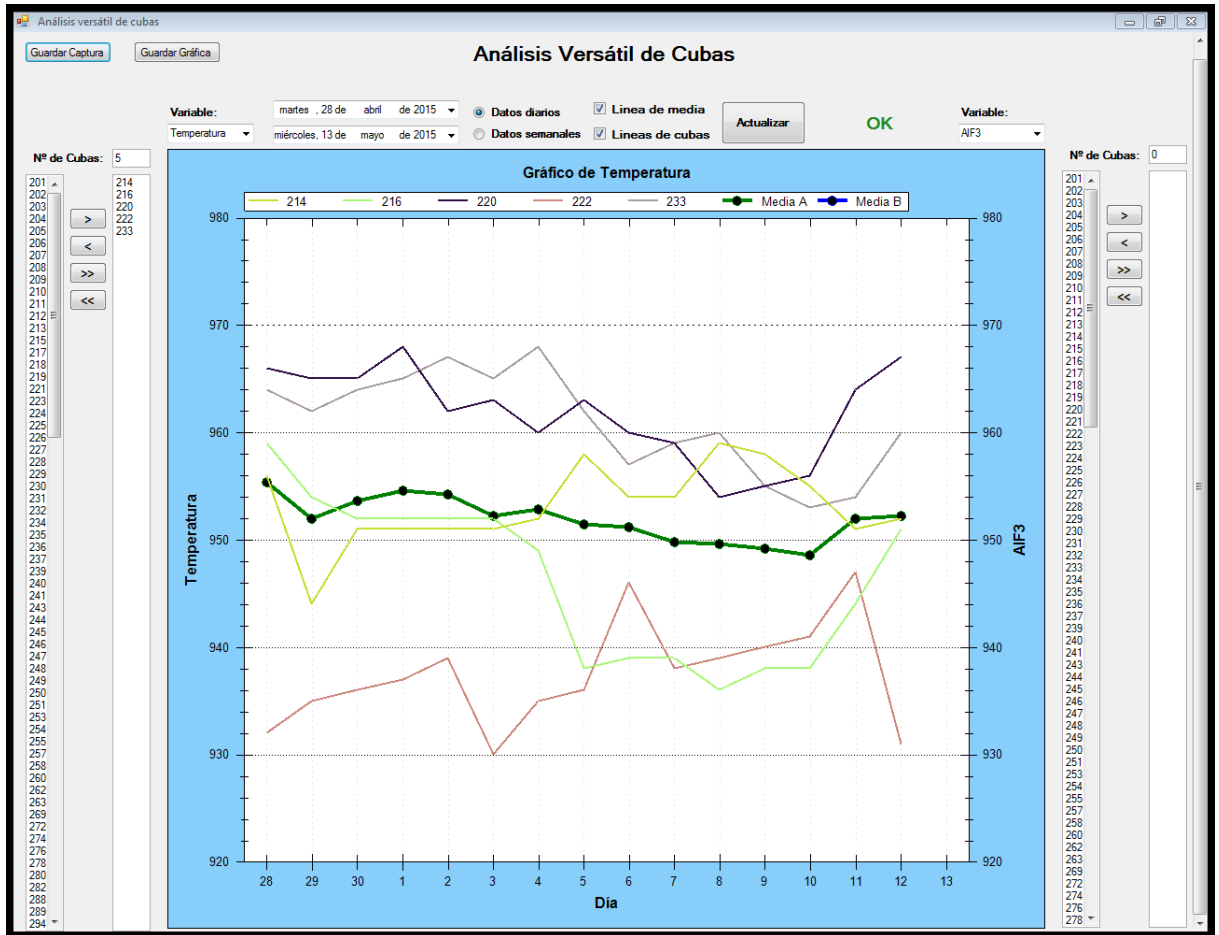
1. Seleccione el intervalo temporal del eje x mediante los controles de fecha.
2. Seleccione uno de las tres opciones conmutables para indicar a qué serie pertenecen las cubas a analizar. Esto marcará todas las checkbox de la serie seleccionada y limpiará las demás para facilitar la posterior selección.

3. Marque y desmarque casillas hasta conseguir la selección deseada de cubas.
4. Ajuste el rango de edad.
5. Presione el botón actualizar y espere a que se genere la representación

### 3.3 Análisis Específico

La tercera y última ventana del programa, llamada análisis específico, está dedicada a la creación de gráficas en las que la selección de cubas a examinar se realiza manualmente. Siendo este método mucho más lento y tedioso que el que ofrece la segunda ventana, también resulta más preciso en caso de tener unas condiciones muy concretas.

### 3.3.1 Configuración de ventana



**Ilustración 33 - Ventana de análisis específico**

Al igual que en análisis de grupos, ésta ventana se encuentra construida alrededor de una gráfica central que representa los datos seleccionados por el usuario para dos grupos separados de cubas. Cada cuba individual puede ser representada mediante una línea que representa una variable seleccionada a lo largo de un periodo de tiempo. Adicionalmente, la media del grupo de cubas puede mostrarse también en una línea de mayor grosor.

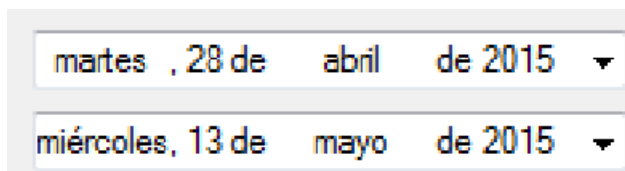


A izquierda y derecha, dos largas listas recogen todas las cubas que disponen de tensión para la fecha seleccionada. A partir de ellas se pueden formar manualmente los grupos de cubas.

En la parte superior, se agrupan multitud de controles. Entre ellos se encuentran selectores de fecha y variable, casillas de representación de líneas, conmutador de tipo de datos de salida, etc.

### 3.3.2 Controles disponibles

#### 3.3.2.1 Controles de fecha

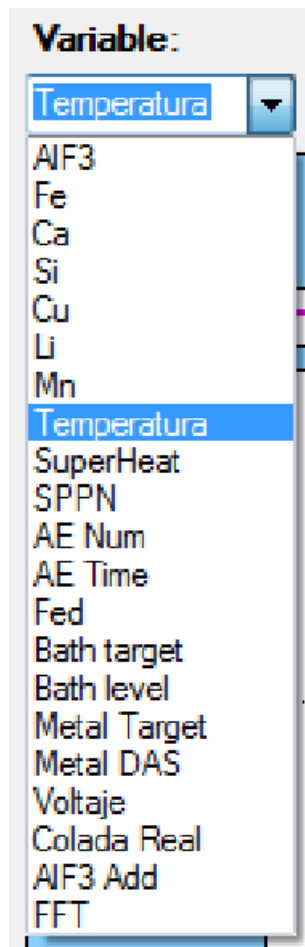


**Ilustración 34 - Controles de fecha de análisis específico**

Su función es acotar el intervalo temporal reflejado en el eje x de la gráfica mediante la selección los puntos inicial y final. El día del mes se mostrará en la graduación.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.3.2.2 Selectores de variable



**Ilustración 35 - Selector de variable de análisis específico expandido**

Hay dos selectores de variables sobre la gráfica, uno a derecha y otro a izquierda. Cada uno corresponde con un grupo de cubas y sus opciones son exactamente iguales. La variable seleccionada se representará en el eje y a un lado o al otro según corresponda. En caso de haber sido marcada la misma opción en ambas, solo se mostrará un eje y al lado izquierdo.

Las variables disponibles mediante los selectores son las siguientes:

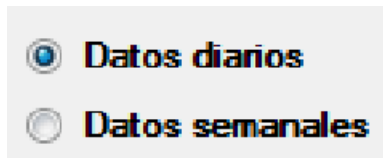
- Fluoruro de aluminio

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

- Hierro
- Calcio
- Silicio
- Manganeso
- Temperatura
- Superheat
- SPPN
- Número de embalajes
- Tiempo total de embalaje
- Número de fallos de alimentación
- Altura de baño objetivo
- Altura de baño
- DAS
- Voltaje
- Colada real

- Adición de fluoruro de aluminio
  
- FFT

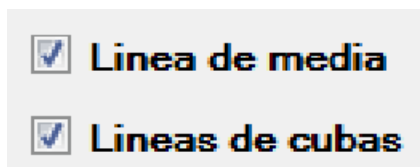
### 3.3.2.3 Selector de tipo de dato



**Ilustración 36 - Selector de tipo de dato de análisis específico**

Este selector permite elegir entre datos diarios o semanales. Los datos semanales están calculados como la media de los 7 días de la semana. Esos siete días, siempre coinciden entre lunes y viernes, dado que el marcado de ésta opción fuerza los controles de fecha a un funcionamiento automático de ajuste de la selección del usuario.

### 3.3.2.4 Casillas de representación de líneas

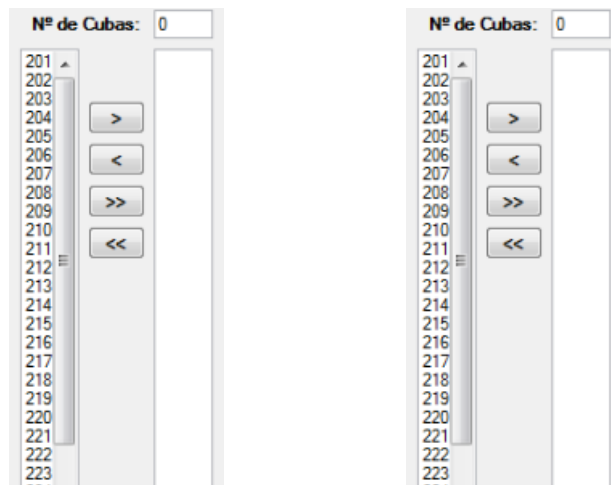


**Ilustración 37 - Casillas de representación de líneas de gráfico de análisis específico**

Permiten activar o desactivar la representación de la línea de media y las líneas de cuba individual, resultando de utilidad cuando los grupos de cubas son demasiado extensos como para representar cada cuba individualmente, o cuando la línea de media no es requerida.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.3.2.5 Gestores de grupos de cubas



**Ilustración 38 - Gestores de grupos de cubas (cortado)**

Los gestores de grupos de cubas tienen como función permitir la creación de grupos de cubas de forma manual mediante la selección de las mismas de entre una lista. Dicha lista, contiene los números de todas las cubas con tensión.

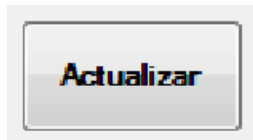
Mediante un doble click sobre la cuba deseada, o su selección y pulsación del botón de símbolo >, ese elemento se retira de la lista izquierda y se agrega a la derecha, es decir, se agrega al grupo de cubas. El proceso puede ser realizado a la inversa para eliminar cubas del grupo.

Los botones >> y << tienen efecto sobre la totalidad de los elementos de las listas, por lo que resultan de utilidad para vaciar el grupo, o rellenarlo completamente si por el volumen de cubas que se desea agregar, resulta más cómodo retirar las sobrantes.

Una caja de texto muestra en la parte superior de los gestores de grupos de cubas el número de cubas seleccionadas; un dato que puede resultar de utilidad.

### 3.3.3 Generación de la gráfica

Una vez configuradas las opciones mediante los controles disponibles a tal efecto, es posible generar la gráfica a través de la pulsación del botón actualizar:



**Ilustración 39 - Botón de actualizado de gráfica de análisis específico**

Desde el momento en que se pulsa este botón hasta que la gráfica se muestra en pantalla, puede transcurrir un tiempo más o menos largo en función del volumen de datos y la variable a representar. Por favor, espere con paciencia y no pulse repetidas veces, pues podría saturar el programa.

Si tras representar la gráfica se modifican los controles, un texto en rojo le hará saber que la gráfica no se encuentra actualizada, y que por tanto no se corresponde con las opciones marcadas. Una vez que se pulse de nuevo el botón de actualizar, el mensaje desaparecerá y se mostrará un OK en letras verdes en su lugar.

### 3.3.4 Salvado de capturas



**Ilustración 40 - Botones de guardado de ventana de análisis específico**

Para guardar una captura de la ventana, pulse el botón de guardar captura situado debajo del título. Un diálogo se abrirá y le permitirá asignar un nombre y ubicación para el archivo. Para salvar únicamente la gráfica, presione guardar gráfica.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

### 3.3.5 Instrucciones generales

Siga los siguientes pasos para utilizar de manera efectiva la ventana de análisis versátil:

1. Seleccione el intervalo temporal deseado mediante los dos controles de fecha disponibles
2. Agregue las cubas a analizar formando uno o dos grupos en los gestores de grupos de cubas.
3. Elija las variables a representar para los grupos 1 y 2
4. Seleccione datos diarios o semanales
5. Desactive la línea de media o las líneas de cuba si lo desea
6. Presione actualizar y espere hasta que la gráfica muestre los resultados

**Check That!**

**Programa de apoyo al análisis de datos  
de Alcoa**

**Código**

**David González Aguado**



# Índice

## CHECK THAT! PROGRAMA DE APOYO AL ANÁLISIS DE DATOS DE ALCOA

CÓDIGO	DAVID GONZÁLEZ AGUADO .....	0
ÍNDICE .....		1
<b>1</b>	<b>INTRODUCCIÓN .....</b>	<b>3</b>
1.1	Identificación del proyecto.....	3
1.2	Descripción del documento.....	3
<b>2</b>	<b>ARCHIVOS Y ORGANIZACIÓN .....</b>	<b>4</b>
2.1	Form1.h - Menú .....	4
2.2	Form2.h - Ferrosas .....	8
2.3	Form3.h - Grupos .....	69
2.4	Form4.h - Específico .....	131
2.5	Querys .....	174
2.5.1	Comparativa .....	174
2.5.2	Comparativa 1.....	175
2.5.3	Pot_Day_Barras_Latest.....	176
2.5.4	Pot_Day_Barras.....	176
2.5.5	Pot_Shift_Hist.....	177
2.5.6	Pot_Day_Hist.....	178
2.5.7	Pot_Day_Hist1.....	179
2.5.8	Pot_Day_Hist2 .....	180
2.5.9	All_Event_Log.....	180
2.5.10	All_Event_Log1.....	181
2.5.11	All_Event_Log2 .....	181
2.5.12	All_Event_Log3 .....	182

---

2.5.13	L_Pot_Action_Latest .....	182
--------	---------------------------	-----

# 1 Introducción

## 1.1 Identificación del proyecto

Título: Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio

Director: Ricardo Mayo Bayón

Autor: David González Aguado

Fecha: Febrero de 2015

Empresa: ALCOA Inespal S.A. España (Avilés)

Tutores de empresa: Jose Luís García García, Leonor Otero

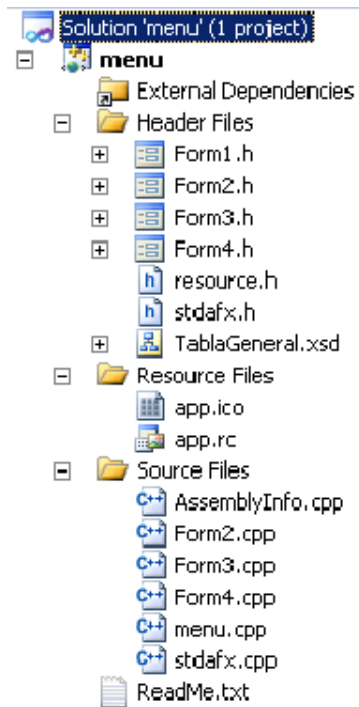
## 1.2 Descripción del documento

Este documento recoge el código del programa organizado jerárquicamente. El objetivo del mismo no es otro que el de servir de consulta en cuestiones puntuales. Para comprender los razonamientos que motivan las decisiones tomadas en él, deben leerse los documentos “Memoria” y “Elaboración”. Para análisis más detallados del código, consultar el código fuente.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado	Fecha: Febrero 2015
---	---------------------

## 2 Archivos y organización

La estructura jerárquica en la que los archivos de la solución han sido ordenados, ha venido dada por la configuración automática de Visual Studio. Las categorías existentes se corresponden con el tipo de archivo que contienen, y dado que se trata de un programa de una envergadura moderada, no resulta procedente agruparlos de otra manera.



Dentro de Header Files, podemos encontrar varios archivos. Aparte de resource.h y stdafx.h, creados de forma automática al inicio del proyecto, existen cuatro archivos del tipo Form\*.h, y TablaGeneral.xsd. Cada uno de los archivos de cabecera Form, contiene el código asociado a una de las ventanas del programa, mientras que TablaGeneral.xsd, es un archivo de código generado automáticamente a partir de los editores de conexiones de datos al configurar la conexión con la base de datos.

### 2.1 Form1.h – Menú

El código contenido en este archivo corresponde a la ventana inicial de menú, que da acceso a las otras ventanas.

Aplicación de escritorio para visualización de datos de análisis del proceso electrolítico del aluminio – David González Aguado
---

Fecha: Febrero 2015
---------------------

```

#pragma once

#include "Form2.h"
#include "Form3.h"
#include "Form4.h"

namespace menu {

    using namespace System;
    using namespace System::ComponentModel;
    using namespace System::Collections;
    using namespace System::Windows::Forms;
    using namespace System::Data;
    using namespace System::Drawing;

    /// <summary>
    /// Summary for Form1
    ///
    /// WARNING: If you change the name of this class, you will need to
change the
    ///         'Resource File Name' property for the managed resource
compiler tool
    ///         associated with all .resx files this class depends on.
Otherwise,
    ///         the designers will not be able to interact properly with
localized
    ///         resources associated with this form.
    /// </summary>
    public ref class Form1 : public System::Windows::Forms::Form
    {
    public:
        Form1(void)
        {
            InitializeComponent();
            dos = gcnew Form2();
            tres = gcnew Form3();
            cuatro = gcnew Form4();
            //
            //TODO: Add the constructor code here
            //
        }

    protected:
        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        ~Form1()
        {
            if (components)
            {
                delete components;
            }
        }

    private: System::Windows::Forms::Button^ button1;

```

```

private: System::Windows::Forms::Button^ button2;
private: System::Windows::Forms::Button^ button3;
protected:

private:
    /// <summary>
    /// Required designer variable.
    Form2^ dos;
    Form3^ tres;
    Form4^ cuatro;
private: System::Windows::Forms::Label^ programador_lab;
private: System::Windows::Forms::Label^ title_lab;
    /// </summary>
    System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
    /// <summary>
    /// Required method for Designer support - do not modify
    /// the contents of this method with the code editor.
    /// </summary>
    void InitializeComponent(void)
    {
        this->button1 = (gcnew System::Windows::Forms::Button());
        this->button2 = (gcnew System::Windows::Forms::Button());
        this->button3 = (gcnew System::Windows::Forms::Button());
        this->programador_lab = (gcnew
System::Windows::Forms::Label());
        this->title_lab = (gcnew
System::Windows::Forms::Label());
        this->SuspendLayout();
        //
        // button1
        //
        this->button1->Location = System::Drawing::Point(86,
102);

        this->button1->Name = L"button1";
        this->button1->Size = System::Drawing::Size(111, 23);
        this->button1->TabIndex = 0;
        this->button1->Text = L"Ferrosas";
        this->button1->UseVisualStyleBackColor = true;
        this->button1->Click += gcnew System::EventHandler(this,
&Form1::button1_Click);
        //
        // button2
        //
        this->button2->Location = System::Drawing::Point(86,
131);

        this->button2->Name = L"button2";
        this->button2->Size = System::Drawing::Size(111, 23);
        this->button2->TabIndex = 1;
        this->button2->Text = L"Análisis Grupos";
        this->button2->UseVisualStyleBackColor = true;
        this->button2->Click += gcnew System::EventHandler(this,
&Form1::button2_Click);
        //
        // button3
        //

```

```

        this->button3->Location      =      System::Drawing::Point(86,
160);
        this->button3->Name = L"button3";
        this->button3->Size = System::Drawing::Size(111, 23);
        this->button3->TabIndex = 2;
        this->button3->Text = L"Análisis Específico\r\n";
        this->button3->UseVisualStyleBackColor = true;
        this->button3->Click += gcnew System::EventHandler(this,
&Form1::button3_Click);
        //
        // programador_lab
        //
        this->programador_lab->AutoSize = true;
        this->programador_lab->Location
System::Drawing::Point(158, 251);
        this->programador_lab->Name = L"programador_lab";
        this->programador_lab->Size = System::Drawing::Size(122,
13);
        this->programador_lab->TabIndex = 3;
        this->programador_lab->Text = L"David González Aguado";
        //
        // title_lab
        //
        this->title_lab->AutoSize = true;
        this->title_lab->Font
System::Drawing::Font(L"Microsoft Sans Serif", 16,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0));
        this->title_lab->Location = System::Drawing::Point(69,
52);
        this->title_lab->Name = L"title_lab";
        this->title_lab->Size = System::Drawing::Size(139, 26);
        this->title_lab->TabIndex = 4;
        this->title_lab->Text = L"Check That!";
        //
        // Form1
        //
        this->AutoScaleDimensions = System::Drawing::SizeF(6,
13);
        this->AutoScaleMode
System::Windows::Forms::AutoScaleMode::Font;
        this->ClientSize = System::Drawing::Size(292, 273);
        this->Controls->Add(this->title_lab);
        this->Controls->Add(this->programador_lab);
        this->Controls->Add(this->button3);
        this->Controls->Add(this->button2);
        this->Controls->Add(this->button1);
        this->Name = L"Form1";
        this->Text = L"Menú";
        this->ResumeLayout(false);
        this->PerformLayout();
    }

#pragma endregion
private: System::Void button1_Click(System::Object^ sender,
System::EventArgs^ e) {

```

```

        dos->Visible = true;
    }
    private: System::Void button2_Click(System::Object^ sender,
System::EventArgs^ e) {
        tres->Visible = true;
    }
    private: System::Void button3_Click(System::Object^ sender,
System::EventArgs^ e) {
        cuatro->Visible = true;
    }
};
}

```

## 2.2 Form2.h – Ferrosas

Contiene el código de la ventana de ferrosas.

```

#pragma once

using namespace ZedGraph;
using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;

namespace menu {

    public ref class Form2 : public System::Windows::Forms::Form
    {
    public:

        Form2(void) //Constructor
        {
            InitializeComponent();
        }

    protected:
        ~Form2() //Destructor
        {
            if (components)
            {
                delete components;
            }
        }

        int rbflag1;
    private: System::Windows::Forms::GroupBox^ metal;
}

```



```

        private: System::Windows::Forms::DataGridViewTextBoxColumn^
dateTimeDataGridViewTextBoxColumn3;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
s1NumDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
s2NumDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
s3NumDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
aENumTotalDataGridViewTextBoxColumn1;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
s1TimeDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
s2TimeDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
s3TimeDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
aETimeTotalDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
Fed_Failure_Int_Day;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
Fed_Failure_Ext_Day;
        private: System::Windows::Forms::Label^ titulo_lab;
        private: System::Windows::Forms::ComboBox^ diasbarras_cb;
        private: System::Windows::Forms::Label^ diasbarras_lab;
        private: ZedGraph::ZedGraphControl^ pie_zg;
        private: System::Windows::Forms::TextBox^ pureza_tb;
        private: System::Windows::Forms::TextBox^ ferrosa_tb;

        private: System::Windows::Forms::Label^ label6;
        private: System::Windows::Forms::Label^ label7;
        private: System::Windows::Forms::Button^ print_but;
        private: System::Windows::Forms::DataGridView^ barras2_dgv;
        private: System::Windows::Forms::BindingSource^
Pot_Day_Barras_LatestBindingSource;
        private:
menu::TablaGeneralTableAdapters::Pot_Day_Barras_LatestTableAdapter^
Pot_Day_Barras_LatestTableAdapter;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
potDataGridViewTextBoxColumn2;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalIronFeDataGridViewTextBoxColumn3;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
lineDataGridViewTextBoxColumn1;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
resCycleAvgAmpsControlDataGridViewTextBoxColumn;
        private: System::Windows::Forms::Button^ savehist_but;
        private: System::Windows::Forms::Button^ saveline_but;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
dateTimeDataGridViewTextBoxColumn1;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
ageDataGridViewTextBoxColumn1;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalIronFeDataGridViewTextBoxColumn1;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Bath_Temperature_Control;

```

```

private:          System::Windows::Forms::DataGridViewTextBoxColumn^
noisSPNVoltsDataGridViewTextBoxColumn1;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalPadLevelDataGridViewTextBoxColumn1;
private: System::Windows::Forms::DataGridViewTextBoxColumn^  ELDAS;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measBathLevelDataGridViewTextBoxColumn1;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
MEAS_STP_BATH_XSALF3;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalCopperCuDataGridViewTextBoxColumn1;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalManganeseMnDataGridViewTextBoxColumn1;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalSiliconSiDataGridViewTextBoxColumn1;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measCathodeDropIncBusDataGridViewTextBoxColumn1;
protected:

protected:
    int rbflag2;

void histograma ()
{
    double piepureza=0;
    double pienormal=0;
    double pieferrosa=0;

    this->zedGraphControl2->GraphPane->CurveList->Clear();
    this->zedGraphControl2->Invalidate();
    int i=0;
    int j=0;
    int k=0;
    int flagrepetida=0;
    int counter;
    int hueco=0;
    double Fe[1000];
    for (j; j<1000;j++) Fe[j]=0;

    //Para los datos de media de los últimos días
    if (diasbarras_cb->SelectedIndex!=0) {
        this->Pot_Day_barrasTableAdapter->Fill(this->TablaGeneral->Pot_Day_barras, this->dateTimePicker1->Value.Date.AddDays(-System::Convert::ToInt32(this->diasbarras_cb->Text)), this->dateTimePicker1->Value.Date, rbflag1, rbflag2);

        //Para cada cuba, se comprueba si esa cuba ya se guardó
        antes. Si es así, se activa el flag.
        for (i;i<barras_dgv->Rows->Count-1;i++) {
            flagrepetida=0;
            j=0;
            for (j;i>j;j++) {
                if (barras_dgv->Rows[i]->Cells[1]->Value->Equals(barras_dgv->Rows[j]->Cells[1]->Value)==1) {
                    flagrepetida=1;
                    break;
                }
            }
        }
    }
}

```

```

    }
    }
    if (flagrepetida==1) //Con cuba repetida se
suma un hueco
        hueco++;
        if (flagrepetida==0) { //Con cuba única, se
buscan las cubas repetidas posteriores y suma el Fe y se añade uno al
contador de número de valores, que se guarda en la posición i menos el
número de huecos hasta ahora
            counter=1;
            Fe[i-hueco]=
Convert::ToDouble(barras_dgv->Rows[i]->Cells[2]->Value);
            k=i+1;
            for (k;k<barras_dgv->Rows->Count-1;k++)
{
                if (barras_dgv->Rows[k]->Cells[1]-
>Value->Equals(barras_dgv->Rows[i]->Cells[1]->Value)==1) {
                    Fe[i-hueco]=    Fe[i-hueco]+
Convert::ToDouble(barras_dgv->Rows[k]->Cells[2]->Value);
                    counter++;
                }
            }
            Fe[i-hueco]=Fe[i-hueco]/counter; //Al
acabar de sumar todos los valores de hierro, se divide entre el número de
ellos para hacer la media
        }
    }
}
//Para los datos actuales
else {
    for (i;i<barras2_dgv->Rows->Count-1;i++) {
        Fe[i]=Convert::ToDouble(barras2_dgv->Rows[i]-
>Cells[1]->Value);
    }
}

array<String^>^ labels = gcnew array<String^> {"0-2",
"2-4", "4-6", "6-8", "8-10", "10-12", "12-14", "14-16", "16-18", "18-20",
"20-22", "22-24", "24-26", "26-28", "28-30", "30-32", "32-34", "34-36",
"36-38", "28-40", ">40"};
array<double>^ x = gcnew array<double>(21);
array<double>^ y = gcnew array<double>(21);
array<double>^ z = gcnew array<double>(21);

for (i=0;i<21;i++) { x[i]=0; y[i]=0; z[i]=0; }
i=0;
while (Fe[i]>0) {
    if (Fe[i]<40 && Fe[i]>=10.9)
    {
        y[Convert::ToInt32(Math::Truncate(Fe[i]/2))];
        pienormal++;
    }
    else if (Fe[i]>=40) {
        z[20]++;
        pieferrosa++;
    }
}

```

```

    }
    else if (Fe[i]<10) {

x[Convert::ToInt32(Math::Truncate(Fe[i]/2))]+=;
        piepureza++;
    }
    i++;
}

BarItem   ^barrashierro3   =   this->zedGraphControl2-
>GraphPane->AddBar("Pureza",   nullptr, x, Color::Green);
BarItem   ^barrashierro1   =   this->zedGraphControl2-
>GraphPane->AddBar("Normal",   nullptr, y, Color::Yellow);
BarItem   ^barrashierro2   =   this->zedGraphControl2-
>GraphPane->AddBar("Ferrosas",   nullptr, z, Color::Red);

//Fabrica el eje X a partir de los labels
this->zedGraphControl2->GraphPane->BarSettings->Type=
BarType::SortedOverlay;
this->zedGraphControl2->GraphPane->XAxis->MajorTic-
>IsBetweenLabels=true;
this->zedGraphControl2->GraphPane->XAxis->Scale-
>TextLabel=labels;
this->zedGraphControl2->GraphPane->XAxis->Type=
AxisType::Text;
this->zedGraphControl2->AxisChange();
//Fija el eje Y
this->zedGraphControl2->GraphPane->YAxis->Scale->Max=50;

//Piechart
this->pie_zg->GraphPane->CurveList->Clear();
this->pie_zg->Invalidate();
this->pie_zg->GraphPane->Title->IsVisible=false;
this->pie_zg->GraphPane->Legend->Position           =
ZedGraph::LegendPos::Left;
this->pie_zg->GraphPane->Legend->FontSpec->Size = 40;

PieItem   pieslice1       =       this->pie_zg->GraphPane-
>AddPieSlice(piepureza, Color::LightGreen, 0, "Pureza");
PieItem   pieslice2       =       this->pie_zg->GraphPane-
>AddPieSlice(pienormal, Color::Yellow, 0, "Normal");
PieItem   pieslice3       =       this->pie_zg->GraphPane-
>AddPieSlice(pieferrosa, Color::Red, 0, "Ferrosa");
/*
//pieslice1.IsVisible = false;
pieslice1.Label->IsVisible = false;
pieslice2.LabelType = PieLabelType::None;
pieslice3.LabelType = PieLabelType::None;
*/
this->pie_zg->Invalidate();
this->pie_zg->AxisChange();

//textbox de porcentajes
double purezatotal, ferrosatotal;
purezatotal           =
piepureza/(piepureza+pienormal+pieferrosa)*100;

```

```

        ferrosatotal =
pieferrosa/(piepureza+pienormal+pieferrosa)*100;
        this->pureza_tb->Text = ((int)purezatotal).ToString();
        this->ferrosa_tb->Text= ((int)ferrosatotal).ToString();
    }

    //Función que actualiza la gráfica de contenido en metal a partir del
data temporal en función de las checkbox marcadas
    void curvas() {
        //zedgraph
        if (this->pot_cb5->Visible==true) {
            this->zedGraphControll->GraphPane->CurveList-
>Clear();

            this->zedGraphControll->Invalidate();
            int i;
            double arrayY[32];
            //Fe zedgraph
            if(this->Fe_cke->Checked==true) {
                i=0;
                LineItem ^myCurve1;
                PointPairList ^list1 = gcnew PointPairList();
                for(i;i<temporal_dgv->Rows->Count-1;i++) {
                    if(Convert::IsDBNull(temporal_dgv-
>Rows[i]->Cells[2]->Value))
                        arrayY[i]=1;
                    else
{arrayY[i]=Convert::ToDouble(temporal_dgv->Rows[i]->Cells[2]->Value);
                        list1->Add(-i, arrayY[i]);
                    }
                }
                myCurve1 = this->zedGraphControll->GraphPane-
>AddCurve( "Fe", list1, Color::Gray, SymbolType::Diamond);
            }
            //Cu zedgraph
            if(this->Cu_cke->Checked==true) {
                i=0;
                LineItem ^myCurve2;
                PointPairList ^list2 = gcnew PointPairList();
                for(i;i<temporal_dgv->Rows->Count-1;i++) {
                    if(Convert::IsDBNull(temporal_dgv-
>Rows[i]->Cells[8]->Value))
                        arrayY[i]=1;
                    else
{arrayY[i]=Convert::ToDouble(temporal_dgv->Rows[i]->Cells[8]->Value);
                        list2->Add(-i, arrayY[i]);
                    }
                }
                myCurve2 = this->zedGraphControll->GraphPane-
>AddCurve( "Cu", list2, Color::Orange, SymbolType::Diamond);
                myCurve2->IsY2Axis=true;
            }
            //Mn zedgraph
            if(this->Mn_cke->Checked==true) {
                i=0;
                LineItem ^myCurve3;

```

```

        PointPairList ^list3 = gcnew PointPairList();
        for(i;i<temporal_dgv->Rows->Count-1;i++) {
            if(Convert::IsDBNull(temporal_dgv-
>Rows[i]->Cells[9]->Value))
                arrayY[i]=1;
            else
{arrayY[i]=Convert::ToDouble(temporal_dgv->Rows[i]->Cells[9]->Value);
                list3->Add(-i, arrayY[i]);
            }
        }
        myCurve3 = this->zedGraphControl1->GraphPane-
>AddCurve( "Mn", list3, Color::Blue, SymbolType::Diamond);
        myCurve3->IsY2Axis=true;
    }
    //Si zedgraph
    if(this->Si_ckb->Checked==true) {
        i=0;
        LineItem ^myCurve4;
        PointPairList ^list4 = gcnew PointPairList();
        for(i;i<temporal_dgv->Rows->Count-1;i++) {
            if(Convert::IsDBNull(temporal_dgv-
>Rows[i]->Cells[10]->Value))
                arrayY[i]=1;
            else
{arrayY[i]=Convert::ToDouble(temporal_dgv->Rows[i]->Cells[10]->Value);
                list4->Add(-i, arrayY[i]);
            }
        }
        myCurve4 = this->zedGraphControl1->GraphPane-
>AddCurve( "Si", list4, Color::Gold, SymbolType::Diamond);
        myCurve4->IsY2Axis=true;
    }
    this->zedGraphControl1->AxisChange();
    this->zedGraphControl1->GraphPane->XAxis->Scale-
>Max=0;
    this->zedGraphControl1->GraphPane->XAxis->Scale-
>Min=-30;
    this->zedGraphControl1->GraphPane->YAxis->Scale-
>Min=0;
    this->zedGraphControl1->GraphPane->YAxis->Scale-
>Max=200;
    this->zedGraphControl1->GraphPane->Y2Axis->Scale-
>Min=0;
    this->zedGraphControl1->GraphPane->Y2Axis->Scale-
>Max=40;
    this->zedGraphControl1->Invalidate();//Necesario
después del AxisChange para que refresque pantalla
        }
    }

    // cosas de datos

    private: menu::TablaGeneral^ TablaGeneral;
    private: System::Windows::Forms::BindingSource^
tablaGeneralBindingSource;

```

```

        private: System::Windows::Forms::BindingSource^
potDayHistBindingSource;
        private: System::Windows::Forms::BindingSource^
Pot_Day_Hist1BindingSource;

        private: System::Windows::Forms::BindingSource^
All_Event_LogBindingSource1;
        private: System::Windows::Forms::BindingSource^
Pot_Shift_HistBindingSource;
        private: menu::TablaGeneralTableAdapters::Pot_Day_HistTableAdapter^
Pot_Day_HistTableAdapter;
        private: menu::TablaGeneralTableAdapters::Pot_Day_Hist1TableAdapter^
Pot_Day_Hist1TableAdapter;

        private: menu::TablaGeneralTableAdapters::All_Event_LogTableAdapter^
All_Event_LogTableAdapter;
        private: menu::TablaGeneralTableAdapters::Pot_Shift_HistTableAdapter^
Pot_Shift_HistTableAdapter;

        private: System::Windows::Forms::BindingSource^
All_Event_Log1BindingSource;
        private: menu::TablaGeneralTableAdapters::All_Event_Log1TableAdapter^
All_Event_Log1TableAdapter;
        private: System::Windows::Forms::BindingSource^
All_Event_Log2BindingSource;
        private: menu::TablaGeneralTableAdapters::All_Event_Log2TableAdapter^
All_Event_Log2TableAdapter;
        private: System::Windows::Forms::BindingSource^
All_Event_Log3BindingSource;
        private: menu::TablaGeneralTableAdapters::All_Event_Log3TableAdapter^
All_Event_Log3TableAdapter;
        private: System::Windows::Forms::BindingSource^
L_Pot_Action_LatestBindingSource;
        private:
menu::TablaGeneralTableAdapters::L_Pot_Action_LatestTableAdapter^
L_Pot_Action_LatestTableAdapter;
        // general dgv
        private: System::Windows::Forms::DataGridView^ general_dgv;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
lineDataGridViewTextBoxColumn;
        private: System::Windows::Forms::DataGridViewTextBoxColumn^
dateTimeDataGridViewTextBoxColumn;
        // controles varios
        private: System::Windows::Forms::DateTimePicker^ dateTimePicker1;
        private: System::Windows::Forms::ComboBox^ pot_cb1;
        private: System::Windows::Forms::ComboBox^ pot_cb2;
        private: System::Windows::Forms::ComboBox^ pot_cb3;
        private: System::Windows::Forms::ComboBox^ pot_cb4;
        private: System::Windows::Forms::ComboBox^ pot_cb5;
        private: System::Windows::Forms::ComboBox^ pot_temp_cb;
        private: System::Windows::Forms::Label^ pot_temp_lb;
        private: System::Windows::Forms::Label^ pot_lb;
        private: System::Windows::Forms::Button^ clearpots_bt;
        private: System::Windows::Forms::Label^ Fe_lab;
        private: System::Windows::Forms::ComboBox^ Fe_cb;
        // temporal dgv
        private: System::Windows::Forms::DataGridView^ temporal_dgv;

```

```

        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
potDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
ageDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalIronFeDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measBathTemperatureControlDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
noisSPPNVoltsDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalPadLevelDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalDASRealDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measBathLevelDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
mEASSTPBATHXSALF3DataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalCopperCuDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalManganeseMnDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalSiliconSiDataGridViewTextBoxColumn;
        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measCathodeDropIncBusDataGridViewTextBoxColumn;

```

```
// AE dgv
```

```

        private:          System::Windows::Forms::DataGridViewTextBoxColumn^
aETimeAbove8VoltsDataGridViewTextBoxColumn;
        private: System::Windows::Forms::TextBox^   textBox1;
        private: System::Windows::Forms::TextBox^   textBox2;
        private: System::Windows::Forms::TextBox^   textBox3;
        private: System::Windows::Forms::TextBox^   textBox4;
        private: System::Windows::Forms::TextBox^   textBox5;
        private: System::Windows::Forms::TextBox^   textBox7;
        private: System::Windows::Forms::TextBox^   textBox8;
        private: System::Windows::Forms::TextBox^   textBox9;
        private: System::Windows::Forms::TextBox^   textBox10;
        private: System::Windows::Forms::TextBox^   textBox11;
        private: System::Windows::Forms::TextBox^   textBox12;
        private: System::Windows::Forms::TextBox^   textBox6;
        private: System::Windows::Forms::Label^     label1;
        private: System::Windows::Forms::Label^     label2;
        private: System::Windows::Forms::TextBox^   textBox13;

```



```

private: System::Windows::Forms::TextBox^  textBox14;
private: System::Windows::Forms::TextBox^  textBox15;
private: System::Windows::Forms::TextBox^  textBox16;
private: System::Windows::Forms::Label^    label3;
private: System::Windows::Forms::Label^    label4;

private: System::Windows::Forms::TextBox^  textBox17;
private: System::Windows::Forms::TextBox^  textBox18;
private: System::Windows::Forms::TextBox^  textBox19;
private: System::Windows::Forms::TextBox^  textBox20;
private: System::Windows::Forms::Label^    label5;
private: System::Windows::Forms::GroupBox^  AE_gb;

private: System::Windows::Forms::DataGridView^  AE_dgv;

private: ZedGraph::ZedGraphControl^  zedGraphControl1;
private: System::Windows::Forms::CheckBox^  Fe_ckb;
private: System::Windows::Forms::CheckBox^  Cu_ckb;
private: System::Windows::Forms::CheckBox^  Mn_ckb;
private: System::Windows::Forms::CheckBox^  Si_ckb;
private: ZedGraph::ZedGraphControl^  zedGraphControl2;
private: System::Windows::Forms::DataGridView^  barras_dgv;
private:
    System::Windows::Forms::BindingSource^
Pot_Day_barrasBindingSource;
private: menu::TablaGeneralTableAdapters::Pot_Day_barrasTableAdapter^
Pot_Day_barrasTableAdapter;
private:
    System::Windows::Forms::DataGridViewTextBoxColumn^
dateTimeDataGridViewTextBoxColumn2;
private:
    System::Windows::Forms::DataGridViewTextBoxColumn^
potDataGridViewTextBoxColumn1;
private:
    System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalIronFeDataGridViewTextBoxColumn2;
private: System::Windows::Forms::RadioButton^  serie2_rb;
private: System::Windows::Forms::RadioButton^  serie3_rb;
private: System::Windows::Forms::RadioButton^  series_rb;

private: System::ComponentModel::IContainer^  components;

private:
    /// <summary>
    /// Required designer variable.
    /// </summary>

#pragma region Windows Form Designer generated code

```

```

void InitializeComponent(void)
{
    this->components = (gcnew
System::ComponentModel::Container());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle8 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle14 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle12 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle13 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle9 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle10 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle11 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle1 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle2 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle3 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle4 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle5 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle6 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    System::Windows::Forms::DataGridViewCellStyle^
dataGridViewCellStyle7 = (gcnew
System::Windows::Forms::DataGridViewCellStyle());
    this->general_dgv = (gcnew
System::Windows::Forms::DataGridView());
    this->dateTimeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
    this->lineDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
    this->dateTimePicker1 = (gcnew
System::Windows::Forms::DateTimePicker());
    this->pot_cb1 = (gcnew
System::Windows::Forms::ComboBox());

```

```

        this->pot_cb2 = (gcnew
System::Windows::Forms::ComboBox());
        this->pot_cb3 = (gcnew
System::Windows::Forms::ComboBox());
        this->pot_cb4 = (gcnew
System::Windows::Forms::ComboBox());
        this->pot_cb5 = (gcnew
System::Windows::Forms::ComboBox());
        this->pot_temp_cb = (gcnew
System::Windows::Forms::ComboBox());
        this->pot_lb = (gcnew System::Windows::Forms::Label());
        this->clearpots_bt = (gcnew
System::Windows::Forms::Button());
        this->Fe_lab = (gcnew System::Windows::Forms::Label());
        this->Fe_cb = (gcnew System::Windows::Forms::ComboBox());
        this->temporal_dgv = (gcnew
System::Windows::Forms::DataGridView());
        this->pot_temp_lb = (gcnew
System::Windows::Forms::Label());
        this->textBox1 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox2 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox3 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox4 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox5 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox7 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox8 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox9 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox10 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox11 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox12 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox6 = (gcnew
System::Windows::Forms::TextBox());
        this->label1 = (gcnew System::Windows::Forms::Label());
        this->label2 = (gcnew System::Windows::Forms::Label());
        this->textBox13 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox14 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox15 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox16 = (gcnew
System::Windows::Forms::TextBox());
        this->label3 = (gcnew System::Windows::Forms::Label());
        this->label4 = (gcnew System::Windows::Forms::Label());
        this->textBox17 = (gcnew
System::Windows::Forms::TextBox());

```

```

        this->textBox18 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox19 = (gcnew
System::Windows::Forms::TextBox());
        this->textBox20 = (gcnew
System::Windows::Forms::TextBox());
        this->label5 = (gcnew System::Windows::Forms::Label());
        this->AE_gb = (gcnew System::Windows::Forms::GroupBox());
        this->AE_dgv = (gcnew
System::Windows::Forms::DataGridView());
        this->Fed_Failure_Int_Day = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Fed_Failure_Ext_Day = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->zedGraphControl1 = (gcnew
ZedGraph::ZedGraphControl());
        this->Fe_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->Cu_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->Mn_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->Si_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->zedGraphControl2 = (gcnew
ZedGraph::ZedGraphControl());
        this->barras_dgv = (gcnew
System::Windows::Forms::DataGridView());
        this->serie2_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->serie3_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->series_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->metal = (gcnew System::Windows::Forms::GroupBox());
        this->titulo_lab = (gcnew
System::Windows::Forms::Label());
        this->diasbarras_cb = (gcnew
System::Windows::Forms::ComboBox());
        this->diasbarras_lab = (gcnew
System::Windows::Forms::Label());
        this->pie_zg = (gcnew ZedGraph::ZedGraphControl());
        this->pureza_tb = (gcnew
System::Windows::Forms::TextBox());
        this->ferrosa_tb = (gcnew
System::Windows::Forms::TextBox());
        this->label6 = (gcnew System::Windows::Forms::Label());
        this->label7 = (gcnew System::Windows::Forms::Label());
        this->print_but = (gcnew
System::Windows::Forms::Button());
        this->barras2_dgv = (gcnew
System::Windows::Forms::DataGridView());
        this->savehist_but = (gcnew
System::Windows::Forms::Button());
        this->saveline_but = (gcnew
System::Windows::Forms::Button());

```

```

        this->potDataGridViewTextBoxColumn2 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalIronFeDataGridViewTextBoxColumn3 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->lineDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Pot_Day_Barras_LatestBindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->TablaGeneral = (gcnew menu::TablaGeneral());
        this->dateTimeDataGridViewTextBoxColumn2 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->potDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalIronFeDataGridViewTextBoxColumn2 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Pot_Day_barrasBindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->dateTimeDataGridViewTextBoxColumn3 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->s1NumDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->s2NumDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->s3NumDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->aENumTotalDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->s1TimeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->s2TimeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->s3TimeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->aETimeTotalDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Pot_Shift_HistBindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->All_Event_Log1BindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->All_Event_LogBindingSource1 = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->All_Event_Log3BindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->L_Pot_Action_LatestBindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->All_Event_Log2BindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->Pot_Day_Hist1BindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->potDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->ageDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalIronFeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());

```

```

        this->measBathTemperatureControlDataGridViewTextBoxColumn
= (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->noisSPPNVoltsDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalPadLevelDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measBathLevelDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalCopperCuDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalManganeseMnDataGridViewTextBoxColumn =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalSiliconSiDataGridViewTextBoxColumn =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measCathodeDropIncBusDataGridViewTextBoxColumn =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->potDayHistBindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->tablaGeneralBindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->Pot_Day_HistTableAdapter = (gcnew
menu::TablaGeneralTableAdapters::Pot_Day_HistTableAdapter());
        this->Pot_Day_Hist1TableAdapter = (gcnew
menu::TablaGeneralTableAdapters::Pot_Day_Hist1TableAdapter());
        this->All_Event_LogTableAdapter = (gcnew
menu::TablaGeneralTableAdapters::All_Event_LogTableAdapter());
        this->All_Event_Log1TableAdapter = (gcnew
menu::TablaGeneralTableAdapters::All_Event_Log1TableAdapter());
        this->All_Event_Log2TableAdapter = (gcnew
menu::TablaGeneralTableAdapters::All_Event_Log2TableAdapter());
        this->All_Event_Log3TableAdapter = (gcnew
menu::TablaGeneralTableAdapters::All_Event_Log3TableAdapter());
        this->L_Pot_Action_LatestTableAdapter = (gcnew
menu::TablaGeneralTableAdapters::L_Pot_Action_LatestTableAdapter());
        this->Pot_Shift_HistTableAdapter = (gcnew
menu::TablaGeneralTableAdapters::Pot_Shift_HistTableAdapter());
        this->Pot_Day_barrasTableAdapter = (gcnew
menu::TablaGeneralTableAdapters::Pot_Day_barrasTableAdapter());
        this->Pot_Day_Barras_LatestTableAdapter = (gcnew
menu::TablaGeneralTableAdapters::Pot_Day_Barras_LatestTableAdapter());
        this->dateTimeDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->ageDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalIronFeDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Bath_Temperature_Control = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->noisSPPNVoltsDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalDASRealDataGridViewTextBoxColumn=(gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalPadLevelDataGridViewTextBoxColumn1 =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

```

```

        this->ELDAS = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measBathLevelDataGridViewTextBoxColumn1 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->MEAS_STP_BATH_XSALF3 = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalCopperCuDataGridViewTextBoxColumn1 =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalManganeseMnDataGridViewTextBoxColumn1 =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalSiliconSiDataGridViewTextBoxColumn1 =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measCathodeDropIncBusDataGridViewTextBoxColumn1 =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>general_dgv)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>temporal_dgv)->BeginInit();
        this->AE_gb->SuspendLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>AE_dgv)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>barras_dgv)->BeginInit();
        this->metal->SuspendLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>barras2_dgv)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>Pot_Day_Barras_LatestBindingSource)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>TablaGeneral)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>Pot_Day_barrasBindingSource)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>Pot_Shift_HistBindingSource)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>All_Event_Log1BindingSource)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>All_Event_LogBindingSource1)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>All_Event_Log3BindingSource)->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>L_Pot_Action_LatestBindingSource)->BeginInit();

```

```

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>All_Event_Log2BindingSource)) ->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>Pot_Day_Hist1BindingSource)) ->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>potDayHistBindingSource)) ->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>tablaGeneralBindingSource)) ->BeginInit();
        this->SuspendLayout();
        //
        // general_dgv
        //
        this->general_dgv->AutoGenerateColumns = false;
        this->general_dgv->ColumnHeadersHeightSizeMode
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
        this->general_dgv->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^ >(13) {this-
>potDataGridViewTextBoxColumn,
        this->ageDataGridViewTextBoxColumn, this-
>measMetalIronFeDataGridViewTextBoxColumn, this-
>measBathTemperatureControlDataGridViewTextBoxColumn,
        this->noisSPPNVoltsDataGridViewTextBoxColumn, this-
>measMetalPadLevelDataGridViewTextBoxColumn, this-
>measMetalDASRealDataGridViewTextBoxColumn, this-
>measBathLevelDataGridViewTextBoxColumn,
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn,
this->measMetalCopperCuDataGridViewTextBoxColumn, this-
>measMetalManganeseMnDataGridViewTextBoxColumn,
        this->measMetalSiliconSiDataGridViewTextBoxColumn,
this->measCathodeDropIncBusDataGridViewTextBoxColumn});
        this->general_dgv->DataSource = this-
>potDayHistBindingSource;
        dataGridViewCellStyle8->Alignment =
System::Windows::Forms::DataGridViewContentAlignment::MiddleCenter;
        dataGridViewCellStyle8->BackColor =
System::Drawing::SystemColors::Window;
        dataGridViewCellStyle8->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular,
        System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
        dataGridViewCellStyle8->ForeColor =
System::Drawing::SystemColors::ControlText;
        dataGridViewCellStyle8->SelectionBackColor =
System::Drawing::SystemColors::Highlight;
        dataGridViewCellStyle8->SelectionForeColor =
System::Drawing::SystemColors::HighlightText;
        dataGridViewCellStyle8->WrapMode =
System::Windows::Forms::DataGridViewTriState::False;
        this->general_dgv->DefaultCellStyle =
dataGridViewCellStyle8;
        this->general_dgv->Location = System::Drawing::Point(106,
57);

```



```

this->general_dgv->Name = L"general_dgv";
this->general_dgv->RowHeadersWidth = 30;
this->general_dgv->Size = System::Drawing::Size(819,
178);

this->general_dgv->TabIndex = 0;
//
// dateTimeDataGridViewTextBoxColumn
//
this->dateTimeDataGridViewTextBoxColumn->DataPropertyName
= L"Date_Time";
this->dateTimeDataGridViewTextBoxColumn->HeaderText =
L"Date";
this->dateTimeDataGridViewTextBoxColumn->Name =
L"dateTimeDataGridViewTextBoxColumn";
this->dateTimeDataGridViewTextBoxColumn->Width = 70;
//
// lineDataGridViewTextBoxColumn
//
this->lineDataGridViewTextBoxColumn->DataPropertyName =
L"Line";
this->lineDataGridViewTextBoxColumn->HeaderText =
L"Line";
this->lineDataGridViewTextBoxColumn->Name =
L"lineDataGridViewTextBoxColumn";
//
// dateTimePicker1
//
this->dateTimePicker1->Location =
System::Drawing::Point(12, 12);
this->dateTimePicker1->Name = L"dateTimePicker1";
this->dateTimePicker1->Size = System::Drawing::Size(200,
20);

this->dateTimePicker1->TabIndex = 1;
this->dateTimePicker1->ValueChanged += gcnw
System::EventHandler(this, &Form2::dateTimePicker1_ValueChanged);
//
// pot_cb1
//
this->pot_cb1->FormattingEnabled = true;
this->pot_cb1->Items->AddRange(gcnw cli::array<
System::Object^
L"206",
L"207", L"208", L"209", L"210", L"211", L"212",
L"213", L"214", L"215", L"216", L"217", L"218", L"219", L"220", L"221",
L"222",
L"223", L"224", L"225", L"226", L"227", L"228",
L"229", L"230", L"231", L"232", L"233", L"234", L"235", L"236", L"237",
L"238",
L"239", L"240", L"241", L"242", L"243", L"244",
L"245", L"246", L"247", L"248", L"249", L"250", L"251", L"252", L"253",
L"254",
L"255", L"256", L"257", L"258", L"259", L"260",
L"261", L"262", L"263", L"264", L"265", L"266", L"267", L"268", L"269",
L"270",
L"271", L"272", L"273", L"274", L"275", L"276",
L"277", L"278", L"279", L"280", L"281", L"282", L"283", L"284", L"285",
L"286",

```

```

L"287", L"288", L"289", L"290", L"291", L"292",
L"293", L"294", L"295", L"296", L"297", L"298", L"299", L"300", L"301",
L"302",
L"303", L"304", L"305", L"306", L"307", L"308",
L"309", L"310", L"311", L"312", L"313", L"314", L"315", L"316", L"317",
L"318",
L"319", L"320", L"321", L"322", L"323", L"324",
L"325", L"326", L"327", L"328", L"329", L"330", L"331", L"332", L"333",
L"334",
L"335", L"336", L"337", L"338", L"339", L"340",
L"341", L"342", L"343", L"344", L"345", L"401", L"402", L"403", L"404",
L"405",
L"406", L"407", L"408", L"409", L"410", L"411",
L"412", L"413", L"414", L"415", L"416", L"417", L"418", L"419", L"420",
L"421",
L"422", L"423", L"424", L"425", L"426", L"427",
L"428", L"429", L"430", L"431", L"432", L"433", L"434", L"435", L"436",
L"437",
L"438", L"439", L"440", L"441", L"442", L"443",
L"444", L"445", L"446", L"447", L"448", L"449", L"450", L"451", L"452",
L"453",
L"454", L"455", L"456", L"457", L"458", L"459",
L"460", L"461", L"462", L"463", L"464", L"465", L"466", L"467", L"468",
L"469",
L"470", L"471", L"472", L"473", L"474", L"475",
L"476", L"477", L"478", L"479", L"480", L"481", L"482", L"483", L"484",
L"485",
L"486", L"487", L"488", L"489", L"490", L"491",
L"492", L"493", L"494", L"495", L"496", L"497", L"498", L"499", L"500",
L"501",
L"502", L"503", L"504", L"505", L"506", L"507",
L"508", L"509", L"510", L"511", L"512", L"513", L"514", L"515", L"516",
L"517",
L"518", L"519", L"520", L"521", L"522", L"523",
L"524", L"525", L"526", L"527", L"528", L"529", L"530", L"531", L"532",
L"533",
L"534", L"535", L"536", L"537", L"538", L"539",
L"540", L"541", L"542", L"543", L"544"});
this->pot_cb1->Location = System::Drawing::Point(26, 72);
this->pot_cb1->Name = L"pot_cb1";
this->pot_cb1->Size = System::Drawing::Size(54, 21);
this->pot_cb1->TabIndex = 2;
this->pot_cb1->SelectedValueChanged += gcnew
System::EventHandler(this, &Form2::pot_cb1_SelectedValueChanged);
//
// pot_cb2
//
this->pot_cb2->FormattingEnabled = true;
this->pot_cb2->Items->AddRange(gcnew cli::array<
System::Object^ >(290) {L"0", L"201", L"202", L"203", L"204", L"205",
L"206",
L"207", L"208", L"209", L"210", L"211", L"212",
L"213", L"214", L"215", L"216", L"217", L"218", L"219", L"220", L"221",
L"222",
L"223", L"224", L"225", L"226", L"227", L"228",
L"229", L"230", L"231", L"232", L"233", L"234", L"235", L"236", L"237",
L"238",

```

```

L"239", L"240", L"241", L"242", L"243", L"244",
L"245", L"246", L"247", L"248", L"249", L"250", L"251", L"252", L"253",
L"254",
L"255", L"256", L"257", L"258", L"259", L"260",
L"261", L"262", L"263", L"264", L"265", L"266", L"267", L"268", L"269",
L"270",
L"271", L"272", L"273", L"274", L"275", L"276",
L"277", L"278", L"279", L"280", L"281", L"282", L"283", L"284", L"285",
L"286",
L"287", L"288", L"289", L"290", L"291", L"292",
L"293", L"294", L"295", L"296", L"297", L"298", L"299", L"300", L"301",
L"302",
L"303", L"304", L"305", L"306", L"307", L"308",
L"309", L"310", L"311", L"312", L"313", L"314", L"315", L"316", L"317",
L"318",
L"319", L"320", L"321", L"322", L"323", L"324",
L"325", L"326", L"327", L"328", L"329", L"330", L"331", L"332", L"333",
L"334",
L"335", L"336", L"337", L"338", L"339", L"340",
L"341", L"342", L"343", L"344", L"345", L"401", L"402", L"403", L"404",
L"405",
L"406", L"407", L"408", L"409", L"410", L"411",
L"412", L"413", L"414", L"415", L"416", L"417", L"418", L"419", L"420",
L"421",
L"422", L"423", L"424", L"425", L"426", L"427",
L"428", L"429", L"430", L"431", L"432", L"433", L"434", L"435", L"436",
L"437",
L"438", L"439", L"440", L"441", L"442", L"443",
L"444", L"445", L"446", L"447", L"448", L"449", L"450", L"451", L"452",
L"453",
L"454", L"455", L"456", L"457", L"458", L"459",
L"460", L"461", L"462", L"463", L"464", L"465", L"466", L"467", L"468",
L"469",
L"470", L"471", L"472", L"473", L"474", L"475",
L"476", L"477", L"478", L"479", L"480", L"481", L"482", L"483", L"484",
L"485",
L"486", L"487", L"488", L"489", L"490", L"491",
L"492", L"493", L"494", L"495", L"496", L"497", L"498", L"499", L"500",
L"501",
L"502", L"503", L"504", L"505", L"506", L"507",
L"508", L"509", L"510", L"511", L"512", L"513", L"514", L"515", L"516",
L"517",
L"518", L"519", L"520", L"521", L"522", L"523",
L"524", L"525", L"526", L"527", L"528", L"529", L"530", L"531", L"532",
L"533",
L"534", L"535", L"536", L"537", L"538", L"539",
L"540", L"541", L"542", L"543", L"544"});
this->pot_cb2->Location = System::Drawing::Point(26, 99);
this->pot_cb2->Name = L"pot_cb2";
this->pot_cb2->Size = System::Drawing::Size(54, 21);
this->pot_cb2->TabIndex = 3;
this->pot_cb2->SelectedValueChanged += gcnew
System::EventHandler(this, &Form2::pot_cb2_SelectedValueChanged);
//
// pot_cb3
//
this->pot_cb3->FormattingEnabled = true;

```

```

this->pot_cb3->Items->AddRange(gcnew cli::array<
System::Object^ >(290) {L"0", L"201", L"202", L"203", L"204", L"205",
L"206",
L"207", L"208", L"209", L"210", L"211", L"212",
L"213", L"214", L"215", L"216", L"217", L"218", L"219", L"220", L"221",
L"222",
L"223", L"224", L"225", L"226", L"227", L"228",
L"229", L"230", L"231", L"232", L"233", L"234", L"235", L"236", L"237",
L"238",
L"239", L"240", L"241", L"242", L"243", L"244",
L"245", L"246", L"247", L"248", L"249", L"250", L"251", L"252", L"253",
L"254",
L"255", L"256", L"257", L"258", L"259", L"260",
L"261", L"262", L"263", L"264", L"265", L"266", L"267", L"268", L"269",
L"270",
L"271", L"272", L"273", L"274", L"275", L"276",
L"277", L"278", L"279", L"280", L"281", L"282", L"283", L"284", L"285",
L"286",
L"287", L"288", L"289", L"290", L"291", L"292",
L"293", L"294", L"295", L"296", L"297", L"298", L"299", L"300", L"301",
L"302",
L"303", L"304", L"305", L"306", L"307", L"308",
L"309", L"310", L"311", L"312", L"313", L"314", L"315", L"316", L"317",
L"318",
L"319", L"320", L"321", L"322", L"323", L"324",
L"325", L"326", L"327", L"328", L"329", L"330", L"331", L"332", L"333",
L"334",
L"335", L"336", L"337", L"338", L"339", L"340",
L"341", L"342", L"343", L"344", L"345", L"401", L"402", L"403", L"404",
L"405",
L"406", L"407", L"408", L"409", L"410", L"411",
L"412", L"413", L"414", L"415", L"416", L"417", L"418", L"419", L"420",
L"421",
L"422", L"423", L"424", L"425", L"426", L"427",
L"428", L"429", L"430", L"431", L"432", L"433", L"434", L"435", L"436",
L"437",
L"438", L"439", L"440", L"441", L"442", L"443",
L"444", L"445", L"446", L"447", L"448", L"449", L"450", L"451", L"452",
L"453",
L"454", L"455", L"456", L"457", L"458", L"459",
L"460", L"461", L"462", L"463", L"464", L"465", L"466", L"467", L"468",
L"469",
L"470", L"471", L"472", L"473", L"474", L"475",
L"476", L"477", L"478", L"479", L"480", L"481", L"482", L"483", L"484",
L"485",
L"486", L"487", L"488", L"489", L"490", L"491",
L"492", L"493", L"494", L"495", L"496", L"497", L"498", L"499", L"500",
L"501",
L"502", L"503", L"504", L"505", L"506", L"507",
L"508", L"509", L"510", L"511", L"512", L"513", L"514", L"515", L"516",
L"517",
L"518", L"519", L"520", L"521", L"522", L"523",
L"524", L"525", L"526", L"527", L"528", L"529", L"530", L"531", L"532",
L"533",
L"534", L"535", L"536", L"537", L"538", L"539",
L"540", L"541", L"542", L"543", L"544"});

```

```

        this->pot_cb3->Location      =      System::Drawing::Point(26,
126);

        this->pot_cb3->Name = L"pot_cb3";
        this->pot_cb3->Size = System::Drawing::Size(54, 21);
        this->pot_cb3->TabIndex = 4;
        this->pot_cb3->SelectedValueChanged      +=      gcnew
System::EventHandler(this, &Form2::pot_cb3_SelectedValueChanged);
        //
        // pot_cb4
        //
        this->pot_cb4->FormattingEnabled = true;
        this->pot_cb4->Items->AddRange(gcnew      cli::array<
System::Object^
L"206",
        L"207", L"208", L"209", L"210", L"211", L"212",
L"213", L"214", L"215", L"216", L"217", L"218", L"219", L"220", L"221",
L"222",
        L"223", L"224", L"225", L"226", L"227", L"228",
L"229", L"230", L"231", L"232", L"233", L"234", L"235", L"236", L"237",
L"238",
        L"239", L"240", L"241", L"242", L"243", L"244",
L"245", L"246", L"247", L"248", L"249", L"250", L"251", L"252", L"253",
L"254",
        L"255", L"256", L"257", L"258", L"259", L"260",
L"261", L"262", L"263", L"264", L"265", L"266", L"267", L"268", L"269",
L"270",
        L"271", L"272", L"273", L"274", L"275", L"276",
L"277", L"278", L"279", L"280", L"281", L"282", L"283", L"284", L"285",
L"286",
        L"287", L"288", L"289", L"290", L"291", L"292",
L"293", L"294", L"295", L"296", L"297", L"298", L"299", L"300", L"301",
L"302",
        L"303", L"304", L"305", L"306", L"307", L"308",
L"309", L"310", L"311", L"312", L"313", L"314", L"315", L"316", L"317",
L"318",
        L"319", L"320", L"321", L"322", L"323", L"324",
L"325", L"326", L"327", L"328", L"329", L"330", L"331", L"332", L"333",
L"334",
        L"335", L"336", L"337", L"338", L"339", L"340",
L"341", L"342", L"343", L"344", L"345", L"401", L"402", L"403", L"404",
L"405",
        L"406", L"407", L"408", L"409", L"410", L"411",
L"412", L"413", L"414", L"415", L"416", L"417", L"418", L"419", L"420",
L"421",
        L"422", L"423", L"424", L"425", L"426", L"427",
L"428", L"429", L"430", L"431", L"432", L"433", L"434", L"435", L"436",
L"437",
        L"438", L"439", L"440", L"441", L"442", L"443",
L"444", L"445", L"446", L"447", L"448", L"449", L"450", L"451", L"452",
L"453",
        L"454", L"455", L"456", L"457", L"458", L"459",
L"460", L"461", L"462", L"463", L"464", L"465", L"466", L"467", L"468",
L"469",
        L"470", L"471", L"472", L"473", L"474", L"475",
L"476", L"477", L"478", L"479", L"480", L"481", L"482", L"483", L"484",
L"485",

```

```

        L"486", L"487", L"488", L"489", L"490", L"491",
L"492", L"493", L"494", L"495", L"496", L"497", L"498", L"499", L"500",
L"501",
        L"502", L"503", L"504", L"505", L"506", L"507",
L"508", L"509", L"510", L"511", L"512", L"513", L"514", L"515", L"516",
L"517",
        L"518", L"519", L"520", L"521", L"522", L"523",
L"524", L"525", L"526", L"527", L"528", L"529", L"530", L"531", L"532",
L"533",
        L"534", L"535", L"536", L"537", L"538", L"539",
L"540", L"541", L"542", L"543", L"544"});
        this->pot_cb4->Location = System::Drawing::Point(26,
153);

        this->pot_cb4->Name = L"pot_cb4";
        this->pot_cb4->Size = System::Drawing::Size(54, 21);
        this->pot_cb4->TabIndex = 6;
        this->pot_cb4->SelectedValueChanged += gcnew
System::EventHandler(this, &Form2::pot_cb4_SelectedValueChanged);
        //
        // pot_cb5
        //
        this->pot_cb5->FormattingEnabled = true;
        this->pot_cb5->Items->AddRange(gcnew cli::array<
System::Object^
L"206",
        L"207", L"208", L"209", L"210", L"211", L"212",
L"213", L"214", L"215", L"216", L"217", L"218", L"219", L"220", L"221",
L"222",
        L"223", L"224", L"225", L"226", L"227", L"228",
L"229", L"230", L"231", L"232", L"233", L"234", L"235", L"236", L"237",
L"238",
        L"239", L"240", L"241", L"242", L"243", L"244",
L"245", L"246", L"247", L"248", L"249", L"250", L"251", L"252", L"253",
L"254",
        L"255", L"256", L"257", L"258", L"259", L"260",
L"261", L"262", L"263", L"264", L"265", L"266", L"267", L"268", L"269",
L"270",
        L"271", L"272", L"273", L"274", L"275", L"276",
L"277", L"278", L"279", L"280", L"281", L"282", L"283", L"284", L"285",
L"286",
        L"287", L"288", L"289", L"290", L"291", L"292",
L"293", L"294", L"295", L"296", L"297", L"298", L"299", L"300", L"301",
L"302",
        L"303", L"304", L"305", L"306", L"307", L"308",
L"309", L"310", L"311", L"312", L"313", L"314", L"315", L"316", L"317",
L"318",
        L"319", L"320", L"321", L"322", L"323", L"324",
L"325", L"326", L"327", L"328", L"329", L"330", L"331", L"332", L"333",
L"334",
        L"335", L"336", L"337", L"338", L"339", L"340",
L"341", L"342", L"343", L"344", L"345", L"401", L"402", L"403", L"404",
L"405",
        L"406", L"407", L"408", L"409", L"410", L"411",
L"412", L"413", L"414", L"415", L"416", L"417", L"418", L"419", L"420",
L"421",

```

```

        L"422", L"423", L"424", L"425", L"426", L"427",
L"428", L"429", L"430", L"431", L"432", L"433", L"434", L"435", L"436",
L"437",
        L"438", L"439", L"440", L"441", L"442", L"443",
L"444", L"445", L"446", L"447", L"448", L"449", L"450", L"451", L"452",
L"453",
        L"454", L"455", L"456", L"457", L"458", L"459",
L"460", L"461", L"462", L"463", L"464", L"465", L"466", L"467", L"468",
L"469",
        L"470", L"471", L"472", L"473", L"474", L"475",
L"476", L"477", L"478", L"479", L"480", L"481", L"482", L"483", L"484",
L"485",
        L"486", L"487", L"488", L"489", L"490", L"491",
L"492", L"493", L"494", L"495", L"496", L"497", L"498", L"499", L"500",
L"501",
        L"502", L"503", L"504", L"505", L"506", L"507",
L"508", L"509", L"510", L"511", L"512", L"513", L"514", L"515", L"516",
L"517",
        L"518", L"519", L"520", L"521", L"522", L"523",
L"524", L"525", L"526", L"527", L"528", L"529", L"530", L"531", L"532",
L"533",
        L"534", L"535", L"536", L"537", L"538", L"539",
L"540", L"541", L"542", L"543", L"544"});
        this->pot_cb5->Location = System::Drawing::Point(26,
180);
        this->pot_cb5->Name = L"pot_cb5";
        this->pot_cb5->Size = System::Drawing::Size(54, 21);
        this->pot_cb5->TabIndex = 7;
        this->pot_cb5->SelectedValueChanged += gcnew
System::EventHandler(this, &Form2::pot_cb5_SelectedValueChanged);
        //
        // pot_temp_cb
        //
        this->pot_temp_cb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 12,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
        this->pot_temp_cb->FormattingEnabled = true;
        this->pot_temp_cb->Items->AddRange(gcnew cli::array<
System::Object^ >(290) {L"0", L"201", L"202", L"203", L"204", L"205",
L"206", L"207", L"208", L"209", L"210", L"211",
L"212", L"213", L"214", L"215", L"216", L"217", L"218", L"219", L"220",
L"221",
L"222", L"223", L"224", L"225", L"226", L"227",
L"228", L"229", L"230", L"231", L"232", L"233", L"234", L"235", L"236",
L"237",
L"238", L"239", L"240", L"241", L"242", L"243",
L"244", L"245", L"246", L"247", L"248", L"249", L"250", L"251", L"252",
L"253",
L"254", L"255", L"256", L"257", L"258", L"259",
L"260", L"261", L"262", L"263", L"264", L"265", L"266", L"267", L"268",
L"269",
L"270", L"271", L"272", L"273", L"274", L"275",
L"276", L"277", L"278", L"279", L"280", L"281", L"282", L"283", L"284",
L"285",

```

```

L"286", L"287", L"288", L"289", L"290", L"291",
L"292", L"293", L"294", L"295", L"296", L"297", L"298", L"299", L"300",
L"301",
L"302", L"303", L"304", L"305", L"306", L"307",
L"308", L"309", L"310", L"311", L"312", L"313", L"314", L"315", L"316",
L"317",
L"318", L"319", L"320", L"321", L"322", L"323",
L"324", L"325", L"326", L"327", L"328", L"329", L"330", L"331", L"332",
L"333",
L"334", L"335", L"336", L"337", L"338", L"339",
L"340", L"341", L"342", L"343", L"344", L"345", L"401", L"402", L"403",
L"404",
L"405", L"406", L"407", L"408", L"409", L"410",
L"411", L"412", L"413", L"414", L"415", L"416", L"417", L"418", L"419",
L"420",
L"421", L"422", L"423", L"424", L"425", L"426",
L"427", L"428", L"429", L"430", L"431", L"432", L"433", L"434", L"435",
L"436",
L"437", L"438", L"439", L"440", L"441", L"442",
L"443", L"444", L"445", L"446", L"447", L"448", L"449", L"450", L"451",
L"452",
L"453", L"454", L"455", L"456", L"457", L"458",
L"459", L"460", L"461", L"462", L"463", L"464", L"465", L"466", L"467",
L"468",
L"469", L"470", L"471", L"472", L"473", L"474",
L"475", L"476", L"477", L"478", L"479", L"480", L"481", L"482", L"483",
L"484",
L"485", L"486", L"487", L"488", L"489", L"490",
L"491", L"492", L"493", L"494", L"495", L"496", L"497", L"498", L"499",
L"500",
L"501", L"502", L"503", L"504", L"505", L"506",
L"507", L"508", L"509", L"510", L"511", L"512", L"513", L"514", L"515",
L"516",
L"517", L"518", L"519", L"520", L"521", L"522",
L"523", L"524", L"525", L"526", L"527", L"528", L"529", L"530", L"531",
L"532",
L"533", L"534", L"535", L"536", L"537", L"538",
L"539", L"540", L"541", L"542", L"543", L"544"});
this->pot_temp_cb->Location =
System::Drawing::Point(1122, 44);
this->pot_temp_cb->Name = L"pot_temp_cb";
this->pot_temp_cb->Size = System::Drawing::Size(75, 28);
this->pot_temp_cb->TabIndex = 13;
this->pot_temp_cb->SelectedValueChanged += gcnw
System::EventHandler(this, &Form2::pot_temp_cb_SelectedValueChanged);
//
// pot_lb
//
this->pot_lb->AutoSize = true;
this->pot_lb->Font = (gcnw
System::Drawing::Font(L"Microsoft Sans Serif", 9,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->pot_lb->Location = System::Drawing::Point(12, 44);
this->pot_lb->Name = L"pot_lb";
this->pot_lb->Size = System::Drawing::Size(88, 15);
this->pot_lb->TabIndex = 5;

```



```

this->pot_lb->Text = L"Cubas Extra:";
//
// clearpots_bt
//
this->clearpots_bt->Location = System::Drawing::Point(26,
212);

this->clearpots_bt->Name = L"clearpots_bt";
this->clearpots_bt->Size = System::Drawing::Size(54, 23);
this->clearpots_bt->TabIndex = 8;
this->clearpots_bt->Text = L"Limpiar";
this->clearpots_bt->UseVisualStyleBackColor = true;
this->clearpots_bt->Click += gcnew
System::EventHandler(this, &Form2::clearpots_bt_Click);
//
// Fe_lab
//
this->Fe_lab->AutoSize = true;
this->Fe_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->Fe_lab->Location = System::Drawing::Point(238, 16);
this->Fe_lab->Name = L"Fe_lab";
this->Fe_lab->Size = System::Drawing::Size(39, 13);
this->Fe_lab->TabIndex = 10;
this->Fe_lab->Text = L"Fe >=";
//
// Fe_cb
//
this->Fe_cb->FormattingEnabled = true;
this->Fe_cb->Items->AddRange(gcnew cli::array<
System::Object^ >(9) {L"0.0", L"0.1", L"0.2", L"0.3", L"0.4", L"0.5",
L"0.6",
L"0.7", L"0.8"});
this->Fe_cb->Location = System::Drawing::Point(278, 12);
this->Fe_cb->Name = L"Fe_cb";
this->Fe_cb->Size = System::Drawing::Size(61, 21);
this->Fe_cb->TabIndex = 11;
this->Fe_cb->SelectedValueChanged += gcnew
System::EventHandler(this, &Form2::Fe_cb_SelectedValueChanged);
//
// temporal_dgv
//
this->temporal_dgv->AutoGenerateColumns = false;
this->temporal_dgv->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
this->temporal_dgv->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^ >(13) {this-
>dateTimeDataGridViewTextBoxColumn1,
this->ageDataGridViewTextBoxColumn1, this-
>measMetalIronFeDataGridViewTextBoxColumn1, this-
>Meas_Bath_Temperature_Control, this-
>noisSPPNVoltsDataGridViewTextBoxColumn1,
this->measMetalPadLevelDataGridViewTextBoxColumn1,
this->ELDAS, this->measBathLevelDataGridViewTextBoxColumn1, this-
>MEAS_STP_BATH_XSALF3,

```

```

        this->measMetalCopperCuDataGridViewTextBoxColumn1,
this->measMetalManganeseMnDataGridViewTextBoxColumn1,          this-
>measMetalSiliconSiDataGridViewTextBoxColumn1,
        this-
>measCathodeDropIncBusDataGridViewTextBoxColumn1});
        this->temporal_dgv->DataMember = L"Pot_Day_Hist1";
        this->temporal_dgv->DataSource =          this-
>Pot_Day_Hist1BindingSource;
        this->temporal_dgv->Location =
System::Drawing::Point(413, 250);
        this->temporal_dgv->Name = L"temporal_dgv";
        this->temporal_dgv->RowHeadersWidth = 30;
        this->temporal_dgv->Size = System::Drawing::Size(845,
148);
        this->temporal_dgv->TabIndex = 12;
        //
        // pot_temp_lb
        //
        this->pot_temp_lb->AutoSize = true;
        this->pot_temp_lb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 12,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->pot_temp_lb->Location = System::Drawing::Point(992,
47);
        this->pot_temp_lb->Name = L"pot_temp_lb";
        this->pot_temp_lb->Size = System::Drawing::Size(124, 20);
        this->pot_temp_lb->TabIndex = 15;
        this->pot_temp_lb->Text = L"Histórico cuba";
        //
        // textBox1
        //
        this->textBox1->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_LogBindingSource1,
L"VAR_09",
        true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation,          nullptr,
L"N0")));
        this->textBox1->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox1->Location = System::Drawing::Point(200,
65);
        this->textBox1->Name = L"textBox1";
        this->textBox1->Size = System::Drawing::Size(34, 20);
        this->textBox1->TabIndex = 17;
        this->textBox1->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox2
        //
        this->textBox2->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_Log3BindingSource,
L"Event_Start_Datetime",
        true)));

```

```

        this->textBox2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox2->Location = System::Drawing::Point(6,
117);

        this->textBox2->Name = L"textBox2";
        this->textBox2->Size = System::Drawing::Size(119, 20);
        this->textBox2->TabIndex = 18;
        this->textBox2->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox3
        //
        this->textBox3->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_Log1BindingSource,
L"Event_Start_Datetime",
        true)));
        this->textBox3->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox3->Location = System::Drawing::Point(6, 39);
        this->textBox3->Name = L"textBox3";
        this->textBox3->Size = System::Drawing::Size(119, 20);
        this->textBox3->TabIndex = 19;
        this->textBox3->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox4
        //
        this->textBox4->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_Log2BindingSource,
L"Event_Start_Datetime",
        true)));
        this->textBox4->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox4->Location = System::Drawing::Point(6, 91);
        this->textBox4->Name = L"textBox4";
        this->textBox4->Size = System::Drawing::Size(119, 20);
        this->textBox4->TabIndex = 20;
        this->textBox4->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox5
        //
        this->textBox5->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_LogBindingSource1,
L"Event_Start_Datetime",
        true)));
        this->textBox5->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox5->Location = System::Drawing::Point(6, 65);

```

```

        this->textBox5->Name = L"textBox5";
        this->textBox5->Size = System::Drawing::Size(119, 20);
        this->textBox5->TabIndex = 21;
        this->textBox5->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox7
        //
        this->textBox7->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_Log2BindingSource,
L"VAR_09",
                                true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation,
                                nullptr,
L"NO"));
        this->textBox7->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
                                static_cast<System::Byte>(0)));
        this->textBox7->Location = System::Drawing::Point(200,
91);

        this->textBox7->Name = L"textBox7";
        this->textBox7->Size = System::Drawing::Size(34, 20);
        this->textBox7->TabIndex = 23;
        this->textBox7->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox8
        //
        this->textBox8->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_Log3BindingSource,
L"VAR_09",
                                true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation,
                                nullptr,
L"NO"));
        this->textBox8->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
                                static_cast<System::Byte>(0)));
        this->textBox8->Location = System::Drawing::Point(200,
117);

        this->textBox8->Name = L"textBox8";
        this->textBox8->Size = System::Drawing::Size(34, 20);
        this->textBox8->TabIndex = 24;
        this->textBox8->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox9
        //
        this->textBox9->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev_Init_Volts",
                                true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation,
                                nullptr,
L"NO"));
        this->textBox9->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

```

```

        static_cast<System::Byte>(0));
        this->textBox9->Location = System::Drawing::Point(240,
65);
        this->textBox9->Name = L"textBox9";
        this->textBox9->Size = System::Drawing::Size(34, 20);
        this->textBox9->TabIndex = 25;
        this->textBox9->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox10
        //
        this->textBox10->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev2_Init_Volts",
        true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0")));
        this->textBox10->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0));
        this->textBox10->Location = System::Drawing::Point(240,
91);
        this->textBox10->Name = L"textBox10";
        this->textBox10->Size = System::Drawing::Size(34, 20);
        this->textBox10->TabIndex = 26;
        this->textBox10->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox11
        //
        this->textBox11->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev3_init_Volts",
        true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0")));
        this->textBox11->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0));
        this->textBox11->Location = System::Drawing::Point(240,
117);
        this->textBox11->Name = L"textBox11";
        this->textBox11->Size = System::Drawing::Size(34, 20);
        this->textBox11->TabIndex = 27;
        this->textBox11->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox12
        //
        this->textBox12->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Init_Volts",
        true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0")));

```

```

        this->textBox12->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox12->Location = System::Drawing::Point(240,
39);

        this->textBox12->Name = L"textBox12";
        this->textBox12->Size = System::Drawing::Size(34, 20);
        this->textBox12->TabIndex = 28;
        this->textBox12->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox6
        //
        this->textBox6->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this->All_Event_Log1BindingSource,
L"VAR_09",
                true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0")));

        this->textBox6->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox6->Location = System::Drawing::Point(200,
39);

        this->textBox6->Name = L"textBox6";
        this->textBox6->Size = System::Drawing::Size(34, 20);
        this->textBox6->TabIndex = 29;
        this->textBox6->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // label1
        //
        this->label1->AutoSize = true;
        this->label1->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->label1->Location = System::Drawing::Point(208, 23);
        this->label1->Name = L"label1";
        this->label1->Size = System::Drawing::Size(17, 13);
        this->label1->TabIndex = 30;
        this->label1->Text = L"M";
        //
        // label2
        //
        this->label2->AutoSize = true;
        this->label2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->label2->Location = System::Drawing::Point(37, 23);
        this->label2->Name = L"label2";
        this->label2->Size = System::Drawing::Size(65, 13);
        this->label2->TabIndex = 31;
        this->label2->Text = L"Date-Time";

```

```

//
// textBox13
//
this->textBox13->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Avg_Voltage",
true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0"));
this->textBox13->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->textBox13->Location = System::Drawing::Point(280,
39);
this->textBox13->Name = L"textBox13";
this->textBox13->Size = System::Drawing::Size(34, 20);
this->textBox13->TabIndex = 32;
this->textBox13->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
//
// textBox14
//
this->textBox14->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev_Avg_Volts",
true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0"));
this->textBox14->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->textBox14->Location = System::Drawing::Point(280,
65);
this->textBox14->Name = L"textBox14";
this->textBox14->Size = System::Drawing::Size(34, 20);
this->textBox14->TabIndex = 33;
this->textBox14->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
//
// textBox15
//
this->textBox15->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev2_Avg_Volts",
true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0"));
this->textBox15->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->textBox15->Location = System::Drawing::Point(280,
91);
this->textBox15->Name = L"textBox15";
this->textBox15->Size = System::Drawing::Size(34, 20);

```

```

        this->textBox15->TabIndex = 34;
        this->textBox15->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox16
        //
        this->textBox16->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev3_Avg_Volts",
        true,
System::Windows::Forms::DataSourceUpdateMode::OnValidation, nullptr,
L"N0"));
        this->textBox16->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox16->Location = System::Drawing::Point(280,
117);

        this->textBox16->Name = L"textBox16";
        this->textBox16->Size = System::Drawing::Size(34, 20);
        this->textBox16->TabIndex = 35;
        this->textBox16->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // label3
        //
        this->label3->AutoSize = true;
        this->label3->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->label3->Location = System::Drawing::Point(251, 23);
        this->label3->Name = L"label3";
        this->label3->Size = System::Drawing::Size(11, 13);
        this->label3->TabIndex = 36;
        this->label3->Text = L"I";
        //
        // label4
        //
        this->label4->AutoSize = true;
        this->label4->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->label4->Location = System::Drawing::Point(289, 23);
        this->label4->Name = L"label4";
        this->label4->Size = System::Drawing::Size(15, 13);
        this->label4->TabIndex = 37;
        this->label4->Text = L"A";
        //
        // textBox17
        //
        this->textBox17->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Time_Above_8V",
        true)));

```



```

        this->textBox17->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox17->Location = System::Drawing::Point(131,
39);

        this->textBox17->Name = L"textBox17";
        this->textBox17->Size = System::Drawing::Size(63, 20);
        this->textBox17->TabIndex = 38;
        this->textBox17->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox18
        //
        this->textBox18->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev3_Time_Above_8V",
        true)));
        this->textBox18->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox18->Location = System::Drawing::Point(131,
117);

        this->textBox18->Name = L"textBox18";
        this->textBox18->Size = System::Drawing::Size(63, 20);
        this->textBox18->TabIndex = 39;
        this->textBox18->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox19
        //
        this->textBox19->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev2_Time_Above_8V",
        true)));
        this->textBox19->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->textBox19->Location = System::Drawing::Point(131,
91);

        this->textBox19->Name = L"textBox19";
        this->textBox19->Size = System::Drawing::Size(63, 20);
        this->textBox19->TabIndex = 40;
        this->textBox19->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // textBox20
        //
        this->textBox20->DataBindings->Add((gcnew
System::Windows::Forms::Binding(L"Text", this-
>L_Pot_Action_LatestBindingSource, L"AE_Prev_Time_Above_8V",
        true)));
        this->textBox20->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

```

```

        static_cast<System::Byte>(0));
this->textBox20->Location = System::Drawing::Point(131,
65);

this->textBox20->Name = L"textBox20";
this->textBox20->Size = System::Drawing::Size(63, 20);
this->textBox20->TabIndex = 41;
this->textBox20->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
//
// label5
//
this->label5->AutoSize = true;
this->label5->Font =
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0));
this->label5->Location = System::Drawing::Point(131, 23);
this->label5->Name = L"label5";
this->label5->Size = System::Drawing::Size(55, 13);
this->label5->TabIndex = 42;
this->label5->Text = L"Duration";
//
// AE_gb
//
this->AE_gb->Controls->Add(this->textBox3);
this->AE_gb->Controls->Add(this->textBox1);
this->AE_gb->Controls->Add(this->label5);
this->AE_gb->Controls->Add(this->textBox2);
this->AE_gb->Controls->Add(this->textBox20);
this->AE_gb->Controls->Add(this->textBox4);
this->AE_gb->Controls->Add(this->textBox19);
this->AE_gb->Controls->Add(this->textBox5);
this->AE_gb->Controls->Add(this->textBox18);
this->AE_gb->Controls->Add(this->textBox7);
this->AE_gb->Controls->Add(this->textBox17);
this->AE_gb->Controls->Add(this->textBox8);
this->AE_gb->Controls->Add(this->label4);
this->AE_gb->Controls->Add(this->textBox9);
this->AE_gb->Controls->Add(this->label3);
this->AE_gb->Controls->Add(this->textBox10);
this->AE_gb->Controls->Add(this->textBox16);
this->AE_gb->Controls->Add(this->textBox11);
this->AE_gb->Controls->Add(this->textBox15);
this->AE_gb->Controls->Add(this->textBox12);
this->AE_gb->Controls->Add(this->textBox14);
this->AE_gb->Controls->Add(this->textBox6);
this->AE_gb->Controls->Add(this->textBox13);
this->AE_gb->Controls->Add(this->label1);
this->AE_gb->Controls->Add(this->label2);
this->AE_gb->Font =
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0));
this->AE_gb->Location = System::Drawing::Point(936, 92);
this->AE_gb->Name = L"AE_gb";
this->AE_gb->Size = System::Drawing::Size(322, 143);
this->AE_gb->TabIndex = 44;

```

```

        this->AE_gb->TabStop = false;
        this->AE_gb->Text = L"Últimos AEs";
        //
        // AE_dgv
        //
        this->AE_dgv->AutoGenerateColumns = false;
        this->AE_dgv->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
        this->AE_dgv->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^ >(11) {this-
>dateTimeDataGridViewTextBoxColumn3,
                this->s1NumDataGridViewTextBoxColumn,
                this-
>s2NumDataGridViewTextBoxColumn,
                this->s3NumDataGridViewTextBoxColumn,
                this->aENumTotalDataGridViewTextBoxColumn1,
                this->s1TimeDataGridViewTextBoxColumn,
                this-
>s2TimeDataGridViewTextBoxColumn,
                this->s3TimeDataGridViewTextBoxColumn,
                this->aETimeTotalDataGridViewTextBoxColumn,
                this->Fed_Failure_Int_Day,
                this-
>Fed_Failure_Ext_Day});
        this->AE_dgv->DataMember = L"Pot_Shift_Hist";
        this->AE_dgv->DataSource =
this-
>Pot_Shift_HistBindingSource;
        this->AE_dgv->Location = System::Drawing::Point(705,
414);
        this->AE_dgv->Name = L"AE_dgv";
        dataGridViewCellStyle14->Alignment =
System::Windows::Forms::DataGridViewContentAlignment::MiddleCenter;
        dataGridViewCellStyle14->BackColor =
System::Drawing::SystemColors::Control;
        dataGridViewCellStyle14->Font =
(gcnew
System::Drawing::Font(L"Microsoft Sans Serif",
8.25F,
System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
        dataGridViewCellStyle14->ForeColor =
System::Drawing::SystemColors::WindowText;
        dataGridViewCellStyle14->SelectionBackColor =
System::Drawing::SystemColors::Highlight;
        dataGridViewCellStyle14->SelectionForeColor =
System::Drawing::SystemColors::HighlightText;
        dataGridViewCellStyle14->WrapMode =
System::Windows::Forms::DataGridViewTriState::True;
        this->AE_dgv->RowHeadersDefaultCellStyle =
dataGridViewCellStyle14;
        this->AE_dgv->RowHeadersWidth = 30;
        this->AE_dgv->RowTemplate->DefaultCellStyle->Alignment =
System::Windows::Forms::DataGridViewContentAlignment::MiddleCenter;
        this->AE_dgv->Size = System::Drawing::Size(553, 150);
        this->AE_dgv->TabIndex = 45;
        //
        // Fed_Failure_Int_Day
        //
        this->Fed_Failure_Int_Day->DataPropertyName =
L"Fed_Failure_Int_Day";
        dataGridViewCellStyle12->BackColor =
System::Drawing::Color::AquaMarine;

```

```

        this->Fed_Failure_Int_Day->DefaultCellStyle =
dataGridViewCellStyle12;
        this->Fed_Failure_Int_Day->HeaderText = L"Fed Int";
        this->Fed_Failure_Int_Day->Name = L"Fed_Failure_Int_Day";
        this->Fed_Failure_Int_Day->Width = 30;
        //
        // Fed_Failure_Ext_Day
        //
        this->Fed_Failure_Ext_Day->>DataPropertyName =
L"Fed_Failure_Ext_Day";
        dataGridViewCellStyle13->BackColor =
System::Drawing::Color::AquaMarine;
        this->Fed_Failure_Ext_Day->DefaultCellStyle =
dataGridViewCellStyle13;
        this->Fed_Failure_Ext_Day->HeaderText = L"Fed Ext";
        this->Fed_Failure_Ext_Day->Name = L"Fed_Failure_Ext_Day";
        this->Fed_Failure_Ext_Day->Width = 30;
        //
        // zedGraphControl1
        //
        this->zedGraphControl1->Location =
System::Drawing::Point(641, 584);
        this->zedGraphControl1->Name = L"zedGraphControl1";
        this->zedGraphControl1->ScrollGrace = 0;
        this->zedGraphControl1->ScrollMaxX = 0;
        this->zedGraphControl1->ScrollMaxY = 0;
        this->zedGraphControl1->ScrollMaxY2 = 0;
        this->zedGraphControl1->ScrollMinX = 0;
        this->zedGraphControl1->ScrollMinY = 0;
        this->zedGraphControl1->ScrollMinY2 = 0;
        this->zedGraphControl1->Size = System::Drawing::Size(617,
374);

        this->zedGraphControl1->TabIndex = 46;
        //
        // Fe_ckb
        //
        this->Fe_ckb->AutoSize = true;
        this->Fe_ckb->Location = System::Drawing::Point(6, 19);
        this->Fe_ckb->Name = L"Fe_ckb";
        this->Fe_ckb->Size = System::Drawing::Size(40, 17);
        this->Fe_ckb->TabIndex = 47;
        this->Fe_ckb->Text = L"Fe";
        this->Fe_ckb->UseVisualStyleBackColor = true;
        this->Fe_ckb->MouseClicked += gcnew
System::Windows::Forms::EventHandler(this, &Form2::Fe_ckb_MouseClick);
        //
        // Cu_ckb
        //
        this->Cu_ckb->AutoSize = true;
        this->Cu_ckb->Location = System::Drawing::Point(6, 43);
        this->Cu_ckb->Name = L"Cu_ckb";
        this->Cu_ckb->Size = System::Drawing::Size(41, 17);
        this->Cu_ckb->TabIndex = 48;
        this->Cu_ckb->Text = L"Cu";
        this->Cu_ckb->UseVisualStyleBackColor = true;
        this->Cu_ckb->MouseClicked += gcnew
System::Windows::Forms::EventHandler(this, &Form2::Cu_ckb_MouseClick);

```

```

//
// Mn_ckb
//
this->Mn_ckb->AutoSize = true;
this->Mn_ckb->Location = System::Drawing::Point(6, 67);
this->Mn_ckb->Name = L"Mn_ckb";
this->Mn_ckb->Size = System::Drawing::Size(43, 17);
this->Mn_ckb->TabIndex = 49;
this->Mn_ckb->Text = L"Mn";
this->Mn_ckb->UseVisualStyleBackColor = true;
this->Mn_ckb->MouseClicked += gcnew
System::Windows::Forms::EventHandler(this, &Form2::Mn_ckb_MouseClick);
//
// Si_ckb
//
this->Si_ckb->AutoSize = true;
this->Si_ckb->Location = System::Drawing::Point(6, 91);
this->Si_ckb->Name = L"Si_ckb";
this->Si_ckb->Size = System::Drawing::Size(37, 17);
this->Si_ckb->TabIndex = 50;
this->Si_ckb->Text = L"Si";
this->Si_ckb->UseVisualStyleBackColor = true;
this->Si_ckb->MouseClicked += gcnew
System::Windows::Forms::EventHandler(this, &Form2::Si_ckb_MouseClick);
//
// zedGraphControl2
//
this->zedGraphControl2->Location =
System::Drawing::Point(15, 414);
this->zedGraphControl2->Name = L"zedGraphControl2";
this->zedGraphControl2->ScrollGrace = 0;
this->zedGraphControl2->ScrollMaxX = 0;
this->zedGraphControl2->ScrollMaxY = 0;
this->zedGraphControl2->ScrollMaxY2 = 0;
this->zedGraphControl2->ScrollMinX = 0;
this->zedGraphControl2->ScrollMinY = 0;
this->zedGraphControl2->ScrollMinY2 = 0;
this->zedGraphControl2->Size = System::Drawing::Size(611,
544);
this->zedGraphControl2->TabIndex = 51;
//
// barras_dgv
//
this->barras_dgv->AutoGenerateColumns = false;
this->barras_dgv->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
this->barras_dgv->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^ >(3) {this-
>dateTimeDataGridViewTextBoxColumn2,
this->potDataGridViewTextBoxColumn1, this-
>measMetalIronFeDataGridViewTextBoxColumn2});
this->barras_dgv->DataSource = this-
>Pot_Day_barrasBindingSource;
this->barras_dgv->Location = System::Drawing::Point(506,
153);
this->barras_dgv->Name = L"barras_dgv";
this->barras_dgv->RowHeadersWidth = 30;

```

```

        this->barras_dgv->Size = System::Drawing::Size(570, 150);
        this->barras_dgv->TabIndex = 52;
        this->barras_dgv->Visible = false;
        //
        // serie2_rb
        //
        this->serie2_rb->AutoSize = true;
        this->serie2_rb->BackColor =
System::Drawing::Color::LightSkyBlue;
        this->serie2_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->serie2_rb->Location = System::Drawing::Point(400,
451);

        this->serie2_rb->Name = L"serie2_rb";
        this->serie2_rb->Size = System::Drawing::Size(65, 17);
        this->serie2_rb->TabIndex = 53;
        this->serie2_rb->TabStop = true;
        this->serie2_rb->Text = L"Serie 2";
        this->serie2_rb->UseVisualStyleBackColor = false;
        this->serie2_rb->Click += gcnew
System::EventHandler(this, &Form2::serie2_rb_Click);
        //
        // serie3_rb
        //
        this->serie3_rb->AutoSize = true;
        this->serie3_rb->BackColor =
System::Drawing::Color::LightSkyBlue;
        this->serie3_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->serie3_rb->Location = System::Drawing::Point(464,
451);

        this->serie3_rb->Name = L"serie3_rb";
        this->serie3_rb->Size = System::Drawing::Size(65, 17);
        this->serie3_rb->TabIndex = 54;
        this->serie3_rb->TabStop = true;
        this->serie3_rb->Text = L"Serie 3";
        this->serie3_rb->UseVisualStyleBackColor = false;
        this->serie3_rb->Click += gcnew
System::EventHandler(this, &Form2::serie3_rb_Click);
        //
        // series_rb
        //
        this->series_rb->AutoSize = true;
        this->series_rb->BackColor =
System::Drawing::Color::LightSkyBlue;
        this->series_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->series_rb->Location = System::Drawing::Point(528,
451);

        this->series_rb->Name = L"series_rb";
        this->series_rb->Size = System::Drawing::Size(86, 17);

```

```

        this->series_rb->TabIndex = 55;
        this->series_rb->TabStop = true;
        this->series_rb->Text = L"Serie 2 y 3";
        this->series_rb->UseVisualStyleBackColor = false;
        this->series_rb->Click += gcnew
System::EventHandler(this, &Form2::series_rb_Click);
        //
        // metal
        //
        this->metal->BackColor =
System::Drawing::Color::MediumSpringGreen;
        this->metal->Controls->Add(this->Fe_cke);
        this->metal->Controls->Add(this->Cu_cke);
        this->metal->Controls->Add(this->Mn_cke);
        this->metal->Controls->Add(this->Si_cke);
        this->metal->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->metal->Location = System::Drawing::Point(641, 462);
        this->metal->Name = L"metal";
        this->metal->Size = System::Drawing::Size(55, 116);
        this->metal->TabIndex = 56;
        this->metal->TabStop = false;
        this->metal->Text = L"Metal";
        //
        // titulo_lab
        //
        this->titulo_lab->AutoSize = true;
        this->titulo_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 14,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->titulo_lab->Location = System::Drawing::Point(452,
12);
        this->titulo_lab->Name = L"titulo_lab";
        this->titulo_lab->Size = System::Drawing::Size(265, 24);
        this->titulo_lab->TabIndex = 57;
        this->titulo_lab->Text = L"Análisis de Cubas Ferrosas";
        //
        // diasbarras_cb
        //
        this->diasbarras_cb->FormattingEnabled = true;
        this->diasbarras_cb->Items->AddRange(gcnew cli::array<
System::Object^ >(4) {L"Último", L"7", L"15", L"30"});
        this->diasbarras_cb->Location =
System::Drawing::Point(67, 423);
        this->diasbarras_cb->Name = L"diasbarras_cb";
        this->diasbarras_cb->Size = System::Drawing::Size(63,
21);
        this->diasbarras_cb->TabIndex = 58;
        this->diasbarras_cb->SelectedValueChanged += gcnew
System::EventHandler(this, &Form2::diasbarras_cb_SelectedValueChanged);
        //
        // diasbarras_lab
        //
        this->diasbarras_lab->AutoSize = true;

```

```

        this->diasbarras_lab->BackColor =
System::Drawing::Color::LightSkyBlue;
        this->diasbarras_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->diasbarras_lab->Location =
System::Drawing::Point(26, 426);
        this->diasbarras_lab->Name = L"diasbarras_lab";
        this->diasbarras_lab->Size = System::Drawing::Size(34,
13);

        this->diasbarras_lab->TabIndex = 59;
        this->diasbarras_lab->Text = L"Días";
        //
        // pie_zg
        //
        this->pie_zg->Location = System::Drawing::Point(12, 250);
        this->pie_zg->Name = L"pie_zg";
        this->pie_zg->ScrollGrace = 0;
        this->pie_zg->ScrollMaxX = 0;
        this->pie_zg->ScrollMaxY = 0;
        this->pie_zg->ScrollMaxY2 = 0;
        this->pie_zg->ScrollMinX = 0;
        this->pie_zg->ScrollMinY = 0;
        this->pie_zg->ScrollMinY2 = 0;
        this->pie_zg->Size = System::Drawing::Size(371, 148);
        this->pie_zg->TabIndex = 60;
        //
        // pureza_tb
        //
        this->pureza_tb->Location = System::Drawing::Point(101,
332);

        this->pureza_tb->Name = L"pureza_tb";
        this->pureza_tb->Size = System::Drawing::Size(39, 20);
        this->pureza_tb->TabIndex = 61;
        this->pureza_tb->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // ferrosa_tb
        //
        this->ferrosa_tb->Location = System::Drawing::Point(100,
358);

        this->ferrosa_tb->Name = L"ferrosa_tb";
        this->ferrosa_tb->Size = System::Drawing::Size(39, 20);
        this->ferrosa_tb->TabIndex = 62;
        this->ferrosa_tb->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
        //
        // label6
        //
        this->label6->AutoSize = true;
        this->label6->BackColor = System::Drawing::Color::Bisque;
        this->label6->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->label6->Location = System::Drawing::Point(34, 335);

```



```

this->label6->Name = L"label6";
this->label6->Size = System::Drawing::Size(59, 13);
this->label6->TabIndex = 63;
this->label6->Text = L"% Pureza";
//
// label7
//
this->label7->AutoSize = true;
this->label7->BackColor = System::Drawing::Color::Bisque;
this->label7->Font
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
      static_cast<System::Byte>(0));
this->label7->Location = System::Drawing::Point(26, 361);
this->label7->Name = L"label7";
this->label7->Size = System::Drawing::Size(68, 13);
this->label7->TabIndex = 64;
this->label7->Text = L"% Ferrosas";
//
// print_but
//
this->print_but->Location = System::Drawing::Point(756,
16);

this->print_but->Name = L"print_but";
this->print_but->Size = System::Drawing::Size(99, 23);
this->print_but->TabIndex = 65;
this->print_but->Text = L"Guardar captura";
this->print_but->UseVisualStyleBackColor = true;
this->print_but->Click += gcnew
System::EventHandler(this, &Form2::print_but_Click);
//
// barras2_dgv
//
this->barras2_dgv->AutoGenerateColumns = false;
this->barras2_dgv->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
this->barras2_dgv->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^
>(4) {this->potDataGridViewTextBoxColumn2,
      this->measMetalIronFeDataGridViewTextBoxColumn3,
this->lineDataGridViewTextBoxColumn1, this->
>resCycleAvgAmpsControlDataGridViewTextBoxColumn});
this->barras2_dgv->DataSource = this->
>Pot_Day_Barras_LatestBindingSource;
this->barras2_dgv->Location = System::Drawing::Point(714,
348);

this->barras2_dgv->Name = L"barras2_dgv";
this->barras2_dgv->Size = System::Drawing::Size(240,
150);

this->barras2_dgv->TabIndex = 66;
this->barras2_dgv->Visible = false;
//
// savehist_but
//
this->savehist_but->Location =
System::Drawing::Point(539, 421);
this->savehist_but->Name = L"savehist_but";

```

```

        this->savehist_but->Size = System::Drawing::Size(75, 23);
        this->savehist_but->TabIndex = 67;
        this->savehist_but->Text = L"Guardar";
        this->savehist_but->UseVisualStyleBackColor = true;
        this->savehist_but->Click += gcnew
System::EventHandler(this, &Form2::savehist_but_Click);
        //
        // saveline_but
        //
        this->saveline_but->Location =
System::Drawing::Point(1176, 593);
        this->saveline_but->Name = L"saveline_but";
        this->saveline_but->Size = System::Drawing::Size(75, 23);
        this->saveline_but->TabIndex = 68;
        this->saveline_but->Text = L"Guardar";
        this->saveline_but->UseVisualStyleBackColor = true;
        this->saveline_but->Click += gcnew
System::EventHandler(this, &Form2::saveline_but_Click);
        //
        // potDataGridViewTextBoxColumn2
        //
        this->potDataGridViewTextBoxColumn2->DataPropertyName =
L"Pot";
        this->potDataGridViewTextBoxColumn2->HeaderText = L"Pot";
        this->potDataGridViewTextBoxColumn2->Name =
L"potDataGridViewTextBoxColumn2";
        //
        // measMetalIronFeDataGridViewTextBoxColumn3
        //
        this->measMetalIronFeDataGridViewTextBoxColumn3-
>DataPropertyName = L"Meas_Metal_Iron_Fe";
        this->measMetalIronFeDataGridViewTextBoxColumn3-
>HeaderText = L"Meas_Metal_Iron_Fe";
        this->measMetalIronFeDataGridViewTextBoxColumn3->Name =
L"measMetalIronFeDataGridViewTextBoxColumn3";
        //
        // lineDataGridViewTextBoxColumn1
        //
        this->lineDataGridViewTextBoxColumn1->DataPropertyName =
L"Line";
        this->lineDataGridViewTextBoxColumn1->HeaderText =
L"Line";
        this->lineDataGridViewTextBoxColumn1->Name =
L"lineDataGridViewTextBoxColumn1";
        //
        // resCycleAvgAmpsControlDataGridViewTextBoxColumn
        //
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn-
>DataPropertyName = L"Res_Cycle_Avg_Amps_Control";
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn-
>HeaderText = L"Res_Cycle_Avg_Amps_Control";
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn-
>Name = L"resCycleAvgAmpsControlDataGridViewTextBoxColumn";
        //
        // Pot_Day_Barras_LatestBindingSource
        //

```

```

        this->Pot_Day_Barras_LatestBindingSource->DataMember =
L"Pot_Day_Barras_Latest";
        this->Pot_Day_Barras_LatestBindingSource->DataSource =
this->TablaGeneral;
        //
        // TablaGeneral
        //
        this->TablaGeneral->DataSetName = L"TablaGeneral";
        this->TablaGeneral->SchemaSerializationMode =
System::Data::SchemaSerializationMode::IncludeSchema;
        //
        // dateTimeDataGridViewTextBoxColumn2
        //
        this->dateTimeDataGridViewTextBoxColumn2-
>DataPropertyName = L"Date_Time";
        this->dateTimeDataGridViewTextBoxColumn2->HeaderText =
L"Date_Time";
        this->dateTimeDataGridViewTextBoxColumn2->Name =
L"dateTimeDataGridViewTextBoxColumn2";
        //
        // potDataGridViewTextBoxColumn1
        //
        this->potDataGridViewTextBoxColumn1->DataPropertyName =
L"Pot";
        this->potDataGridViewTextBoxColumn1->HeaderText = L"Pot";
        this->potDataGridViewTextBoxColumn1->Name =
L"potDataGridViewTextBoxColumn1";
        //
        // measMetalIronFeDataGridViewTextBoxColumn2
        //
        this->measMetalIronFeDataGridViewTextBoxColumn2-
>DataPropertyName = L"Meas_Metal_Iron_Fe";
        this->measMetalIronFeDataGridViewTextBoxColumn2-
>HeaderText = L"Meas_Metal_Iron_Fe";
        this->measMetalIronFeDataGridViewTextBoxColumn2->Name =
L"measMetalIronFeDataGridViewTextBoxColumn2";
        //
        // Pot_Day_barrasBindingSource
        //
        this->Pot_Day_barrasBindingSource->DataMember =
L"Pot_Day_barras";
        this->Pot_Day_barrasBindingSource->DataSource = this-
>TablaGeneral;
        //
        // dateTimeDataGridViewTextBoxColumn3
        //
        this->dateTimeDataGridViewTextBoxColumn3-
>DataPropertyName = L"Date_Time";
        dataGridViewCellStyle9->BackColor =
System::Drawing::Color::Aquamarine;
        this->dateTimeDataGridViewTextBoxColumn3-
>DefaultCellStyle = dataGridViewCellStyle9;
        this->dateTimeDataGridViewTextBoxColumn3->HeaderText =
L"Date";
        this->dateTimeDataGridViewTextBoxColumn3->Name =
L"dateTimeDataGridViewTextBoxColumn3";
        this->dateTimeDataGridViewTextBoxColumn3->Width = 80;

```

```

//
// s1NumDataGridViewTextBoxColumn
//
this->s1NumDataGridViewTextBoxColumn->DataPropertyName =
L"S1_Num";
this->s1NumDataGridViewTextBoxColumn->HeaderText = L"AE
S1";
this->s1NumDataGridViewTextBoxColumn->Name =
L"s1NumDataGridViewTextBoxColumn";
this->s1NumDataGridViewTextBoxColumn->ReadOnly = true;
this->s1NumDataGridViewTextBoxColumn->Width = 30;
//
// s2NumDataGridViewTextBoxColumn
//
this->s2NumDataGridViewTextBoxColumn->DataPropertyName =
L"S2_Num";
this->s2NumDataGridViewTextBoxColumn->HeaderText = L"AE
S2";
this->s2NumDataGridViewTextBoxColumn->Name =
L"s2NumDataGridViewTextBoxColumn";
this->s2NumDataGridViewTextBoxColumn->ReadOnly = true;
this->s2NumDataGridViewTextBoxColumn->Width = 30;
//
// s3NumDataGridViewTextBoxColumn
//
this->s3NumDataGridViewTextBoxColumn->DataPropertyName =
L"S3_Num";
this->s3NumDataGridViewTextBoxColumn->HeaderText = L"AE
S3";
this->s3NumDataGridViewTextBoxColumn->Name =
L"s3NumDataGridViewTextBoxColumn";
this->s3NumDataGridViewTextBoxColumn->ReadOnly = true;
this->s3NumDataGridViewTextBoxColumn->Width = 30;
//
// aEEnumTotalDataGridViewTextBoxColumn1
//
this->aEEnumTotalDataGridViewTextBoxColumn1-
>DataPropertyName = L"AE_Num_Total";
dataGridViewCellStyle10->BackColor =
System::Drawing::Color::Aquamarine;
this->aEEnumTotalDataGridViewTextBoxColumn1-
>DefaultCellStyle = dataGridViewCellStyle10;
this->aEEnumTotalDataGridViewTextBoxColumn1->HeaderText =
L"AE Tot";
this->aEEnumTotalDataGridViewTextBoxColumn1->Name =
L"aEEnumTotalDataGridViewTextBoxColumn1";
this->aEEnumTotalDataGridViewTextBoxColumn1->ReadOnly =
true;
this->aEEnumTotalDataGridViewTextBoxColumn1->Width = 30;
//
// s1TimeDataGridViewTextBoxColumn
//
this->s1TimeDataGridViewTextBoxColumn->DataPropertyName =
L"S1_Time";
this->s1TimeDataGridViewTextBoxColumn->HeaderText = L"AET
S1";

```

```

        this->s1TimeDataGridViewTextBoxColumn->Name =
L"s1TimeDataGridViewTextBoxColumn";
        this->s1TimeDataGridViewTextBoxColumn->ReadOnly = true;
        this->s1TimeDataGridViewTextBoxColumn->Width = 60;
        //
        // s2TimeDataGridViewTextBoxColumn
        //
        this->s2TimeDataGridViewTextBoxColumn->DataPropertyName =
L"S2_Time";
        this->s2TimeDataGridViewTextBoxColumn->HeaderText = L"AET
S2";
        this->s2TimeDataGridViewTextBoxColumn->Name =
L"s2TimeDataGridViewTextBoxColumn";
        this->s2TimeDataGridViewTextBoxColumn->ReadOnly = true;
        this->s2TimeDataGridViewTextBoxColumn->Width = 60;
        //
        // s3TimeDataGridViewTextBoxColumn
        //
        this->s3TimeDataGridViewTextBoxColumn->DataPropertyName =
L"S3_Time";
        this->s3TimeDataGridViewTextBoxColumn->HeaderText = L"AET
S3";
        this->s3TimeDataGridViewTextBoxColumn->Name =
L"s3TimeDataGridViewTextBoxColumn";
        this->s3TimeDataGridViewTextBoxColumn->ReadOnly = true;
        this->s3TimeDataGridViewTextBoxColumn->Width = 60;
        //
        // aETimeTotalDataGridViewTextBoxColumn
        //
        this->aETimeTotalDataGridViewTextBoxColumn-
>DataPropertyName = L"AE_Time_Total";
        dataGridViewCellStyle1->BackColor =
System::Drawing::Color::Aqua;
        this->aETimeTotalDataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle1;
        this->aETimeTotalDataGridViewTextBoxColumn->HeaderText =
L"AET Tot";
        this->aETimeTotalDataGridViewTextBoxColumn->Name =
L"aETimeTotalDataGridViewTextBoxColumn";
        this->aETimeTotalDataGridViewTextBoxColumn->ReadOnly =
true;
        this->aETimeTotalDataGridViewTextBoxColumn->Width = 60;
        //
        // Pot_Shift_HistBindingSource
        //
        this->Pot_Shift_HistBindingSource->DataSource = this-
>TablaGeneral;
        this->Pot_Shift_HistBindingSource->Position = 0;
        //
        // All_Event_Log1BindingSource
        //
        this->All_Event_Log1BindingSource->DataMember =
L"All_Event_Log1";
        this->All_Event_Log1BindingSource->DataSource = this-
>TablaGeneral;
        //
        // All_Event_LogBindingSource1

```

```

//
this->All_Event_LogBindingSource1->DataMember =
L"All_Event_Log";
this->All_Event_LogBindingSource1->DataSource = this-
>TablaGeneral;
//
// All_Event_Log3BindingSource
//
this->All_Event_Log3BindingSource->DataMember =
L"All_Event_Log3";
this->All_Event_Log3BindingSource->DataSource = this-
>TablaGeneral;
//
// L_Pot_Action_LatestBindingSource
//
this->L_Pot_Action_LatestBindingSource->DataMember =
L"L_Pot_Action_Latest";
this->L_Pot_Action_LatestBindingSource->DataSource =
this->TablaGeneral;
//
// All_Event_Log2BindingSource
//
this->All_Event_Log2BindingSource->DataMember =
L"All_Event_Log2";
this->All_Event_Log2BindingSource->DataSource = this-
>TablaGeneral;
//
// Pot_Day_Hist1BindingSource
//
this->Pot_Day_Hist1BindingSource->DataSource = this-
>TablaGeneral;
this->Pot_Day_Hist1BindingSource->Position = 0;
//
// potDataGridViewTextBoxColumn
//
this->potDataGridViewTextBoxColumn->DataPropertyName =
L"Pot";
this->potDataGridViewTextBoxColumn->HeaderText = L"Pot";
this->potDataGridViewTextBoxColumn->Name =
L"potDataGridViewTextBoxColumn";
this->potDataGridViewTextBoxColumn->Width = 50;
//
// ageDataGridViewTextBoxColumn
//
this->ageDataGridViewTextBoxColumn->DataPropertyName =
L"Age";
this->ageDataGridViewTextBoxColumn->HeaderText = L"Age";
this->ageDataGridViewTextBoxColumn->Name =
L"ageDataGridViewTextBoxColumn";
this->ageDataGridViewTextBoxColumn->Width = 60;
//
// measMetalIronFeDataGridViewTextBoxColumn
//
this->measMetalIronFeDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Iron_Fe";
dataGridViewCellStyle1->Format = L"N1";
dataGridViewCellStyle1->NullValue = nullptr;

```

```

        this->measMetalIronFeDataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle1;
        this->measMetalIronFeDataGridViewTextBoxColumn-
>HeaderText = L"Fe";
        this->measMetalIronFeDataGridViewTextBoxColumn->Name =
L"measMetalIronFeDataGridViewTextBoxColumn";
        this->measMetalIronFeDataGridViewTextBoxColumn->Width =
70;
        //
        // measBathTemperatureControlDataGridViewTextBoxColumn
        //
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->DataPropertyName =
L"Meas_Bath_Temperature_Control";
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->HeaderText =
L"Temperature";
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->Name =
L"measBathTemperatureControlDataGridViewTextBoxColumn";
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->Width = 80;
        //
        // noisSPPNVoltsDataGridViewTextBoxColumn
        //
        this->noisSPPNVoltsDataGridViewTextBoxColumn-
>DataPropertyName = L"Nois_SPPN_Volts";
        dataGridViewCellStyle2->Format = L"N0";
        dataGridViewCellStyle2->NullValue = nullptr;
        this->noisSPPNVoltsDataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle2;
        this->noisSPPNVoltsDataGridViewTextBoxColumn->HeaderText
= L"SPPN";
        this->noisSPPNVoltsDataGridViewTextBoxColumn->Name =
L"noisSPPNVoltsDataGridViewTextBoxColumn";
        this->noisSPPNVoltsDataGridViewTextBoxColumn->Width = 65;
        //
        // measMetalPadLevelDataGridViewTextBoxColumn
        //
        this->measMetalPadLevelDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Pad_Level";
        this->measMetalPadLevelDataGridViewTextBoxColumn-
>HeaderText = L"MetalPadLevel";
        this->measMetalPadLevelDataGridViewTextBoxColumn->Name =
L"measMetalPadLevelDataGridViewTextBoxColumn";
        this->measMetalPadLevelDataGridViewTextBoxColumn->Width =
85;
        //
        //measMetalDASRealDataGridViewTextBoxColumn
        //
        this->measMetalDASRealDataGridViewTextBoxColumn-
>DataPropertyName= L"Meas_Metal_DAS_Real";
        this->measMetalDASRealDataGridViewTextBoxColumn-
>HeaderText=L"DAS";
        this->measMetalDASRealDataGridViewTextBoxColumn-
>Name=L"measMetalDASRealDataGridViewTextBoxColumn";

```

```

        this->measMetalDASRealDataGridViewTextBoxColumn-
>Width=85;
        //
        // measBathLevelDataGridViewTextBoxColumn
        //
        this->measBathLevelDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Bath_Level";
        dataGridViewCellStyle3->Format = L"N0";
        dataGridViewCellStyle3->NullValue = nullptr;
        this->measBathLevelDataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle3;
        this->measBathLevelDataGridViewTextBoxColumn->HeaderText
= L"BathLevel";
        this->measBathLevelDataGridViewTextBoxColumn->Name =
L"measBathLevelDataGridViewTextBoxColumn";
        this->measBathLevelDataGridViewTextBoxColumn->Width = 70;
        //
        // mEASSTPBATHXSALF3DataGridViewTextBoxColumn
        //
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn-
>DataPropertyName = L"MEAS_STP_BATH_XSALF3";
        dataGridViewCellStyle4->Format = L"N2";
        dataGridViewCellStyle4->NullValue = nullptr;
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle4;
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn-
>HeaderText = L"ALF3";
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn->Name =
L"mEASSTPBATHXSALF3DataGridViewTextBoxColumn";
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn->Width =
50;
        //
        // measMetalCopperCuDataGridViewTextBoxColumn
        //
        this->measMetalCopperCuDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Copper_Cu";
        dataGridViewCellStyle5->Format = L"N1";
        dataGridViewCellStyle5->NullValue = nullptr;
        this->measMetalCopperCuDataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle5;
        this->measMetalCopperCuDataGridViewTextBoxColumn-
>HeaderText = L"Cu";
        this->measMetalCopperCuDataGridViewTextBoxColumn->Name =
L"measMetalCopperCuDataGridViewTextBoxColumn";
        this->measMetalCopperCuDataGridViewTextBoxColumn->Width =
50;
        //
        // measMetalManganeseMnDataGridViewTextBoxColumn
        //
        this->measMetalManganeseMnDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Manganese_Mn";
        dataGridViewCellStyle6->Format = L"N1";
        dataGridViewCellStyle6->NullValue = nullptr;
        this->measMetalManganeseMnDataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle6;
        this->measMetalManganeseMnDataGridViewTextBoxColumn-
>HeaderText = L"Mn";

```



```

        this->measMetalManganeseMnDataGridViewTextBoxColumn->Name
= L"measMetalManganeseMnDataGridViewTextBoxColumn";
        this->measMetalManganeseMnDataGridViewTextBoxColumn-
>Width = 50;
        //
        // measMetalSiliconSiDataGridViewTextBoxColumn
        //
        this->measMetalSiliconSiDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Silicon_Si";
        dataGridViewCellStyle7->Format = L"N1";
        dataGridViewCellStyle7->NullValue = nullptr;
        this->measMetalSiliconSiDataGridViewTextBoxColumn-
>DefaultCellStyle = dataGridViewCellStyle7;
        this->measMetalSiliconSiDataGridViewTextBoxColumn-
>HeaderText = L"Si";
        this->measMetalSiliconSiDataGridViewTextBoxColumn->Name =
L"measMetalSiliconSiDataGridViewTextBoxColumn";
        this->measMetalSiliconSiDataGridViewTextBoxColumn->Width
= 50;
        //
        // measCathodeDropIncBusDataGridViewTextBoxColumn
        //
        this->measCathodeDropIncBusDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Cathode_Drop_Inc_Bus";
        this->measCathodeDropIncBusDataGridViewTextBoxColumn-
>HeaderText = L"CathodeDrop";
        this->measCathodeDropIncBusDataGridViewTextBoxColumn-
>Name = L"measCathodeDropIncBusDataGridViewTextBoxColumn";
        this->measCathodeDropIncBusDataGridViewTextBoxColumn-
>Width = 80;
        //
        // potDayHistBindingSource
        //
        this->potDayHistBindingSource->DataMember =
L"Pot_Day_Hist";
        this->potDayHistBindingSource->DataSource = this-
>TablaGeneral;
        //
        // tablaGeneralBindingSource
        //
        this->tablaGeneralBindingSource->DataSource = this-
>TablaGeneral;
        this->tablaGeneralBindingSource->Position = 0;
        //
        // Pot_Day_HistTableAdapter
        //
        this->Pot_Day_HistTableAdapter->ClearBeforeFill = true;
        //
        // Pot_Day_Hist1TableAdapter
        //
        this->Pot_Day_Hist1TableAdapter->ClearBeforeFill = true;
        //
        // All_Event_LogTableAdapter
        //
        this->All_Event_LogTableAdapter->ClearBeforeFill = true;
        //
        // All_Event_Log1TableAdapter

```

```

//
this->All_Event_Log1TableAdapter->ClearBeforeFill = true;
//
// All_Event_Log2TableAdapter
//
this->All_Event_Log2TableAdapter->ClearBeforeFill = true;
//
// All_Event_Log3TableAdapter
//
this->All_Event_Log3TableAdapter->ClearBeforeFill = true;
//
// L_Pot_Action_LatestTableAdapter
//
this->L_Pot_Action_LatestTableAdapter->ClearBeforeFill =
true;
//
// Pot_Shift_HistTableAdapter
//
this->Pot_Shift_HistTableAdapter->ClearBeforeFill = true;
//
// Pot_Day_barrasTableAdapter
//
this->Pot_Day_barrasTableAdapter->ClearBeforeFill = true;
//
// Pot_Day_Barras_LatestTableAdapter
//
this->Pot_Day_Barras_LatestTableAdapter->ClearBeforeFill
= true;
//
// dateTimeDataGridViewTextBoxColumn1
//
this->dateTimeDataGridViewTextBoxColumn1-
>DataPropertyName = L"Date_Time";
this->dateTimeDataGridViewTextBoxColumn1->HeaderText =
L"Date";
this->dateTimeDataGridViewTextBoxColumn1->Name =
L"dateTimeDataGridViewTextBoxColumn1";
this->dateTimeDataGridViewTextBoxColumn1->Width = 80;
//
// ageDataGridViewTextBoxColumn1
//
this->ageDataGridViewTextBoxColumn1->DataPropertyName =
L"Age";
this->ageDataGridViewTextBoxColumn1->HeaderText = L"Age";
this->ageDataGridViewTextBoxColumn1->Name =
L"ageDataGridViewTextBoxColumn1";
this->ageDataGridViewTextBoxColumn1->Width = 60;
//
// measMetalIronFeDataGridViewTextBoxColumn1
//
this->measMetalIronFeDataGridViewTextBoxColumn1-
>DataPropertyName = L"Meas_Metal_Iron_Fe";
this->measMetalIronFeDataGridViewTextBoxColumn1-
>HeaderText = L"Fe";
this->measMetalIronFeDataGridViewTextBoxColumn1->Name =
L"measMetalIronFeDataGridViewTextBoxColumn1";

```

```

70;
        this->measMetalIronFeDataGridViewTextBoxColumn1->Width =
//
// Meas_Bath_Temperature_Control
//
        this->Meas_Bath_Temperature_Control->DataPropertyName =
L"Meas_Bath_Temperature_Control";
        this->Meas_Bath_Temperature_Control->HeaderText =
L"Temperature";
        this->Meas_Bath_Temperature_Control->Name =
L"Meas_Bath_Temperature_Control";
        this->Meas_Bath_Temperature_Control->Width = 80;
//
// noisSPPNVoltsDataGridViewTextBoxColumn1
//
        this->noisSPPNVoltsDataGridViewTextBoxColumn1-
>DataPropertyName = L"Nois_SPPN_Volts";
        this->noisSPPNVoltsDataGridViewTextBoxColumn1->HeaderText
= L"SPPN";
        this->noisSPPNVoltsDataGridViewTextBoxColumn1->Name =
L"noisSPPNVoltsDataGridViewTextBoxColumn1";
        this->noisSPPNVoltsDataGridViewTextBoxColumn1->Width =
65;
//
// measMetalPadLevelDataGridViewTextBoxColumn1
//
        this->measMetalPadLevelDataGridViewTextBoxColumn1-
>DataPropertyName = L"Meas_Metal_Pad_Level";
        this->measMetalPadLevelDataGridViewTextBoxColumn1-
>HeaderText = L"MetalPadLevel";
        this->measMetalPadLevelDataGridViewTextBoxColumn1->Name =
L"measMetalPadLevelDataGridViewTextBoxColumn1";
        this->measMetalPadLevelDataGridViewTextBoxColumn1->Width
= 85;
//
// ELDAS
//
        this->ELDAS->DataPropertyName = L"ELDAS";
        this->ELDAS->HeaderText = L"DAS";
        this->ELDAS->Name = L"ELDAS";
//
// measBathLevelDataGridViewTextBoxColumn1
//
        this->measBathLevelDataGridViewTextBoxColumn1-
>DataPropertyName = L"Meas_Bath_Level";
        this->measBathLevelDataGridViewTextBoxColumn1->HeaderText
= L"BathLevel";
        this->measBathLevelDataGridViewTextBoxColumn1->Name =
L"measBathLevelDataGridViewTextBoxColumn1";
        this->measBathLevelDataGridViewTextBoxColumn1->Width =
70;
//
// MEAS_STP_BATH_XSALF3
//
        this->MEAS_STP_BATH_XSALF3->DataPropertyName =
L"MEAS_STP_BATH_XSALF3";
        this->MEAS_STP_BATH_XSALF3->HeaderText = L"AlF3";

```

```

        this->MEAS_STP_BATH_XSALF3->Name =
L"MEAS_STP_BATH_XSALF3";
        this->MEAS_STP_BATH_XSALF3->Width = 50;
        //
        // measMetalCopperCuDataGridViewTextBoxColumn1
        //
        this->measMetalCopperCuDataGridViewTextBoxColumn1-
>DataPropertyName = L"Meas_Metal_Copper_Cu";
        this->measMetalCopperCuDataGridViewTextBoxColumn1-
>HeaderText = L"Cu";
        this->measMetalCopperCuDataGridViewTextBoxColumn1->Name =
L"measMetalCopperCuDataGridViewTextBoxColumn1";
        this->measMetalCopperCuDataGridViewTextBoxColumn1->Width
= 50;
        //
        // measMetalManganeseMnDataGridViewTextBoxColumn1
        //
        this->measMetalManganeseMnDataGridViewTextBoxColumn1-
>DataPropertyName = L"Meas_Metal_Manganese_Mn";
        this->measMetalManganeseMnDataGridViewTextBoxColumn1-
>HeaderText = L"Mn";
        this->measMetalManganeseMnDataGridViewTextBoxColumn1-
>Name = L"measMetalManganeseMnDataGridViewTextBoxColumn1";
        this->measMetalManganeseMnDataGridViewTextBoxColumn1-
>Width = 50;
        //
        // measMetalSiliconSiDataGridViewTextBoxColumn1
        //
        this->measMetalSiliconSiDataGridViewTextBoxColumn1-
>DataPropertyName = L"Meas_Metal_Silicon_Si";
        this->measMetalSiliconSiDataGridViewTextBoxColumn1-
>HeaderText = L"Si";
        this->measMetalSiliconSiDataGridViewTextBoxColumn1->Name
= L"measMetalSiliconSiDataGridViewTextBoxColumn1";
        this->measMetalSiliconSiDataGridViewTextBoxColumn1->Width
= 50;
        //
        // measCathodeDropIncBusDataGridViewTextBoxColumn1
        //
        this->measCathodeDropIncBusDataGridViewTextBoxColumn1-
>DataPropertyName = L"Meas_Cathode_Drop_Inc_Bus";
        this->measCathodeDropIncBusDataGridViewTextBoxColumn1-
>HeaderText = L"CathodeDrop";
        this->measCathodeDropIncBusDataGridViewTextBoxColumn1-
>Name = L"measCathodeDropIncBusDataGridViewTextBoxColumn1";
        this->measCathodeDropIncBusDataGridViewTextBoxColumn1-
>Width = 80;
        //
        // Form2
        //
        this->AutoScaleDimensions = System::Drawing::SizeF(6,
13);
        this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
        this->ClientSize = System::Drawing::Size(1284, 970);
        this->Controls->Add(this->saveline_but);
        this->Controls->Add(this->savehist_but);

```

```

this->Controls->Add(this->barras2_dgv);
this->Controls->Add(this->print_but);
this->Controls->Add(this->label7);
this->Controls->Add(this->label6);
this->Controls->Add(this->ferrosa_tb);
this->Controls->Add(this->pureza_tb);
this->Controls->Add(this->pie_zg);
this->Controls->Add(this->diasbarras_lab);
this->Controls->Add(this->diasbarras_cb);
this->Controls->Add(this->titulo_lab);
this->Controls->Add(this->metal);
this->Controls->Add(this->series_rb);
this->Controls->Add(this->serie3_rb);
this->Controls->Add(this->serie2_rb);
this->Controls->Add(this->barras_dgv);
this->Controls->Add(this->zedGraphControl2);
this->Controls->Add(this->zedGraphControl1);
this->Controls->Add(this->AE_dgv);
this->Controls->Add(this->AE_gb);
this->Controls->Add(this->pot_temp_lb);
this->Controls->Add(this->pot_temp_cb);
this->Controls->Add(this->temporal_dgv);
this->Controls->Add(this->Fe_cb);
this->Controls->Add(this->Fe_lab);
this->Controls->Add(this->clearpots_bt);
this->Controls->Add(this->pot_cb5);
this->Controls->Add(this->pot_cb4);
this->Controls->Add(this->pot_lb);
this->Controls->Add(this->pot_cb3);
this->Controls->Add(this->pot_cb2);
this->Controls->Add(this->pot_cb1);
this->Controls->Add(this->dateTimePicker1);
this->Controls->Add(this->general_dgv);
this->Name = L"Form2";
this->Text = L"Análisis de Cubas Ferrosas";
this->WindowState =
System::Windows::Forms::FormWindowState::Maximized;
this->Load += gcnew System::EventHandler(this,
&Form2::Form2_Load);
this->FormClosing += gcnew
System::Windows::Forms::FormClosingEventHandler(this,
&Form2::Form2_FormClosing);

(cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>general_dgv))->EndInit();

(cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>temporal_dgv))->EndInit();
this->AE_gb->ResumeLayout(false);
this->AE_gb->PerformLayout();

(cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>AE_dgv))->EndInit();

(cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>barras_dgv))->EndInit();
this->metal->ResumeLayout(false);

```

```

        this->metal->PerformLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>barras2_dgv)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>Pot_Day_Barras_LatestBindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>TablaGeneral)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>Pot_Day_barrasBindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>Pot_Shift_HistBindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>All_Event_Log1BindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>All_Event_LogBindingSource1)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>All_Event_Log3BindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>L_Pot_Action_LatestBindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>All_Event_Log2BindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>Pot_Day_Hist1BindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>potDayHistBindingSource)->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>tablaGeneralBindingSource)->EndInit();
        this->ResumeLayout(false);
        this->PerformLayout();
    }
#pragma endregion
    private: System::Void Form2_FormClosing(System::Object^ sender,
System::Windows::Forms::FormClosingEventArgs^ e) {
        // Cancela el cierre de la form cuando se intenta cerrar
        e->Cancel = true;
        // En vez de eso, solo la hace invisible
        this->Visible = false;
    }

    private: System::Void Form2_Load(System::Object^ sender,
System::EventArgs^ e) {

```

```

// TODO: This line of code loads data into the
'TablaGeneral.Pot_Day_Barras_Latest' table. You can move, or remove it, as
needed.
        this->Pot_Day_Barras_LatestTableAdapter-
>Fill(this->TablaGeneral->Pot_Day_Barras_Latest);
        //Inicializa lo referente a la tabla barras y sus
botones
        this->zedGraphControl2->GraphPane->Title-
>Text="Histograma del Fe";
        this->zedGraphControl2->GraphPane->XAxis->Title-
>Text="Contenido en Fe";
        this->zedGraphControl2->GraphPane->YAxis->Title-
>Text="Número de cubas";
        this->zedGraphControl2->GraphPane->Fill=          gcnew
Fill(Color::LightSkyBlue);
        this->zedGraphControl2->GraphPane->Chart->Fill=
gcnew Fill(Color::FromArgb(230, 230, 255));
        this->zedGraphControl2->GraphPane->YAxis-
>MajorGrid->IsVisible= true;
        this->zedGraphControl2->GraphPane->YAxis-
>MinorGrid->IsVisible= true;
        this->series_rb->Checked=true;
        rbflag1=2;
        rbflag2=2;
        this->Pot_Day_barrasTableAdapter->Fill(this-
>TablaGeneral->Pot_Day_barras, this->dateTimePicker1->Value.Date.AddDays(-
15), this->dateTimePicker1->Value.Date, rbflag1, rbflag2);
        this->pie_zg->GraphPane->Fill=          gcnew
Fill(Color::Bisque);
        this->pie_zg->GraphPane->Chart->Fill=          gcnew
Fill(Color::Bisque);
        // Inicializa los combobox a 0 utilizando la
visibilidad como flag para que no salten los eventos
        this->pot_cb5->Visible=false;
        this->diasbarras_cb->SelectedIndex=1;
        this->pot_cb1->SelectedIndex=0;
        this->pot_cb2->SelectedIndex=0;
        this->pot_cb3->SelectedIndex=0;
        this->pot_cb4->SelectedIndex=0;
        this->pot_cb5->SelectedIndex=0;
        this->pot_temp_cb->SelectedIndex=0;
        this->Fe_cb->SelectedIndex=4;
        //inicializa la checkbox del Fe a true
        this->Fe_cke->Checked=true;
        this->pot_cb5->Visible=true;
        //Inicializa cosas de zedgraph de linea
        this->zedGraphControl1->GraphPane->Title-
>Text="Evolución del contenido en metal";
        this->zedGraphControl1->GraphPane->XAxis->Title-
>Text="Días";
        this->zedGraphControl1->GraphPane->YAxis->Title-
>Text="Fe (ppm)";
        this->zedGraphControl1->GraphPane->Y2Axis->Title-
>Text="Cu, Mn, Si (ppm)";
        this->zedGraphControl1->GraphPane->Y2Axis-
>IsVisible=true;

```

```

        this->zedGraphControl1->GraphPane->Fill=          gcnew
Fill(Color::MediumSpringGreen);
        this->zedGraphControl1->GraphPane->Chart->Fill=
gcnew Fill(Color::Aquamarine);
        //Inicializa los combobox de los últimos AE
        this->L_Pot_Action_LatestTableAdapter->Fill(this-
>TablaGeneral->L_Pot_Action_Latest,          System::Convert::ToInt32(this-
>pot_temp_cb->Text));
        this->All_Event_Log3TableAdapter->Fill(this-
>TablaGeneral->All_Event_Log3, this->pot_temp_cb->Text);
        this->All_Event_Log2TableAdapter->Fill(this-
>TablaGeneral->All_Event_Log2, this->pot_temp_cb->Text);
        this->All_Event_Log1TableAdapter->Fill(this-
>TablaGeneral->All_Event_Log1, this->pot_temp_cb->Text);
        this->All_Event_LogTableAdapter->Fill(this-
>TablaGeneral->All_Event_Log, this->pot_temp_cb->Text);
        //Relleno la tabla 'TablaGeneral.Pot_Day_Hist'
        this->Pot_Day_HistTableAdapter->Fill(this-
>TablaGeneral->Pot_Day_Hist,          this->dateTimePicker1->Value.Date,
System::Convert::ToDouble(this->Fe_cb->Text),          this->dateTimePicker1-
>Value.Date,          System::Convert::ToInt32(this->pot_cb1->Text),          this-
>dateTimePicker1->Value.Date,          System::Convert::ToInt32(this->pot_cb2-
>Text),          this->dateTimePicker1->Value.Date,          System::Convert::ToInt32(this-
>pot_cb3->Text),          this->dateTimePicker1->Value.Date,
System::Convert::ToInt32(this->pot_cb4->Text),          this->dateTimePicker1-
>Value.Date, System::Convert::ToInt32(this->pot_cb5->Text));
        this->Pot_Day_Hist1TableAdapter->Fill(this-
>TablaGeneral->Pot_Day_Hist1,          this->dateTimePicker1->Value.Date.AddDays(-
15),          System::Convert::ToInt32(this->pot_temp_cb->Text),          this-
>dateTimePicker1->Value.Date);
        this->Pot_Shift_HistTableAdapter->Fill(this-
>TablaGeneral->Pot_Shift_Hist,          this->dateTimePicker1->Value.Date.AddDays(-
15),          System::Convert::ToInt32(this->pot_temp_cb->Text),          this-
>dateTimePicker1->Value.Date);
    }

    // Actualiza el datagridview general, la gráfica de metales y
el histograma cuando cambia la selección de fecha
    private: System::Void dateTimePicker1_ValueChanged(System::Object^
sender, System::EventArgs^ e) {
        if (this->pot_cb5->Visible==true) {
            this->Pot_Day_HistTableAdapter->Fill(this-
>TablaGeneral->Pot_Day_Hist,          this->dateTimePicker1->Value.Date,
System::Convert::ToDouble(this->Fe_cb->Text),          this->dateTimePicker1-
>Value.Date,          System::Convert::ToInt32(this->pot_cb1->Text),          this-
>dateTimePicker1->Value.Date,          System::Convert::ToInt32(this->pot_cb2-
>Text),          this->dateTimePicker1->Value.Date,          System::Convert::ToInt32(this-
>pot_cb3->Text),          this->dateTimePicker1->Value.Date,
System::Convert::ToInt32(this->pot_cb4->Text),          this->dateTimePicker1-
>Value.Date, System::Convert::ToInt32(this->pot_cb5->Text));
            this->Pot_Day_Hist1TableAdapter->Fill(this-
>TablaGeneral->Pot_Day_Hist1,          this->dateTimePicker1->Value.Date.AddDays(-
30),          System::Convert::ToInt32(this->pot_temp_cb->Text),          this-
>dateTimePicker1->Value.Date);
        }
    }

```



```

        this->Pot_Shift_HistTableAdapter->Fill(this-
>TablaGeneral->Pot_Shift_Hist, this->dateTimePicker1->Value.Date.AddDays(
30), System::Convert::ToInt32(this->pot_temp_cb->Text), this-
>dateTimePicker1->Value.Date);
        curvas();
        histograma();
    }
}

//Actualiza el datagridview general cuando se quitan o añaden
cubas extra mediante los combobox
private: System::Void pot_cb1_SelectedValueChanged(System::Object^
sender, System::EventArgs^ e) {
    if (this->pot_cb5->Visible==true) this-
>Pot_Day_HistTableAdapter->Fill(this->TablaGeneral->Pot_Day_Hist, this-
>dateTimePicker1->Value.Date, System::Convert::ToDouble(this->Fe_cb->Text),
this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb1-
>Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this-
>pot_cb2->Text), this->dateTimePicker1->Value.Date,
System::Convert::ToInt32(this->pot_cb3->Text), this->dateTimePicker1-
>Value.Date, System::Convert::ToInt32(this->pot_cb4->Text), this-
>dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb5-
>Text));
}

private: System::Void pot_cb2_SelectedValueChanged(System::Object^
sender, System::EventArgs^ e) {
    if (this->pot_cb5->Visible==true) this-
>Pot_Day_HistTableAdapter->Fill(this->TablaGeneral->Pot_Day_Hist, this-
>dateTimePicker1->Value.Date, System::Convert::ToDouble(this->Fe_cb->Text),
this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb1-
>Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this-
>pot_cb2->Text), this->dateTimePicker1->Value.Date,
System::Convert::ToInt32(this->pot_cb3->Text), this->dateTimePicker1-
>Value.Date, System::Convert::ToInt32(this->pot_cb4->Text), this-
>dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb5-
>Text));
}

private: System::Void pot_cb3_SelectedValueChanged(System::Object^
sender, System::EventArgs^ e) {
    if (this->pot_cb5->Visible==true) this-
>Pot_Day_HistTableAdapter->Fill(this->TablaGeneral->Pot_Day_Hist, this-
>dateTimePicker1->Value.Date, System::Convert::ToDouble(this->Fe_cb->Text),
this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb1-
>Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this-
>pot_cb2->Text), this->dateTimePicker1->Value.Date,
System::Convert::ToInt32(this->pot_cb3->Text), this->dateTimePicker1-
>Value.Date, System::Convert::ToInt32(this->pot_cb4->Text), this-
>dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb5-
>Text));
}

private: System::Void pot_cb4_SelectedValueChanged(System::Object^
sender, System::EventArgs^ e) {

```

```

        if (this->pot_cb5->Visible==true) this->Pot_Day_HistTableAdapter->Fill(this->TablaGeneral->Pot_Day_Hist, this->dateTimePicker1->Value.Date, System::Convert::ToDouble(this->Fe_cb->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb1->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb2->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb3->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb4->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb5->Text));
    }
    private: System::Void pot_cb5_SelectedValueChanged(System::Object^ sender, System::EventArgs^ e) {
        if (this->pot_cb5->Visible==true) this->Pot_Day_HistTableAdapter->Fill(this->TablaGeneral->Pot_Day_Hist, this->dateTimePicker1->Value.Date, System::Convert::ToDouble(this->Fe_cb->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb1->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb2->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb3->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb4->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb5->Text));
    }
    private: System::Void Fe_cb_SelectedValueChanged(System::Object^ sender, System::EventArgs^ e) {
        if (this->pot_cb5->Visible==true) this->Pot_Day_HistTableAdapter->Fill(this->TablaGeneral->Pot_Day_Hist, this->dateTimePicker1->Value.Date, System::Convert::ToDouble(this->Fe_cb->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb1->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb2->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb3->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb4->Text), this->dateTimePicker1->Value.Date, System::Convert::ToInt32(this->pot_cb5->Text));
    }

    //Botón Limpiar: Deja las combobox a 0 y actualiza la tablageral
    private: System::Void clearpots_bt_Click(System::Object^ sender, System::EventArgs^ e) {
        // Inicializa los combobox a 0 utilizando la visibilidad como flag para que no salten los eventos
        this->pot_cb5->Visible=false;
        this->pot_cb1->SelectedIndex=0;
        this->pot_cb2->SelectedIndex=0;
        this->pot_cb3->SelectedIndex=0;
        this->pot_cb4->SelectedIndex=0;
        this->pot_cb5->SelectedIndex=0;
        this->pot_cb5->Visible=true;
        histograma();
    }

    //Actualiza datagrid temporal, el datagrid de AEs/Fed y los cb de últimos AE cuando cambias la selección de cuba

```

```

private: System::Void pot_temp_cb_SelectedValueChanged(System::Object^
sender, System::EventArgs^ e) {
    if (this->pot_cb5->Visible==true) this-
>Pot_Day_Hist1TableAdapter->Fill(this->TablaGeneral->Pot_Day_Hist1, this-
>dateTimePicker1->Value.Date.AddDays(-30), System::Convert::ToInt32(this-
>pot_temp_cb->Text), this->dateTimePicker1->Value.Date);
    if (this->pot_cb5->Visible==true) {
        this->All_Event_Log3TableAdapter->Fill(this-
>TablaGeneral->All_Event_Log3, this->pot_temp_cb->Text);
        this->All_Event_Log2TableAdapter->Fill(this-
>TablaGeneral->All_Event_Log2, this->pot_temp_cb->Text);
        this->All_Event_Log1TableAdapter->Fill(this-
>TablaGeneral->All_Event_Log1, this->pot_temp_cb->Text);
        this->All_Event_LogTableAdapter->Fill(this-
>TablaGeneral->All_Event_Log, this->pot_temp_cb->Text);
        this->L_Pot_Action_LatestTableAdapter->Fill(this-
>TablaGeneral->L_Pot_Action_Latest, System::Convert::ToInt32(this-
>pot_temp_cb->Text));
        this->Pot_Shift_HistTableAdapter->Fill(this-
>TablaGeneral->Pot_Shift_Hist, this->dateTimePicker1->Value.Date.AddDays(-
30), System::Convert::ToInt32(this->pot_temp_cb->Text), this-
>dateTimePicker1->Value.Date);
    }
    curvas();
}

//Se recalcula la gráfica con las pulsaciones en las checkbox
de los metales
private: System::Void Fe_ckb_MouseClick(System::Object^ sender,
System::Windows::Forms::MouseEventEventArgs^ e) {
    curvas();
}
private: System::Void Cu_ckb_MouseClick(System::Object^ sender,
System::Windows::Forms::MouseEventEventArgs^ e) {
    curvas();
}
private: System::Void Mn_ckb_MouseClick(System::Object^ sender,
System::Windows::Forms::MouseEventEventArgs^ e) {
    curvas();
}
private: System::Void Si_ckb_MouseClick(System::Object^ sender,
System::Windows::Forms::MouseEventEventArgs^ e) {
    curvas();
}

//Emparejado de selectores de serie para que la activación de
uno desactive el resto
//Al hacer una selección se recalcula el histograma
private: System::Void serie2_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3_rb->Checked=false;
    this->series_rb->Checked=false;
    rbflag1=2;
    rbflag2=2;
    histograma();
}

```

```

private: System::Void serie3_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2_rb->Checked=false;
    this->series_rb->Checked=false;
    rbflag1=3;
    rbflag2=3;
    histograma();
}
private: System::Void series_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2_rb->Checked=false;
    this->serie3_rb->Checked=false;
    rbflag1=2;
    rbflag2=3;
    histograma();
}
//Cuando se cambia el valor del selector de número de días
para la media del histograma, se recalcula el histograma
private: System::Void
diasbarras_cb_SelectedValueChanged(System::Object^ sender,
System::EventArgs^ e) {
    histograma();
}

//Imprimir captura de pantalla
private: System::Void print_but_Click(System::Object^ sender,
System::EventArgs^ e) {
    SaveFileDialog^ dialog = gcnew SaveFileDialog(); //Se
genera un diálogo de guardado
    dialog->Title = "Guardar captura de pantalla"; //título
del diálogo de guardado
    dialog->ShowDialog(); //Se abre un diálogo de guardado
    if (dialog->FileName != "") //Si se ha escrito un nombre
    {
        Bitmap^ captura; //Declaro el objeto captura para
almacenar el pantallazo
        Rectangle^ rectangulo; //Declaro el objeto
rectángulo para el pantallazo
        captura = gcnew Bitmap(this->Width, this->Height);
//Inicializo el bitmap al tamaño de la form
        rectangulo = gcnew Rectangle(0, 0, this->Width,
this->Height); //Inicializo el rectángulo del pantallazo
        this->DrawToBitmap(captura, *rectangulo); //Saco
el pantallazo y lo almaceno en captura
        captura->Save(String::Concat(dialog->FileName,
".png"), System::Drawing::ImageFormat::Bmp); //Guardo el archivo
con el nombre que se le puso en el diálogo añadiendo .png al final. Además
asigno ese tipo de archivo
    }
}

//Imprimir el histograma
private: System::Void savehist_but_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->zedGraphControl2->SaveAs();
}
//Imprimir el gráfico de metales

```

```

private: System::Void saveline_but_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->zedGraphControl1->SaveAs();
}
};
}

```

## 2.3 Form3.h – Grupos

Contiene el código de la ventana de análisis de grupos.

```

#pragma once

using namespace ZedGraph;
using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;

namespace menu {

public ref class Form3 : public System::Windows::Forms::Form
{
public:
    Form3(void)
    {
        InitializeComponent();
        //
        //TODO: Add the constructor code here
        //
    }

protected:

    ~Form3()
    {
        if (components)
        {
            delete components;
        }
    }

private: System::Windows::Forms::ComboBox^ var1_cb;
protected:
private: System::Windows::Forms::ComboBox^ var2_cb;
private: System::Windows::Forms::GroupBox^ filtrado_gb2;
private: System::Windows::Forms::GroupBox^ lateral_gb2;
private: System::Windows::Forms::Label^ lateralS2_lab2;
private: System::Windows::Forms::Label^ lateralS3_lab2;
private: System::Windows::Forms::CheckBox^ s3par_ckb2;
private: System::Windows::Forms::CheckBox^ s3impar_ckb2;
private: System::Windows::Forms::CheckBox^ s2par_ckb2;

```

```

private: System::Windows::Forms::CheckBox^ s2impar_cke2;
private: System::Windows::Forms::GroupBox^ fase_gb2;
private: System::Windows::Forms::Label^ faseS2_lab2;
private: System::Windows::Forms::Label^ faseS3_lab2;
private: System::Windows::Forms::CheckBox^ s2f1_cke2;
private: System::Windows::Forms::CheckBox^ s2f2_cke2;
private: System::Windows::Forms::CheckBox^ s3f2_cke2;
private: System::Windows::Forms::CheckBox^ s3f1_cke2;
private: System::Windows::Forms::GroupBox^ grupos_gb2;
private: System::Windows::Forms::Label^ grupoS2_lab2;
private: System::Windows::Forms::Label^ grupoS3_lab2;
private: System::Windows::Forms::CheckBox^ s2a1_cke2;
private: System::Windows::Forms::CheckBox^ s2b1_cke2;
private: System::Windows::Forms::CheckBox^ s3c2_cke2;
private: System::Windows::Forms::CheckBox^ s2a2_cke2;
private: System::Windows::Forms::CheckBox^ s3b2_cke2;
private: System::Windows::Forms::CheckBox^ s3a1_cke2;
private: System::Windows::Forms::CheckBox^ s3c1_cke2;
private: System::Windows::Forms::CheckBox^ s3b1_cke2;
private: System::Windows::Forms::RadioButton^ serie23_rb2;
private: System::Windows::Forms::RadioButton^ serie3_rb2;
private: System::Windows::Forms::RadioButton^ serie2_rb2;
private: System::Windows::Forms::NumericUpDown^ edaddesde_nud;
private: System::Windows::Forms::NumericUpDown^ edadhasta_nud;
private: System::Windows::Forms::Label^ edadhasta_lab;
private: System::Windows::Forms::Label^ edaddesde_lab;
private: System::Windows::Forms::Label^ edad_lab;
private: System::Windows::Forms::Label^ edad_lab2;
private: System::Windows::Forms::Label^ edadhasta_lab2;
private: System::Windows::Forms::Label^ edaddesde_lab2;
private: System::Windows::Forms::NumericUpDown^ edadhasta_nud2;
private: System::Windows::Forms::NumericUpDown^ edaddesde_nud2;
private: menu::TablaGeneral^ TablaGeneral;
private: System::Windows::Forms::BindingSource^
comparativaBindingSource;
private: menu::TablaGeneralTableAdapters::comparativaTableAdapter^
comparativaTableAdapter;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
dateTimeDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
noisSPNVoltsDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
mEASSTPBATHXSALF3DataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
aETimeAbove8VoltsDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
tappingWeightScheduleDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalIronFeDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalLithiumLiDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
measBathTemperatureControlDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
fedFailuresDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
resCycleAvgAmpsControlDataGridViewTextBoxColumn;

```

```

private:          System::Windows::Forms::DataGridViewTextBoxColumn^
resCycleAvgVoltsDataGridViewTextBoxColumn;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measBathLevelDataGridViewTextBoxColumn;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalPadLevelDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^ Age;
private:          System::Windows::Forms::DataGridViewTextBoxColumn^
potDataGridViewTextBoxColumn;
private: System::Windows::Forms::Button^ draw_but;
private: System::Windows::Forms::Label^ alerta1_lab;
private: System::Windows::Forms::Button^ print_but;
private: System::Windows::Forms::Button^ print2_but;

```

```

//Flag que controla la inhibición de eventos en momentos clave
bool flagnoeventos;

```

```

//Función para obtener el array
void getarray(){

```

```

    if (this->hasta_dtp->Value>=this->desde_dtp->Value) {
//Solo funciona si hasta es mayor o igual que desde
//Declaraciones e historias
int A=this->var1_cb->SelectedIndex; //Variable que
se usa para elegir la columna del dgv de la que se toman los datos
int counter=0;
int posicion_dia=0;

this->principal_zgc->GraphPane->CurveList->Clear();
this->principal_zgc->Invalidate();
// Declaro e inicializo el array de strings para el
eje x y el array de valores del eje y
array<String^>^ labels = gcnew
array<String^>((this->hasta_dtp->Value.Date-this->desde_dtp-
>Value.Date).TotalDays+1); //La longitud del array es la diferencia de
días entre desde y hasta +1
array<double>^ var = gcnew array<double>((this-
>hasta_dtp->Value.Date-this->desde_dtp->Value.Date).TotalDays+1); //La
longitud del array es la diferencia de días entre desde y hasta +1
for (int i=0;i<((this->hasta_dtp->Value.Date-this-
>desde_dtp->Value.Date).TotalDays+1);i++) {
    labels[i]= this->desde_dtp-
>Value.Date.AddDays(i).Day.ToString(); //A la fecha desde se le suma la
posición del vector y se almacena
    var[i]=double::NaN; //El array se inicializa a
NaN
}
//Empieza el algoritmo
for (int i=0;i<this->interface_dgv->Rows->Count-
1;i++) { //Desde la primera a la última línea del dgv
    if ( //Condiciones de filtrado

```

```

//Por selección activa
(A>0) &&
//Filtrado por grupo de agujas
((this->s2a1_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=201 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=225) ||
(this->s2b1_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=226 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=245) ||
(this->s2a2_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=246 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=297) ||
(this->s3a1_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=401 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=426) ||
(this->s3b1_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=427 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=456) ||
(this->s3c1_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=457 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=484) ||
(this->s3b2_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=485 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=512) ||
(this->s3c2_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=513 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=544)) &&
//Filtrado por fase
((this->s2f1_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=201 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=273) ||
(this->s2f2_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=274 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=297) ||
(this->s3f1_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=401 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=472) ||
(this->s3f2_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=473 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=544)) &&
//Filtrado por pares e impares
((this->s2par_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==0 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=300) ||
(this->s2impar_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=297) ||
(this->s3par_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==0 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=301) ||
(this->s3impar_ckb->Checked==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==1 &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=301)) &&
//Filtrado por edad

```



```

(Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[13]->Value)>=this->edaddesde_nud->Value) &&
(Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[13]->Value)<=this->edadhasta_nud->Value)
    ) {
        //Para las variables directas
        if (A!=4 && A!=9 && A!=5) {
            if (this->interface_dgv->Rows[i]->Cells[A]->Value != DBNull::Value) { //Si el valor de la casilla no es null
                if (counter==0)
                    var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[A]->Value); //Si es la primera cuba, su valor se añade al array
                else
                    var[posicion_dia]=(double)var[posicion_dia]+Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[A]->Value); //Si NO es la primera cuba, el valor de la variable se suma a todos los del mismo día de las cubas válidas
                counter++; //Se incrementa el contador de valores en 1
            }
            else for(int j=i;j>=0;j--) //Si el valor de la casilla es null, se retrocede buscando el último valor válido de esa cuba
                if(Convert::ToInt32(this->interface_dgv->Rows[j]->Cells[14]->Value) == Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]->Cells[A]->Value != DBNull::Value) {
                    if (counter==0)
                        var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[j]->Cells[A]->Value); //Si es la primera cuba, su valor se añade al array
                    else
                        var[posicion_dia]=(double)var[posicion_dia]+Convert::ToDouble(this->interface_dgv->Rows[j]->Cells[A]->Value); //Si NO es la primera cuba, el valor de la variable se suma a todos los del mismo día de las cubas válidas
                    counter++; //Se incrementa el contador de valores en 1
                    break; //Dejo de retroceder en el dgv, o terminaré sobrescribiendo el valor tomado por otro más antiguo
                }
            //Para el faraday
            else if (A==4) {
                if (this->interface_dgv->Rows[i]->Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[i]->Cells[9]->Value != DBNull::Value) {
                    if (counter==0)
                        var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[4]->Value)*1000/Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[9]->Value)/8.06;
                    else
                        var[posicion_dia]= (double)var[posicion_dia]+Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[4]->Value)*1000/Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[9]->Value)/8.06;
                    counter++; //Se incrementa el contador de valores en 1
                }
            }
        }
    }

```

```

    }
    else for (int j=i;j>=0;j--) //Si
el valor de la intensidad o la colada es null, se retrocede buscando el
último valor válido de esa cuba
        if(Convert::ToInt32(this-
>interface_dgv->Rows[j]->Cells[14]->Value) == Convert::ToInt32(this-
>interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]-
>Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[j]-
>Cells[9]->Value != DBNull::Value) { //Si tiene el mismo número de cuba y
ni la intensidad ni la colada están en blanco
            if (counter==0)
var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[j]-
>Cells[4]->Value)*1000/Convert::ToDouble(this->interface_dgv->Rows[j]-
>Cells[9]->Value)/8.06;
            else
var[posicion_dia]= (double)var[posicion_dia]+Convert::ToDouble(this-
>interface_dgv->Rows[j]->Cells[4]->Value)*1000/Convert::ToDouble(this-
>interface_dgv->Rows[j]->Cells[9]->Value)/8.06;
            counter++;
            break;
        }
    }
    //Para el CEE
    else if(A==9) {
        if (this->interface_dgv->Rows[i]-
>Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[i]-
>Cells[9]->Value != DBNull::Value && Convert::ToInt32(this->interface_dgv-
>Rows[i]->Cells[4]->Value) != 0) {
            if (counter==0)
var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[10]->Value)*Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[9]->Value)*24/1000/Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[4]->Value);
            else var[posicion_dia]=
(double)var[posicion_dia]+Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[10]->Value)*Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[9]->Value)*24/1000/Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[4]->Value);
            counter++; //Se incrementa
el contador de valores en 1
        }
    }
    else for (int j=i;j>=0;j--) //Si
el valor de la intensidad o la colada es null, se retrocede buscando el
último valor válido de esa cuba
        if(Convert::ToInt32(this-
>interface_dgv->Rows[j]->Cells[14]->Value) == Convert::ToInt32(this-
>interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]-
>Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[j]-
>Cells[9]->Value != DBNull::Value && Convert::ToInt32(this->interface_dgv-
>Rows[j]->Cells[4]->Value) != 0) { //Si tiene el mismo número de cuba y ni
la intensidad ni la colada están en blanco
            if (counter==0)
var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[j]-
>Cells[10]->Value)*Convert::ToDouble(this->interface_dgv->Rows[j]-
>Cells[9]->Value)*24/1000/Convert::ToDouble(this->interface_dgv->Rows[j]-
>Cells[4]->Value);

```

```

else
var[posicion_dia]=          (double)var[posicion_dia]+Convert::ToDouble(this-
>interface_dgv->Rows[j]->Cells[10]->Value)*Convert::ToDouble(this-
>interface_dgv->Rows[j]->Cells[9]->Value)*24/1000/Convert::ToDouble(this-
>interface_dgv->Rows[j]->Cells[4]->Value);
                                counter++;
                                break;
                                }
                                }
                                //Para el %pureza
                                else if(A==5) {
                                    if (this->interface_dgv->Rows[i]-
>Cells[5]->Value != DBNull::Value) {
                                        if(counter==0) {
                                            if(Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[5]->Value)<0.1)
                                            var[posicion_dia]=1; else var[posicion_dia]=0; }
                                        else
                                        {if(Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[5]->Value)<0.109)
                                        var[posicion_dia]++; }
                                        counter++;
                                    }
                                    else for(int j=i;j>=0;j--)
                                        if(Convert::ToInt32(this-
>interface_dgv->Rows[j]->Cells[14]->Value) == Convert::ToInt32(this-
>interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]-
>Cells[5]->Value != DBNull::Value) {
                                            if(counter==0) {
                                                if(Convert::ToDouble(this->interface_dgv->Rows[j]->Cells[5]->Value)<0.1)
                                                var[posicion_dia]=1; else var[posicion_dia]=0; }
                                                else
                                                {if(Convert::ToDouble(this->interface_dgv->Rows[j]->Cells[5]->Value)<0.109)
                                                var[posicion_dia]++; }
                                                counter++;
                                                break;
                                            }
                                        }
                                    }
                                }
                                //Se culmina la media al avanzar de día
                                if(this->interface_dgv->Rows[i]->Cells[0]-
>Value->Equals(this->interface_dgv->Rows[i+1]->Cells[0]->Value)==0) { //Si
la columna siguiente ya pertenece a otro día:
                                    if(counter!=0)          var[posicion_dia]=
(double)var[posicion_dia]/counter; // Si se guardó al menos un valor de ese
día, se hace la media dividiendo el total entre el número de valores
                                    else          if(posicion_dia>0          &&
var[posicion_dia-1]!=double::NaN) var[posicion_dia]=var[posicion_dia-1];
                                    counter=0; //Se resetea el contador de
valores
                                    posicion_dia++;          //Se avanza una
posición en el array
                                }
                                }

//Representación
LineItem ^myCurve1;

```

```

        myCurve1      =      this->principal_zgc->GraphPane-
>AddCurve("Grupo 1", nullptr, var, Color::OrangeRed, SymbolType::Circle);
        myCurve1->Line->Width= 3; //Fijo el ancho de linea
        myCurve1->Symbol->Fill=  gcnew  Fill(Color::Black);
//Fijo el color de símbolo
        myCurve1->Symbol->Size=6;  //Fijo  el  tamaño  de
símbolo
        this->principal_zgc->GraphPane->XAxis->Scale-
>TextLabel=labels;
        this->principal_zgc->GraphPane->XAxis->Type=
AxisType::Text;
        this->principal_zgc->AxisChange();
    }
}

//Getarray para el segundo conjunto de datos
void getarray2(){
    if (this->hasta_dtp->Value>=this->desde_dtp->Value) {
//Solo funciona si hasta es mayor o igual que desde
//Declaraciones e historias
        int A=this->var2_cb->SelectedIndex; //Variable que
se usa para elegir la columna del dgv de la que se toman los datos
        int counter=0;
        int posicion_dia=0;

//this->principal_zgc->GraphPane->CurveList-
>Clear();
//this->principal_zgc->Invalidate();
// Declaro e inicializo el array de strings para el
eje x y el array de valores del eje y
        array<String^>^ labels = gcnew
array<String^>((this->hasta_dtp->Value.Date-this->desde_dtp-
>Value.Date).TotalDays+1); //La longitud del array es la diferencia de
días entre desde y hasta +1
        array<double>^ var = gcnew array<double>((this-
>hasta_dtp->Value.Date-this->desde_dtp->Value.Date).TotalDays+1); //La
longitud del array es la diferencia de días entre desde y hasta +1
        for (int i=0;i<((this->hasta_dtp->Value.Date-this-
>desde_dtp->Value.Date).TotalDays+1);i++) {
            labels[i]= this->desde_dtp-
>Value.Date.AddDays(i).Day.ToString(); //A la fecha desde se le suma la
posición del vector y se almacena
            var[i]=double::NaN; //El array se inicializa a
NaN
        }
//Empieza el algoritmo
        for (int i=0;i<this->interface_dgv->Rows->Count-
1;i++) { //Desde la primera a la última línea del dgv
            if ( //Condiciones de filtrado
                //Por selección activa
                (A>0) &&
                //Filtrado por grupo de agujas

```

```

        ((this->s2a1_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=201    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=225) ||
        (this->s2b1_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=226    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=245) ||
        (this->s2a2_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=246    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=297) ||
        (this->s3a1_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=401    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=426) ||
        (this->s3b1_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=427    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=456) ||
        (this->s3c1_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=457    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=484) ||
        (this->s3b2_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=485    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=512) ||
        (this->s3c2_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=513    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=544)) &&
        //Filtrado por fase
        ((this->s2f1_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=201    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=273) ||
        (this->s2f2_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=274    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=297) ||
        (this->s3f1_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=401    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=472) ||
        (this->s3f2_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=473    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=544)) &&
        //Filtrado por pares e impares
        ((this->s2par_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==0    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=300) ||
        (this->s2impar_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)<=297) ||
        (this->s3par_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==0    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=301) ||
        (this->s3impar_cke2->Checked==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)%2==1    &&
Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value)>=301)) &&
        //Filtrado por edad
        (Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[13]->Value)>=this->edaddesde_nud2->Value) &&
        (Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[13]->Value)<=this->edadhasta_nud2->Value)
    ) {

```

```

//Para las variables directas
if (A!=4 && A!=9 && A!=5) {
    if (this->interface_dgv->Rows[i]-
>Cells[A]->Value != DBNull::Value) { //Si el valor de la casilla no es
null
        if (counter==0)
var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[A]->Value); //Si es la primera cuba, su valor se añade al array
        else
var[posicion_dia]=(double)var[posicion_dia]+Convert::ToDouble(this-
>interface_dgv->Rows[i]->Cells[A]->Value); //Si NO es la primera cuba, el
valor de la variable se suma a todos los del mismo día de las cubas válidas
        counter++; //Se incrementa
el contador de valores en 1
    }
    else for(int j=i;j>=0;j--) //Si el
valor de la casilla es null, se retrocede buscando el último valor válido
de esa cuba
        if(Convert::ToInt32(this-
>interface_dgv->Rows[j]->Cells[14]->Value) == Convert::ToInt32(this-
>interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]-
>Cells[A]->Value != DBNull::Value) {
            if (counter==0)
var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[j]-
>Cells[A]->Value); //Si es la primera cuba, su valor se añade al array
            else
var[posicion_dia]=(double)var[posicion_dia]+Convert::ToDouble(this-
>interface_dgv->Rows[j]->Cells[A]->Value); //Si NO es la primera cuba, el
valor de la variable se suma a todos los del mismo día de las cubas válidas
            counter++; //Se
incrementa el contador de valores en 1
        }
        break; //Dejo de
retroceder en el dgv, o terminaré sobrescribiendo el valor tomado por otro
más antiguo
    }
}
//Para el faraday
else if (A==4) {
    if (this->interface_dgv->Rows[i]-
>Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[i]-
>Cells[9]->Value != DBNull::Value) {
        if (counter==0)
var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[4]->Value)*1000/Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[9]->Value)/8.06;
        else var[posicion_dia]=
(double)var[posicion_dia]+Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[4]->Value)*1000/Convert::ToDouble(this->interface_dgv->Rows[i]-
>Cells[9]->Value)/8.06;
        counter++; //Se incrementa
el contador de valores en 1
    }
    else for (int j=i;j>=0;j--) //Si
el valor de la intensidad o la colada es null, se retrocede buscando el
último valor válido de esa cuba

```

```

        if(Convert::ToInt32(this->interface_dgv->Rows[j]->Cells[14]->Value)
        == Convert::ToInt32(this->interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]-
        >Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[j]-
        >Cells[9]->Value != DBNull::Value) { //Si tiene el mismo número de cuba y
        ni la intensidad ni la colada están en blanco
            if (counter==0)
                var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[j]-
                >Cells[4]->Value)*1000/Convert::ToDouble(this->interface_dgv->Rows[j]-
                >Cells[9]->Value)/8.06;
            else
                var[posicion_dia]= (double)var[posicion_dia]+Convert::ToDouble(this-
                >interface_dgv->Rows[j]->Cells[4]->Value)*1000/Convert::ToDouble(this-
                >interface_dgv->Rows[j]->Cells[9]->Value)/8.06;
                counter++;
                break;
        }
    }
    //Para el CEE
    else if(A==9) {
        if (this->interface_dgv->Rows[i]-
        >Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[i]-
        >Cells[9]->Value != DBNull::Value && Convert::ToInt32(this->interface_dgv-
        >Rows[i]->Cells[4]->Value) != 0) {
            if (counter==0)
                var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[i]-
                >Cells[10]->Value)*Convert::ToDouble(this->interface_dgv->Rows[i]-
                >Cells[9]->Value)*24/1000/Convert::ToDouble(this->interface_dgv->Rows[i]-
                >Cells[4]->Value);
            else var[posicion_dia]=
            (double)var[posicion_dia]+Convert::ToDouble(this->interface_dgv->Rows[i]-
            >Cells[10]->Value)*Convert::ToDouble(this->interface_dgv->Rows[i]-
            >Cells[9]->Value)*24/1000/Convert::ToDouble(this->interface_dgv->Rows[i]-
            >Cells[4]->Value);
            counter++; //Se incrementa
            el contador de valores en 1
        }
    }
    else for (int j=i;j>=0;j--) //Si
    el valor de la intensidad o la colada es null, se retrocede buscando el
    último valor válido de esa cuba
        if(Convert::ToInt32(this-
        >interface_dgv->Rows[j]->Cells[14]->Value)
        == Convert::ToInt32(this-
        >interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]-
        >Cells[4]->Value != DBNull::Value && this->interface_dgv->Rows[j]-
        >Cells[9]->Value != DBNull::Value && Convert::ToInt32(this->interface_dgv-
        >Rows[j]->Cells[4]->Value) != 0) { //Si tiene el mismo número de cuba y ni
        la intensidad ni la colada están en blanco
            if (counter==0)
                var[posicion_dia]= Convert::ToDouble(this->interface_dgv->Rows[j]-
                >Cells[10]->Value)*Convert::ToDouble(this->interface_dgv->Rows[j]-
                >Cells[9]->Value)*24/1000/Convert::ToDouble(this->interface_dgv->Rows[j]-
                >Cells[4]->Value);
            else
                var[posicion_dia]= (double)var[posicion_dia]+Convert::ToDouble(this-
                >interface_dgv->Rows[j]->Cells[10]->Value)*Convert::ToDouble(this-
                >interface_dgv->Rows[j]->Cells[9]->Value)*24/1000/Convert::ToDouble(this-
                >interface_dgv->Rows[j]->Cells[4]->Value);
        }
    }

```

```

        counter++;
        break;
    }
}
//Para el %pureza
else if(A==5) {
    if (this->interface_dgv->Rows[i]-
>Cells[5]->Value != DBNull::Value) {
        if(counter==0) {
            if(Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[5]->Value)<0.1)
            var[posicion_dia]=1; else var[posicion_dia]=0; }
            else
            {if(Convert::ToDouble(this->interface_dgv->Rows[i]->Cells[5]->Value)<0.1)
            var[posicion_dia]++; }
            counter++;
        }
        else for(int j=i;j>=0;j--)
            if(Convert::ToInt32(this-
>interface_dgv->Rows[j]->Cells[14]->Value) == Convert::ToInt32(this-
>interface_dgv->Rows[i]->Cells[14]->Value) && this->interface_dgv->Rows[j]-
>Cells[5]->Value != DBNull::Value) {
                if(counter==0) {
                    if(Convert::ToDouble(this->interface_dgv->Rows[j]->Cells[5]->Value)<0.1)
                    var[posicion_dia]=1; else var[posicion_dia]=0; }
                    else
                    {if(Convert::ToDouble(this->interface_dgv->Rows[j]->Cells[5]->Value)<0.1)
                    var[posicion_dia]++; }
                    counter++;
                    break;
                }
            }
        }
        //Se culmina la media al avanzar de día
        if(this->interface_dgv->Rows[i]->Cells[0]-
>Value->Equals(this->interface_dgv->Rows[i+1]->Cells[0]->Value)==0) { //Si
la columna siguiente ya pertenece a otro día:
            if(counter!=0) var[posicion_dia]=
(double)var[posicion_dia]/counter; // Si se guardó al menos un valor de ese
día, se hace la media dividiendo el total entre el número de valores
            else if(posicion_dia>0 &&
var[posicion_dia-1]!=double::NaN) var[posicion_dia]=var[posicion_dia-1];
            counter=0; //Se resetea el contador de
valores
            posicion_dia++; //Se avanza una
posición en el array
        }
    }
}
//Representación
LineItem ^myCurve1;

myCurve1 = this->principal_zgc->GraphPane-
>AddCurve("Grupo 2", nullptr, var, Color::YellowGreen, SymbolType::Circle);
myCurve1->Line->Width= 3; //Fijo el grosor de linea
myCurve1->Symbol->Fill= gcnew Fill(Color::Black);
//Fijo el color de símbolo

```



```

myCurve1->Symbol->Size=6; //Fijo el tamaño de
símbolo
myCurve1->IsY2Axis=true;

//Quito los ticks del eje principal en el
secundario y viceversa
//this->principal_zgc->GraphPane->Y2Axis->MajorTic-
>IsInside=false;
//this->principal_zgc->GraphPane->Y2Axis->MinorTic-
>IsInside=false;
this->principal_zgc->GraphPane->Y2Axis->MajorTic-
>IsOpposite=false;
this->principal_zgc->GraphPane->Y2Axis->MinorTic-
>IsOpposite=false;

//this->principal_zgc->GraphPane->YAxis->MajorTic-
>IsInside=false;
//this->principal_zgc->GraphPane->YAxis->MinorTic-
>IsInside=false;
this->principal_zgc->GraphPane->YAxis->MajorTic-
>IsOpposite=false;
this->principal_zgc->GraphPane->YAxis->MinorTic-
>IsOpposite=false;

this->principal_zgc->GraphPane->Y2Axis-
>IsVisible=true;
this->principal_zgc->GraphPane->XAxis->Scale-
>TextLabel=labels; //Coloco el array labels en el eje x
this->principal_zgc->GraphPane->XAxis->Type=
AxisType::Text; //Y cambio el tipo de eje a eje de texto
this->principal_zgc->AxisChange(); //Reescalo los
ejes al tamaño de la curva
    }
}

void ejes() {

    //Cambio los títulos de los ejes
    this->principal_zgc->GraphPane->YAxis->Title->Text=this-
>var1_cb->Text;
    this->principal_zgc->GraphPane->Y2Axis->Title->Text=this-
>var2_cb->Text;

    //Fijo el eje principal a valores razonables y fijos(para
mantener referencias)
    switch(this->var1_cb->SelectedIndex) {
        case 1: { //SPPN
            this->principal_zgc->GraphPane->YAxis->Scale-
>Min=80;
            this->principal_zgc->GraphPane->YAxis->Scale-
>Max=180;

            break;
        }
        case 2: { //ALF3
            this->principal_zgc->GraphPane->YAxis->Scale-
>Min=6.5;

```

```

        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=9.5;
        break;
    }
    case 3: { //Tiempo AE
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=0;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=0.7;
        break;
    }
    case 4: { //Faraday
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=0.82;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=0.95;
        break;
    }
    case 5: { //Pureza
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=0.4;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=0.8;
        break;
    }
    case 6: { //Li
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=1.5;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=2.9;
        break;
    }
    case 7: { //Temperatura
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=936;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=956;
        break;
    }
    case 8: { //Fed/dia
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=0;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=6;
        break;
    }
    case 9: { //CEE
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=13;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=18;
        break;
    }
    case 10: { //Volts
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=4.3;

```

```

        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=4.6;
        break;
    }
    case 11: { //Altura baño
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=17;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=20;
        break;
    }
    case 12: { //Altura metal
        this->principal_zgc->GraphPane->YAxis->Scale-
>Min=36;
        this->principal_zgc->GraphPane->YAxis->Scale-
>Max=38;
        break;
    }
}

//Fijo el eje secundario a valores razonables y
fijos(para mantener referencias)
switch(this->var2_cb->SelectedIndex) {
    case 1: { //SPPN
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=80;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=180;
        break;
    }
    case 2: { //ALF3
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=6.5;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=9.5;
        break;
    }
    case 3: { //Tiempo AE
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=0;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=0.7;
        break;
    }
    case 4: { //Faraday
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=0.82;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=0.95;
        break;
    }
    case 5: { //Pureza
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=0.4;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=0.8;
        break;
    }
}

```

```

    }
    case 6: { //Li
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=1.5;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=2.9;
        break;
    }
    case 7: { //Temperatura
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=936;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=956;
        break;
    }
    case 8: { //Fed/dia
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=0;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=6;
        break;
    }
    case 9: { //CEE
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=13;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=18;
        break;
    }
    case 10: { //Volts
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=4.3;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=4.6;
        break;
    }
    case 11: { //Altura baño
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=17;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=20;
        break;
    }
    case 12: { //Altura metal
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Min=36;
        this->principal_zgc->GraphPane->Y2Axis->Scale-
>Max=38;
        break;
    }
}
}
}

```

```

//Función que controla la activación y desactivación de los rb
de las series
//cuando se activan y desactivan las checkbox de serie 2 de la
columna principal
void serie2(bool casilla) {
    if (this->flagnoeventos==false) { //Solo se ejecuta con
flagnoeventos en false
        if (casilla==1) { //Si al clickar una checkbox,
ésta queda activada:
            if (this->s3a1_ckb->Checked==1 ||
                this->s3b1_ckb->Checked==1 ||
                this->s3c1_ckb->Checked==1 ||
                this->s3b2_ckb->Checked==1 ||
                this->s3c2_ckb->Checked==1 ||
                this->s3f1_ckb->Checked==1 ||
                this->s3f2_ckb->Checked==1 ||
                this->s3par_ckb->Checked==1 ||
                this->s3impar_ckb->Checked==1) this->
serie23_rb->Checked=1; //y solo había checkbox de serie 3 activadas, se
activa el rb de series 2 y 3
            else if((this->serie2_rb->Checked==0 &&
                this->serie3_rb->Checked==0 &&
                this->serie23_rb->Checked==0) ||
                this->serie3_rb->Checked==1) this->
serie2_rb->Checked=1; //y no había ninguna checkbox activada antes, se
activa el rb de serie 2
            }
            else if (this->s2a1_ckb->Checked==0 && //Si al
clickar la checkbox ésta queda desactivada:
                this->s2b1_ckb->Checked==0 &&
                this->s2a2_ckb->Checked==0 &&
                this->s2f1_ckb->Checked==0 &&
                this->s2f2_ckb->Checked==0 &&
                this->s2par_ckb->Checked==0 &&
                this->s2impar_ckb->Checked==0) { //y ya
no queda ninguna más activada,
                if (this->serie2_rb->Checked==1)
this->serie2_rb->Checked=0; //si no quedan tampoco checkbox de serie 3, se
desactiva el rb de serie 2 dejándolos todos desactivados
                else this->serie3_rb->Checked=1;
//si aún quedan checkbox en serie 3, se cambia rb de series 2 y 3 a solo 3
            }
        }
    }
}

//Función que controla la activación y desactivación de los rb
de las series
//cuando se activan y desactivan las checkbox de serie 3 de la
columna principal
void serie3(bool casilla) {
    if (this->flagnoeventos==false) { //Solo se ejecuta con
flagnoeventos en false
        if (casilla==1) { //Si al clickar una checkbox,
ésta queda activada:
            if (this->s2a1_ckb->Checked==1 ||
                this->s2b1_ckb->Checked==1 ||

```

```

        this->s2a2_cke->Checked==1 ||
        this->s2f1_cke->Checked==1 ||
        this->s2f2_cke->Checked==1 ||
        this->s2par_cke->Checked==1 ||
        this->s2impar_cke->Checked==1) this-
>serie23_rb->Checked=1; //y solo había checkbox de serie 2 activadas, se
activa el rb de series 2 y 3
        else if((this->serie2_rb->Checked==0 &&
        this->serie3_rb->Checked==0 &&
        this->serie23_rb->Checked==0) ||
        this->serie2_rb->Checked==1) this-
>serie3_rb->Checked=1; //y no había ninguna checkbox activada antes, se
activa el rb de serie 3
    }
    else if (this->s3a1_cke->Checked==0 && //Si al
clickar la checkbox ésta queda desactivada:
        this->s3b1_cke->Checked==0 &&
        this->s3c1_cke->Checked==0 &&
        this->s3b2_cke->Checked==0 &&
        this->s3c2_cke->Checked==0 &&
        this->s3f1_cke->Checked==0 &&
        this->s3f2_cke->Checked==0 &&
        this->s3par_cke->Checked==0 &&
        this->s3impar_cke->Checked==0) { //y ya
no queda ninguna más activada,
        if (this->serie3_rb->Checked==1)
this->serie3_rb->Checked=0; //si no quedan tampoco checkbox de serie 2, se
desactiva el rb de serie 3 dejándolos todos desactivados
        else this->serie2_rb->Checked=1;
//si aún quedan checkbox en serie 2, se cambia rb de series2 y 3 a solo 2
    }
}
}

//Función que controla la activación y desactivación de los rb
de las series
//cuando se activan y desactivan las checkbox de serie 2 de la
columna secundaria
void serie2sec(bool casilla) {
    if (this->flagnoeventos==false) { //Solo se ejecuta con
flagnoeventos en false
        if (casilla==1) { //Si al clickar una checkbox,
ésta queda activada:
            if (this->s3a1_cke2->Checked==1 ||
                this->s3b1_cke2->Checked==1 ||
                this->s3c1_cke2->Checked==1 ||
                this->s3b2_cke2->Checked==1 ||
                this->s3c2_cke2->Checked==1 ||
                this->s3f1_cke2->Checked==1 ||
                this->s3f2_cke2->Checked==1 ||
                this->s3par_cke2->Checked==1 ||
                this->s3impar_cke2->Checked==1) this-
>serie23_rb2->Checked=1; //y solo había checkbox de serie 3 activadas, se
activa el rb de series 2 y 3
            else if((this->serie2_rb2->Checked==0 &&
                this->serie3_rb2->Checked==0 &&
                this->serie23_rb2->Checked==0) ||

```

```

        this->serie3_rb2->Checked==1)        this-
>serie2_rb2->Checked=1; //y no había ninguna checkbox activada antes, se
activa el rb de serie 2
    }
    else if (this->s2a1_ckb2->Checked==0 && //Si al
clickar la checkbox ésta queda desactivada:
        this->s2b1_ckb2->Checked==0 &&
        this->s2a2_ckb2->Checked==0 &&
        this->s2f1_ckb2->Checked==0 &&
        this->s2f2_ckb2->Checked==0 &&
        this->s2par_ckb2->Checked==0 &&
        this->s2impar_ckb2->Checked==0) { //y ya
no queda ninguna más activada,
        if (this->serie2_rb2->Checked==1)
this->serie2_rb2->Checked=0; //si no quedan tampoco checkbox de serie 3, se
desactiva el rb de serie 2 dejándolos todos desactivados
        else this->serie3_rb2->Checked=1;
//si aún quedan checkbox en serie 3, se cambia rb de series 2 y 3 a solo 3
    }
}

//Función que controla la activación y desactivación de los rb
de las series
//cuando se activan y desactivan las checkbox de serie 3 de la
columna secundaria
void serie3sec(bool casilla) {
    if (this->flagnoeventos==false) { //Solo se ejecuta con
flagnoeventos en false
        if (casilla==1) { //Si al clickar una checkbox,
ésta queda activada:
            if (this->s2a1_ckb2->Checked==1 ||
                this->s2b1_ckb2->Checked==1 ||
                this->s2a2_ckb2->Checked==1 ||
                this->s2f1_ckb2->Checked==1 ||
                this->s2f2_ckb2->Checked==1 ||
                this->s2par_ckb2->Checked==1 ||
                this->s2impar_ckb2->Checked==1) this-
>serie23_rb2->Checked=1; //y solo había checkbox de serie 2 activadas, se
activa el rb de series 2 y 3
            else if((this->serie2_rb2->Checked==0 &&
                this->serie3_rb2->Checked==0 &&
                this->serie23_rb2->Checked==0) ||
                this->serie2_rb2->Checked==1) this-
>serie3_rb2->Checked=1; //y no había ninguna checkbox activada antes, se
activa el rb de serie 3
            }
            else if (this->s3a1_ckb2->Checked==0 && //Si al
clickar la checkbox ésta queda desactivada:
                this->s3b1_ckb2->Checked==0 &&
                this->s3c1_ckb2->Checked==0 &&
                this->s3b2_ckb2->Checked==0 &&
                this->s3c2_ckb2->Checked==0 &&
                this->s3f1_ckb2->Checked==0 &&
                this->s3f2_ckb2->Checked==0 &&
                this->s3par_ckb2->Checked==0 &&

```

```

        this->s3impar_ckb2->Checked==0) { //y ya
no queda ninguna más activada,
        if (this->serie3_rb2->Checked==1)
this->serie3_rb2->Checked=0; //si no quedan tampoco checkbox de serie 2, se
desactiva el rb de serie 3 dejándolos todos desactivados
        else this->serie2_rb2->Checked=1;
//si aún quedan checkbox en serie 2, se cambia rb de series2 y 3 a solo 2
    }
}
}

```

```

private: System::Windows::Forms::DateTimePicker^ desde_dtp;
private: System::Windows::Forms::DateTimePicker^ hasta_dtp;
protected:
protected:
private: System::Windows::Forms::Label^ desde_lab;
private: System::Windows::Forms::Label^ hasta_lab;
private: System::Windows::Forms::RadioButton^ serie2_rb;
private: System::Windows::Forms::RadioButton^ serie3_rb;
private: System::Windows::Forms::RadioButton^ serie23_rb;
private: ZedGraph::ZedGraphControl^ principal_zgc;
private: System::Windows::Forms::Label^ titulo_lab;
private: System::Windows::Forms::CheckBox^ s2a1_ckb;
private: System::Windows::Forms::CheckBox^ s2b1_ckb;
private: System::Windows::Forms::CheckBox^ s2a2_ckb;
private: System::Windows::Forms::CheckBox^ s3a1_ckb;
private: System::Windows::Forms::CheckBox^ s3b1_ckb;
private: System::Windows::Forms::CheckBox^ s3c1_ckb;
private: System::Windows::Forms::CheckBox^ s3b2_ckb;
private: System::Windows::Forms::CheckBox^ s3c2_ckb;
private: System::Windows::Forms::Label^ grupoS2_lab;
private: System::Windows::Forms::Label^ grupoS3_lab;
private: System::Windows::Forms::GroupBox^ grupos_gb;
private: System::Windows::Forms::CheckBox^ s2f1_ckb;
private: System::Windows::Forms::CheckBox^ s2f2_ckb;
private: System::Windows::Forms::CheckBox^ s3f2_ckb;
private: System::Windows::Forms::CheckBox^ s3f1_ckb;
private: System::Windows::Forms::Label^ faseS2_lab;
private: System::Windows::Forms::Label^ faseS3_lab;
private: System::Windows::Forms::GroupBox^ fase_gb;
private: System::Windows::Forms::CheckBox^ s2par_ckb;
private: System::Windows::Forms::CheckBox^ s2impar_ckb;
private: System::Windows::Forms::GroupBox^ lateral_gb;
private: System::Windows::Forms::Label^ lateralS2_lab;
private: System::Windows::Forms::Label^ lateralS3_lab;
private: System::Windows::Forms::CheckBox^ s3par_ckb;
private: System::Windows::Forms::CheckBox^ s3impar_ckb;
private: System::Windows::Forms::GroupBox^ filtrado_gb;
private: System::Windows::Forms::DataGridView^ interface_dgv;
private: System::ComponentModel::IContainer^ components;

private:
    /// <summary>
    /// Required designer variable.
    /// </summary>

```



```

#pragma region Windows Form Designer generated code
    /// <summary>
    /// Required method for Designer support - do not modify
    /// the contents of this method with the code editor.
    /// </summary>
    void InitializeComponent(void)
    {
        this->components = (gcnew
System::ComponentModel::Container());
        this->desde_dtp = (gcnew
System::Windows::Forms::DateTimePicker());
        this->hasta_dtp = (gcnew
System::Windows::Forms::DateTimePicker());
        this->desde_lab = (gcnew
System::Windows::Forms::Label());
        this->hasta_lab = (gcnew
System::Windows::Forms::Label());
        this->serie2_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->serie3_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->serie23_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->principal_zgc = (gcnew
ZedGraph::ZedGraphControl());
        this->titulo_lab = (gcnew
System::Windows::Forms::Label());
        this->s2a1_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s2b1_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s2a2_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s3a1_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s3b1_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s3c1_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s3b2_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s3c2_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->grupoS2_lab = (gcnew
System::Windows::Forms::Label());
        this->grupoS3_lab = (gcnew
System::Windows::Forms::Label());
        this->grupos_gb = (gcnew
System::Windows::Forms::GroupBox());
        this->s2f1_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s2f2_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->s3f2_ckb = (gcnew
System::Windows::Forms::CheckBox());
    }

```

```

        this->s3fl_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->faseS2_lab = (gcnew
System::Windows::Forms::Label());
        this->faseS3_lab = (gcnew
System::Windows::Forms::Label());
        this->fase_gb = (gcnew
System::Windows::Forms::GroupBox());
        this->s2par_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->s2impar_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->lateral_gb = (gcnew
System::Windows::Forms::GroupBox());
        this->lateralS2_lab = (gcnew
System::Windows::Forms::Label());
        this->lateralS3_lab = (gcnew
System::Windows::Forms::Label());
        this->s3par_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->s3impar_cke = (gcnew
System::Windows::Forms::CheckBox());
        this->filtrado_gb = (gcnew
System::Windows::Forms::GroupBox());
        this->edad_lab = (gcnew System::Windows::Forms::Label());
        this->edadhasta_lab = (gcnew
System::Windows::Forms::Label());
        this->edaddesde_lab = (gcnew
System::Windows::Forms::Label());
        this->edadhasta_nud = (gcnew
System::Windows::Forms::NumericUpDown());
        this->edaddesde_nud = (gcnew
System::Windows::Forms::NumericUpDown());
        this->interface_dgv = (gcnew
System::Windows::Forms::DataGridView());
        this->dateTimeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->noisSPNVoltsDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->tappingWeightScheduleDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalIronFeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalLithiumLiDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measBathTemperatureControlDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->fedFailuresDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->resCycleAvgVoltsDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());

```

```

        this->measBathLevelDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalPadLevelDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Age = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->potDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->comparativaBindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->TablaGeneral = (gcnew menu::TablaGeneral());
        this->var1_cb = (gcnew
System::Windows::Forms::ComboBox());
        this->var2_cb = (gcnew
System::Windows::Forms::ComboBox());
        this->filtrado_gb2 = (gcnew
System::Windows::Forms::GroupBox());
        this->lateral_gb2 = (gcnew
System::Windows::Forms::GroupBox());
        this->lateralS2_lab2 = (gcnew
System::Windows::Forms::Label());
        this->lateralS3_lab2 = (gcnew
System::Windows::Forms::Label());
        this->s3par_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3impar_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s2par_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s2impar_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->fase_gb2 = (gcnew
System::Windows::Forms::GroupBox());
        this->faseS2_lab2 = (gcnew
System::Windows::Forms::Label());
        this->faseS3_lab2 = (gcnew
System::Windows::Forms::Label());
        this->s2f1_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s2f2_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3f2_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3f1_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->grupos_gb2 = (gcnew
System::Windows::Forms::GroupBox());
        this->grupoS2_lab2 = (gcnew
System::Windows::Forms::Label());
        this->grupoS3_lab2 = (gcnew
System::Windows::Forms::Label());
        this->s2a1_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s2b1_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3c2_ckb2 = (gcnew
System::Windows::Forms::CheckBox());

```

```

        this->s2a2_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3b2_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3a1_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3c1_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->s3b1_ckb2 = (gcnew
System::Windows::Forms::CheckBox());
        this->serie23_rb2 = (gcnew
System::Windows::Forms::RadioButton());
        this->edad_lab2 = (gcnew
System::Windows::Forms::Label());
        this->serie3_rb2 = (gcnew
System::Windows::Forms::RadioButton());
        this->edadhasta_lab2 = (gcnew
System::Windows::Forms::Label());
        this->serie2_rb2 = (gcnew
System::Windows::Forms::RadioButton());
        this->edaddesde_lab2 = (gcnew
System::Windows::Forms::Label());
        this->edaddesde_nud2 = (gcnew
System::Windows::Forms::NumericUpDown());
        this->edadhasta_nud2 = (gcnew
System::Windows::Forms::NumericUpDown());
        this->draw_but = (gcnew
System::Windows::Forms::Button());
        this->alortal_lab = (gcnew
System::Windows::Forms::Label());
        this->print_but = (gcnew
System::Windows::Forms::Button());
        this->print2_but = (gcnew
System::Windows::Forms::Button());
        this->comparativaTableAdapter = (gcnew
menu::TablaGeneralTableAdapters::comparativaTableAdapter());
        this->grupos_gb->SuspendLayout();
        this->fase_gb->SuspendLayout();
        this->lateral_gb->SuspendLayout();
        this->filtrado_gb->SuspendLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>edadhasta_nud)->BeginInit());

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>edaddesde_nud)->BeginInit());

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>interface_dgv)->BeginInit());

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>comparativaBindingSource)->BeginInit());

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>TablaGeneral)->BeginInit());
        this->filtrado_gb2->SuspendLayout();
        this->lateral_gb2->SuspendLayout();

```

```

        this->fase_gb2->SuspendLayout();
        this->grupos_gb2->SuspendLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this->
        >edadesde_nud2))->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this->
        >edadhasta_nud2))->BeginInit();
        this->SuspendLayout();
        //
        // desde_dtp
        //
        this->desde_dtp->Location = System::Drawing::Point(261,
140);

        this->desde_dtp->Name = L"desde_dtp";
        this->desde_dtp->Size = System::Drawing::Size(200, 20);
        this->desde_dtp->TabIndex = 0;
        this->desde_dtp->ValueChanged += gcnew
System::EventHandler(this, &Form3::desde_dtp_ValueChanged);
        //
        // hasta_dtp
        //
        this->hasta_dtp->Location = System::Drawing::Point(816,
140);

        this->hasta_dtp->Name = L"hasta_dtp";
        this->hasta_dtp->Size = System::Drawing::Size(200, 20);
        this->hasta_dtp->TabIndex = 1;
        this->hasta_dtp->ValueChanged += gcnew
System::EventHandler(this, &Form3::hasta_dtp_ValueChanged);
        //
        // desde_lab
        //
        this->desde_lab->AutoSize = true;
        this->desde_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->desde_lab->Location = System::Drawing::Point(258,
121);

        this->desde_lab->Name = L"desde_lab";
        this->desde_lab->Size = System::Drawing::Size(47, 13);
        this->desde_lab->TabIndex = 2;
        this->desde_lab->Text = L"Desde:";
        //
        // hasta_lab
        //
        this->hasta_lab->AutoSize = true;
        this->hasta_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->hasta_lab->Location = System::Drawing::Point(813,
121);

        this->hasta_lab->Name = L"hasta_lab";
        this->hasta_lab->Size = System::Drawing::Size(44, 13);
        this->hasta_lab->TabIndex = 3;
        this->hasta_lab->Text = L"Hasta:";

```

```

//
// serie2_rb
//
this->serie2_rb->AutoSize = true;
this->serie2_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->serie2_rb->Location = System::Drawing::Point(6,
19);

this->serie2_rb->Name = L"serie2_rb";
this->serie2_rb->Size = System::Drawing::Size(65, 17);
this->serie2_rb->TabIndex = 4;
this->serie2_rb->TabStop = true;
this->serie2_rb->Text = L"Serie 2";
this->serie2_rb->UseVisualStyleBackColor = true;
this->serie2_rb->Click += gcnew
System::EventHandler(this, &Form3::serie2_rb_Click);
this->serie2_rb->MouseClicked += gcnew
System::Windows::Forms::EventHandler(this,
&Form3::serie2_rb_MouseClick);
//
// serie3_rb
//
this->serie3_rb->AutoSize = true;
this->serie3_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->serie3_rb->Location = System::Drawing::Point(6,
43);

this->serie3_rb->Name = L"serie3_rb";
this->serie3_rb->Size = System::Drawing::Size(65, 17);
this->serie3_rb->TabIndex = 5;
this->serie3_rb->TabStop = true;
this->serie3_rb->Text = L"Serie 3";
this->serie3_rb->UseVisualStyleBackColor = true;
this->serie3_rb->Click += gcnew
System::EventHandler(this, &Form3::serie3_rb_Click);
this->serie3_rb->MouseClicked += gcnew
System::Windows::Forms::EventHandler(this,
&Form3::serie3_rb_MouseClick);
//
// serie23_rb
//
this->serie23_rb->AutoSize = true;
this->serie23_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->serie23_rb->Location = System::Drawing::Point(6,
67);

this->serie23_rb->Name = L"serie23_rb";
this->serie23_rb->Size = System::Drawing::Size(86, 17);
this->serie23_rb->TabIndex = 6;
this->serie23_rb->TabStop = true;
this->serie23_rb->Text = L"Serie 2 y 3";

```

```

        this->serie23_rb->UseVisualStyleBackColor = true;
        this->serie23_rb->Click += gcnew
System::EventHandler(this, &Form3::serie23_rb_Click);
        this->serie23_rb->MouseClicked += gcnew
System::Windows::Forms::MouseEventHandler(this,
&Form3::serie23_rb_MouseClick);
        //
        // principal_zgc
        //
        this->principal_zgc->Location =
System::Drawing::Point(169, 176);
        this->principal_zgc->Name = L"principal_zgc";
        this->principal_zgc->ScrollGrace = 0;
        this->principal_zgc->ScrollMaxX = 0;
        this->principal_zgc->ScrollMaxY = 0;
        this->principal_zgc->ScrollMaxY2 = 0;
        this->principal_zgc->ScrollMinX = 0;
        this->principal_zgc->ScrollMinY = 0;
        this->principal_zgc->ScrollMinY2 = 0;
        this->principal_zgc->Size = System::Drawing::Size(942,
573);

        this->principal_zgc->TabIndex = 7;
        //
        // titulo_lab
        //
        this->titulo_lab->AutoSize = true;
        this->titulo_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 20,
static_cast<System::Drawing::FontStyle>((System::Drawing::FontStyle::Bold |
System::Drawing::FontStyle::Underline)),
        System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
        this->titulo_lab->Location = System::Drawing::Point(493,
22);

        this->titulo_lab->Name = L"titulo_lab";
        this->titulo_lab->Size = System::Drawing::Size(286, 31);
        this->titulo_lab->TabIndex = 8;
        this->titulo_lab->Text = L"Análisis de Variables";
        //
        // s2a1_ckb
        //
        this->s2a1_ckb->AutoSize = true;
        this->s2a1_ckb->BackColor =
System::Drawing::Color::Gainsboro;
        this->s2a1_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2a1_ckb->Location = System::Drawing::Point(9, 48);
        this->s2a1_ckb->Name = L"s2a1_ckb";
        this->s2a1_ckb->Size = System::Drawing::Size(39, 17);
        this->s2a1_ckb->TabIndex = 9;
        this->s2a1_ckb->Text = L"A1";
        this->s2a1_ckb->UseVisualStyleBackColor = false;
        this->s2a1_ckb->Click += gcnew System::EventHandler(this,
&Form3::s2a1_ckb_Click);
        //

```

```

        // s2b1_ckb
        //
        this->s2b1_ckb->AutoSize = true;
        this->s2b1_ckb->BackColor =
System::Drawing::Color::Yellow;
        this->s2b1_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2b1_ckb->Location = System::Drawing::Point(9, 71);
        this->s2b1_ckb->Name = L"s2b1_ckb";
        this->s2b1_ckb->Size = System::Drawing::Size(39, 17);
        this->s2b1_ckb->TabIndex = 10;
        this->s2b1_ckb->Text = L"B1";
        this->s2b1_ckb->UseVisualStyleBackColor = false;
        this->s2b1_ckb->Click += gcnew System::EventHandler(this,
&Form3::s2b1_ckb_Click);
        //
        // s2a2_ckb
        //
        this->s2a2_ckb->AutoSize = true;
        this->s2a2_ckb->BackColor =
System::Drawing::Color::Yellow;
        this->s2a2_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2a2_ckb->Location = System::Drawing::Point(9, 94);
        this->s2a2_ckb->Name = L"s2a2_ckb";
        this->s2a2_ckb->Size = System::Drawing::Size(39, 17);
        this->s2a2_ckb->TabIndex = 11;
        this->s2a2_ckb->Text = L"A2";
        this->s2a2_ckb->UseVisualStyleBackColor = false;
        this->s2a2_ckb->Click += gcnew System::EventHandler(this,
&Form3::s2a2_ckb_Click);
        //
        // s3a1_ckb
        //
        this->s3a1_ckb->AutoSize = true;
        this->s3a1_ckb->BackColor =
System::Drawing::Color::Gainsboro;
        this->s3a1_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3a1_ckb->Location = System::Drawing::Point(84,
48);
        this->s3a1_ckb->Name = L"s3a1_ckb";
        this->s3a1_ckb->Size = System::Drawing::Size(39, 17);
        this->s3a1_ckb->TabIndex = 12;
        this->s3a1_ckb->Text = L"A1";
        this->s3a1_ckb->UseVisualStyleBackColor = false;
        this->s3a1_ckb->Click += gcnew System::EventHandler(this,
&Form3::s3a1_ckb_Click);
        //
        // s3b1_ckb
        //

```



```

        this->s3b1_ckb->AutoSize = true;
        this->s3b1_ckb->BackColor =
System::Drawing::Color::Gainsboro;
        this->s3b1_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3b1_ckb->Location = System::Drawing::Point(84,
71);

        this->s3b1_ckb->Name = L"s3b1_ckb";
        this->s3b1_ckb->Size = System::Drawing::Size(39, 17);
        this->s3b1_ckb->TabIndex = 13;
        this->s3b1_ckb->Text = L"B1";
        this->s3b1_ckb->UseVisualStyleBackColor = false;
        this->s3b1_ckb->Click += gcnew System::EventHandler(this,
&Form3::s3b1_ckb_Click);
        //
        // s3c1_ckb
        //
        this->s3c1_ckb->AutoSize = true;
        this->s3c1_ckb->BackColor =
System::Drawing::Color::Yellow;
        this->s3c1_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3c1_ckb->Location = System::Drawing::Point(84,
94);

        this->s3c1_ckb->Name = L"s3c1_ckb";
        this->s3c1_ckb->Size = System::Drawing::Size(39, 17);
        this->s3c1_ckb->TabIndex = 14;
        this->s3c1_ckb->Text = L"C1";
        this->s3c1_ckb->UseVisualStyleBackColor = false;
        this->s3c1_ckb->Click += gcnew System::EventHandler(this,
&Form3::s3c1_ckb_Click);
        //
        // s3b2_ckb
        //
        this->s3b2_ckb->AutoSize = true;
        this->s3b2_ckb->BackColor =
System::Drawing::Color::Yellow;
        this->s3b2_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3b2_ckb->Location = System::Drawing::Point(84,
117);

        this->s3b2_ckb->Name = L"s3b2_ckb";
        this->s3b2_ckb->Size = System::Drawing::Size(39, 17);
        this->s3b2_ckb->TabIndex = 15;
        this->s3b2_ckb->Text = L"B2";
        this->s3b2_ckb->UseVisualStyleBackColor = false;
        this->s3b2_ckb->Click += gcnew System::EventHandler(this,
&Form3::s3b2_ckb_Click);
        //
        // s3c2_ckb
        //

```

```

        this->s3c2_ckb->AutoSize = true;
        this->s3c2_ckb->BackColor =
System::Drawing::Color::Tomato;
        this->s3c2_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3c2_ckb->Location = System::Drawing::Point(84,
140);

        this->s3c2_ckb->Name = L"s3c2_ckb";
        this->s3c2_ckb->Size = System::Drawing::Size(39, 17);
        this->s3c2_ckb->TabIndex = 16;
        this->s3c2_ckb->Text = L"C2";
        this->s3c2_ckb->UseVisualStyleBackColor = false;
        this->s3c2_ckb->Click += gcnew System::EventHandler(this,
&Form3::s3c2_ckb_Click);
        //
        // grupoS2_lab
        //
        this->grupoS2_lab->AutoSize = true;
        this->grupoS2_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->grupoS2_lab->Location = System::Drawing::Point(6,
25);

        this->grupoS2_lab->Name = L"grupoS2_lab";
        this->grupoS2_lab->Size = System::Drawing::Size(22, 13);
        this->grupoS2_lab->TabIndex = 17;
        this->grupoS2_lab->Text = L"S2";
        //
        // grupoS3_lab
        //
        this->grupoS3_lab->AutoSize = true;
        this->grupoS3_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->grupoS3_lab->Location = System::Drawing::Point(81,
25);

        this->grupoS3_lab->Name = L"grupoS3_lab";
        this->grupoS3_lab->Size = System::Drawing::Size(22, 13);
        this->grupoS3_lab->TabIndex = 18;
        this->grupoS3_lab->Text = L"S3";
        //
        // grupos_gb
        //
        this->grupos_gb->Controls->Add(this->grupoS2_lab);
        this->grupos_gb->Controls->Add(this->grupoS3_lab);
        this->grupos_gb->Controls->Add(this->s2a1_ckb);
        this->grupos_gb->Controls->Add(this->s2b1_ckb);
        this->grupos_gb->Controls->Add(this->s3c2_ckb);
        this->grupos_gb->Controls->Add(this->s2a2_ckb);
        this->grupos_gb->Controls->Add(this->s3b2_ckb);
        this->grupos_gb->Controls->Add(this->s3a1_ckb);
        this->grupos_gb->Controls->Add(this->s3c1_ckb);
        this->grupos_gb->Controls->Add(this->s3b1_ckb);

```

```

        this->grupos_gb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->grupos_gb->Location = System::Drawing::Point(6,
90);

        this->grupos_gb->Name = L"grupos_gb";
        this->grupos_gb->Size = System::Drawing::Size(139, 167);
        this->grupos_gb->TabIndex = 20;
        this->grupos_gb->TabStop = false;
        this->grupos_gb->Text = L"Grupos de Arranque";
        //
        // s2f1_ckb
        //
        this->s2f1_ckb->AutoSize = true;
        this->s2f1_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2f1_ckb->Location = System::Drawing::Point(9, 50);
        this->s2f1_ckb->Name = L"s2f1_ckb";
        this->s2f1_ckb->Size = System::Drawing::Size(58, 17);
        this->s2f1_ckb->TabIndex = 21;
        this->s2f1_ckb->Text = L"Fase 1";
        this->s2f1_ckb->UseVisualStyleBackColor = true;
        this->s2f1_ckb->Click += gcnew System::EventHandler(this,
&Form3::s2f1_ckb_Click);
        //
        // s2f2_ckb
        //
        this->s2f2_ckb->AutoSize = true;
        this->s2f2_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2f2_ckb->Location = System::Drawing::Point(9, 73);
        this->s2f2_ckb->Name = L"s2f2_ckb";
        this->s2f2_ckb->Size = System::Drawing::Size(58, 17);
        this->s2f2_ckb->TabIndex = 22;
        this->s2f2_ckb->Text = L"Fase 2";
        this->s2f2_ckb->UseVisualStyleBackColor = true;
        this->s2f2_ckb->Click += gcnew System::EventHandler(this,
&Form3::s2f2_ckb_Click);
        //
        // s3f2_ckb
        //
        this->s3f2_ckb->AutoSize = true;
        this->s3f2_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3f2_ckb->Location = System::Drawing::Point(73,
73);

        this->s3f2_ckb->Name = L"s3f2_ckb";
        this->s3f2_ckb->Size = System::Drawing::Size(58, 17);
        this->s3f2_ckb->TabIndex = 24;
        this->s3f2_ckb->Text = L"Fase 2";

```

```

        this->s3f2_ckb->UseVisualStyleBackColor = true;
        this->s3f2_ckb->Click += gcnew System::EventHandler(this,
&Form3::s3f2_ckb_Click);
        //
        // s3f1_ckb
        //
        this->s3f1_ckb->AutoSize = true;
        this->s3f1_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3f1_ckb->Location = System::Drawing::Point(73,
50);

        this->s3f1_ckb->Name = L"s3f1_ckb";
        this->s3f1_ckb->Size = System::Drawing::Size(58, 17);
        this->s3f1_ckb->TabIndex = 23;
        this->s3f1_ckb->Text = L"Fase 1";
        this->s3f1_ckb->UseVisualStyleBackColor = true;
        this->s3f1_ckb->Click += gcnew System::EventHandler(this,
&Form3::s3f1_ckb_Click);
        //
        // faseS2_lab
        //
        this->faseS2_lab->AutoSize = true;
        this->faseS2_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->faseS2_lab->Location = System::Drawing::Point(6,
25);

        this->faseS2_lab->Name = L"faseS2_lab";
        this->faseS2_lab->Size = System::Drawing::Size(22, 13);
        this->faseS2_lab->TabIndex = 25;
        this->faseS2_lab->Text = L"S2";
        //
        // faseS3_lab
        //
        this->faseS3_lab->AutoSize = true;
        this->faseS3_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->faseS3_lab->Location = System::Drawing::Point(70,
25);

        this->faseS3_lab->Name = L"faseS3_lab";
        this->faseS3_lab->Size = System::Drawing::Size(22, 13);
        this->faseS3_lab->TabIndex = 26;
        this->faseS3_lab->Text = L"S3";
        //
        // fase_gb
        //
        this->fase_gb->Controls->Add(this->faseS2_lab);
        this->fase_gb->Controls->Add(this->faseS3_lab);
        this->fase_gb->Controls->Add(this->s2f1_ckb);
        this->fase_gb->Controls->Add(this->s2f2_ckb);
        this->fase_gb->Controls->Add(this->s3f2_ckb);
        this->fase_gb->Controls->Add(this->s3f1_ckb);

```

```

        this->fase_gb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->fase_gb->Location = System::Drawing::Point(6, 263);
        this->fase_gb->Name = L"fase_gb";
        this->fase_gb->Size = System::Drawing::Size(139, 98);
        this->fase_gb->TabIndex = 27;
        this->fase_gb->TabStop = false;
        this->fase_gb->Text = L"Fases";
        //
        // s2par_ckb
        //
        this->s2par_ckb->AutoSize = true;
        this->s2par_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2par_ckb->Location = System::Drawing::Point(9,
52);

        this->s2par_ckb->Name = L"s2par_ckb";
        this->s2par_ckb->Size = System::Drawing::Size(42, 17);
        this->s2par_ckb->TabIndex = 28;
        this->s2par_ckb->Text = L"Par";
        this->s2par_ckb->UseVisualStyleBackColor = true;
        this->s2par_ckb->Click += gcnew
System::EventHandler(this, &Form3::s2par_ckb_Click);
        //
        // s2impar_ckb
        //
        this->s2impar_ckb->AutoSize = true;
        this->s2impar_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2impar_ckb->Location = System::Drawing::Point(9,
75);

        this->s2impar_ckb->Name = L"s2impar_ckb";
        this->s2impar_ckb->Size = System::Drawing::Size(52, 17);
        this->s2impar_ckb->TabIndex = 29;
        this->s2impar_ckb->Text = L"Impar";
        this->s2impar_ckb->UseVisualStyleBackColor = true;
        this->s2impar_ckb->Click += gcnew
System::EventHandler(this, &Form3::s2impar_ckb_Click);
        //
        // lateral_gb
        //
        this->lateral_gb->Controls->Add(this->lateralS2_lab);
        this->lateral_gb->Controls->Add(this->lateralS3_lab);
        this->lateral_gb->Controls->Add(this->s3par_ckb);
        this->lateral_gb->Controls->Add(this->s3impar_ckb);
        this->lateral_gb->Controls->Add(this->s2par_ckb);
        this->lateral_gb->Controls->Add(this->s2impar_ckb);
        this->lateral_gb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));

```

```

        this->lateral_gb->Location = System::Drawing::Point(6,
367);
        this->lateral_gb->Name = L"lateral_gb";
        this->lateral_gb->Size = System::Drawing::Size(139, 101);
        this->lateral_gb->TabIndex = 30;
        this->lateral_gb->TabStop = false;
        this->lateral_gb->Text = L"Lateral";
        //
        // lateralS2_lab
        //
        this->lateralS2_lab->AutoSize = true;
        this->lateralS2_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->lateralS2_lab->Location = System::Drawing::Point(6,
26);
        this->lateralS2_lab->Name = L"lateralS2_lab";
        this->lateralS2_lab->Size = System::Drawing::Size(22,
13);
        this->lateralS2_lab->TabIndex = 32;
        this->lateralS2_lab->Text = L"S2";
        //
        // lateralS3_lab
        //
        this->lateralS3_lab->AutoSize = true;
        this->lateralS3_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->lateralS3_lab->Location =
System::Drawing::Point(70, 26);
        this->lateralS3_lab->Name = L"lateralS3_lab";
        this->lateralS3_lab->Size = System::Drawing::Size(22,
13);
        this->lateralS3_lab->TabIndex = 33;
        this->lateralS3_lab->Text = L"S3";
        //
        // s3par_ckb
        //
        this->s3par_ckb->AutoSize = true;
        this->s3par_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3par_ckb->Location = System::Drawing::Point(73,
52);
        this->s3par_ckb->Name = L"s3par_ckb";
        this->s3par_ckb->Size = System::Drawing::Size(42, 17);
        this->s3par_ckb->TabIndex = 30;
        this->s3par_ckb->Text = L"Par";
        this->s3par_ckb->UseVisualStyleBackColor = true;
        this->s3par_ckb->Click += gcnew
System::EventHandler(this, &Form3::s3par_ckb_Click);
        //
        // s3impar_ckb
        //

```

```

        this->s3impar_ckb->AutoSize = true;
        this->s3impar_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3impar_ckb->Location = System::Drawing::Point(73,
75);

        this->s3impar_ckb->Name = L"s3impar_ckb";
        this->s3impar_ckb->Size = System::Drawing::Size(52, 17);
        this->s3impar_ckb->TabIndex = 31;
        this->s3impar_ckb->Text = L"Impar";
        this->s3impar_ckb->UseVisualStyleBackColor = true;
        this->s3impar_ckb->Click += gcnew
System::EventHandler(this, &Form3::s3impar_ckb_Click);
        //
        // filtrado_gb
        //
        this->filtrado_gb->Controls->Add(this->edad_lab);
        this->filtrado_gb->Controls->Add(this->edadhasta_lab);
        this->filtrado_gb->Controls->Add(this->edaddesde_lab);
        this->filtrado_gb->Controls->Add(this->edadhasta_nud);
        this->filtrado_gb->Controls->Add(this->edaddesde_nud);
        this->filtrado_gb->Controls->Add(this->lateral_gb);
        this->filtrado_gb->Controls->Add(this->fase_gb);
        this->filtrado_gb->Controls->Add(this->grupos_gb);
        this->filtrado_gb->Controls->Add(this->serie23_rb);
        this->filtrado_gb->Controls->Add(this->serie3_rb);
        this->filtrado_gb->Controls->Add(this->serie2_rb);
        this->filtrado_gb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->filtrado_gb->Location = System::Drawing::Point(12,
160);

        this->filtrado_gb->Name = L"filtrado_gb";
        this->filtrado_gb->Size = System::Drawing::Size(151,
601);

        this->filtrado_gb->TabIndex = 31;
        this->filtrado_gb->TabStop = false;
        this->filtrado_gb->Text = L"Filtrado de Cubas";
        //
        // edad_lab
        //
        this->edad_lab->AutoSize = true;
        this->edad_lab->Location = System::Drawing::Point(32,
474);

        this->edad_lab->Name = L"edad_lab";
        this->edad_lab->Size = System::Drawing::Size(89, 13);
        this->edad_lab->TabIndex = 40;
        this->edad_lab->Text = L"Filtro por edad";
        //
        // edadhasta_lab
        //
        this->edadhasta_lab->AutoSize = true;
        this->edadhasta_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

```

```

        static_cast<System::Byte>(0));
        this->edadhasta_lab->Location =
System::Drawing::Point(85, 495);
        this->edadhasta_lab->Name = L"edadhasta_lab";
        this->edadhasta_lab->Size = System::Drawing::Size(38,
13);

        this->edadhasta_lab->TabIndex = 39;
        this->edadhasta_lab->Text = L"Hasta:";
        //
        // edaddesde_lab
        //
        this->edaddesde_lab->AutoSize = true;
        this->edaddesde_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->edaddesde_lab->Location =
System::Drawing::Point(10, 495);
        this->edaddesde_lab->Name = L"edaddesde_lab";
        this->edaddesde_lab->Size = System::Drawing::Size(41,
13);

        this->edaddesde_lab->TabIndex = 38;
        this->edaddesde_lab->Text = L"Desde:";
        //
        // edadhasta_nud
        //
        this->edadhasta_nud->Location =
System::Drawing::Point(88, 511);
        this->edadhasta_nud->Maximum = System::Decimal(gcnew
cli::array< System::Int32 >(4) {9999, 0, 0, 0});
        this->edadhasta_nud->Minimum = System::Decimal(gcnew
cli::array< System::Int32 >(4) {1, 0, 0, 0});
        this->edadhasta_nud->Name = L"edadhasta_nud";
        this->edadhasta_nud->Size = System::Drawing::Size(54,
20);

        this->edadhasta_nud->TabIndex = 37;
        this->edadhasta_nud->Value = System::Decimal(gcnew
cli::array< System::Int32 >(4) {5000, 0, 0, 0});
        this->edadhasta_nud->ValueChanged += gcnew
System::EventHandler(this, &Form3::edadhasta_nud_ValueChanged);
        //
        // edaddesde_nud
        //
        this->edaddesde_nud->Location =
System::Drawing::Point(11, 511);
        this->edaddesde_nud->Maximum = System::Decimal(gcnew
cli::array< System::Int32 >(4) {9999, 0, 0, 0});
        this->edaddesde_nud->Minimum = System::Decimal(gcnew
cli::array< System::Int32 >(4) {1, 0, 0, 0});
        this->edaddesde_nud->Name = L"edaddesde_nud";
        this->edaddesde_nud->Size = System::Drawing::Size(54,
20);

        this->edaddesde_nud->TabIndex = 36;
        this->edaddesde_nud->Value = System::Decimal(gcnew
cli::array< System::Int32 >(4) {1, 0, 0, 0});
        this->edaddesde_nud->ValueChanged += gcnew
System::EventHandler(this, &Form3::edaddesde_nud_ValueChanged);

```



```

//
// interface_dgv
//
this->interface_dgv->AutoGenerateColumns = false;
this->interface_dgv->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
this->interface_dgv->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^ >(15) {this-
>dateTimeDataGridViewTextBoxColumn,
    this->noisSPPNVoltsDataGridViewTextBoxColumn, this-
>mEASSTPBATHXSALF3DataGridViewTextBoxColumn, this-
>aETimeAbove8VoltsDataGridViewTextBoxColumn,
    this-
>tappingWeightScheduleDataGridViewTextBoxColumn, this-
>measMetalIronFeDataGridViewTextBoxColumn, this-
>measMetalLithiumLiDataGridViewTextBoxColumn,
    this-
>measBathTemperatureControlDataGridViewTextBoxColumn, this-
>fedFailuresDataGridViewTextBoxColumn, this-
>resCycleAvgAmpsControlDataGridViewTextBoxColumn,
    this->resCycleAvgVoltsDataGridViewTextBoxColumn,
this->measBathLevelDataGridViewTextBoxColumn, this-
>measMetalPadLevelDataGridViewTextBoxColumn,
    this->Age, this->potDataGridViewTextBoxColumn});
this->interface_dgv->DataSource = this-
>comparativaBindingSource;
this->interface_dgv->Location =
System::Drawing::Point(12, 817);
this->interface_dgv->Name = L"interface_dgv";
this->interface_dgv->Size = System::Drawing::Size(1152,
144);
this->interface_dgv->TabIndex = 32;
//
// dateTimeDataGridViewTextBoxColumn
//
this->dateTimeDataGridViewTextBoxColumn->DataPropertyName
= L"Date_Time";
this->dateTimeDataGridViewTextBoxColumn->HeaderText =
L"Date_Time";
this->dateTimeDataGridViewTextBoxColumn->Name =
L"dateTimeDataGridViewTextBoxColumn";
//
// noisSPPNVoltsDataGridViewTextBoxColumn
//
this->noisSPPNVoltsDataGridViewTextBoxColumn->
>DataPropertyName = L"Nois_SPPN_Volts";
this->noisSPPNVoltsDataGridViewTextBoxColumn->HeaderText
= L"Nois_SPPN_Volts";
this->noisSPPNVoltsDataGridViewTextBoxColumn->Name =
L"noisSPPNVoltsDataGridViewTextBoxColumn";
//
// mEASSTPBATHXSALF3DataGridViewTextBoxColumn
//
this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn->
>DataPropertyName = L"MEAS_STP_BATH_XSALF3";
this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn->
>HeaderText = L"MEAS_STP_BATH_XSALF3";

```

```

        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn->Name =
L"mEASSTPBATHXSALF3DataGridViewTextBoxColumn";
        //
        // aETimeAbove8VoltsDataGridViewTextBoxColumn
        //
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn-
>DataPropertyName = L"AE_Time_Above_8_Volts";
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn-
>HeaderText = L"AE_Time_Above_8_Volts";
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn->Name =
L"aETimeAbove8VoltsDataGridViewTextBoxColumn";
        //
        // tappingWeightScheduleDataGridViewTextBoxColumn
        //
        this->tappingWeightScheduleDataGridViewTextBoxColumn-
>DataPropertyName = L"Tapping_Weight_Schedule";
        this->tappingWeightScheduleDataGridViewTextBoxColumn-
>HeaderText = L"Tapping_Weight_Schedule";
        this->tappingWeightScheduleDataGridViewTextBoxColumn-
>Name = L"tappingWeightScheduleDataGridViewTextBoxColumn";
        //
        // measMetalIronFeDataGridViewTextBoxColumn
        //
        this->measMetalIronFeDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Iron_Fe";
        this->measMetalIronFeDataGridViewTextBoxColumn-
>HeaderText = L"Meas_Metal_Iron_Fe";
        this->measMetalIronFeDataGridViewTextBoxColumn->Name =
L"measMetalIronFeDataGridViewTextBoxColumn";
        //
        // measMetalLithiumLiDataGridViewTextBoxColumn
        //
        this->measMetalLithiumLiDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Lithium_Li";
        this->measMetalLithiumLiDataGridViewTextBoxColumn-
>HeaderText = L"Meas_Metal_Lithium_Li";
        this->measMetalLithiumLiDataGridViewTextBoxColumn->Name =
L"measMetalLithiumLiDataGridViewTextBoxColumn";
        //
        // measBathTemperatureControlDataGridViewTextBoxColumn
        //
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->DataPropertyName =
L"Meas_Bath_Temperature_Control";
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->HeaderText =
L"Meas_Bath_Temperature_Control";
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->Name =
L"measBathTemperatureControlDataGridViewTextBoxColumn";
        //
        // fedFailuresDataGridViewTextBoxColumn
        //
        this->fedFailuresDataGridViewTextBoxColumn-
>DataPropertyName = L"Fed_Failures";
        this->fedFailuresDataGridViewTextBoxColumn->HeaderText =
L"Fed_Failures";

```

```

        this->fedFailuresDataGridViewTextBoxColumn->Name           =
L"fedFailuresDataGridViewTextBoxColumn";
        this->fedFailuresDataGridViewTextBoxColumn->ReadOnly     =
true;
        //
        // resCycleAvgAmpsControlDataGridViewTextBoxColumn
        //
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn-
>DataPropertyName = L"Res_Cycle_Avg_Amps_Control";
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn-
>HeaderText = L"Res_Cycle_Avg_Amps_Control";
        this->resCycleAvgAmpsControlDataGridViewTextBoxColumn-
>Name = L"resCycleAvgAmpsControlDataGridViewTextBoxColumn";
        //
        // resCycleAvgVoltsDataGridViewTextBoxColumn
        //
        this->resCycleAvgVoltsDataGridViewTextBoxColumn-
>DataPropertyName = L"Res_Cycle_Avg_Volts";
        this->resCycleAvgVoltsDataGridViewTextBoxColumn-
>HeaderText = L"Res_Cycle_Avg_Volts";
        this->resCycleAvgVoltsDataGridViewTextBoxColumn->Name   =
L"resCycleAvgVoltsDataGridViewTextBoxColumn";
        //
        // measBathLevelDataGridViewTextBoxColumn
        //
        this->measBathLevelDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Bath_Level";
        this->measBathLevelDataGridViewTextBoxColumn->HeaderText
= L"Meas_Bath_Level";
        this->measBathLevelDataGridViewTextBoxColumn->Name     =
L"measBathLevelDataGridViewTextBoxColumn";
        //
        // measMetalPadLevelDataGridViewTextBoxColumn
        //
        this->measMetalPadLevelDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Pad_Level";
        this->measMetalPadLevelDataGridViewTextBoxColumn-
>HeaderText = L"Meas_Metal_Pad_Level";
        this->measMetalPadLevelDataGridViewTextBoxColumn->Name   =
L"measMetalPadLevelDataGridViewTextBoxColumn";
        //
        // Age
        //
        this->Age->DataPropertyName = L"Age";
        this->Age->HeaderText = L"Age";
        this->Age->Name = L"Age";
        //
        // potDataGridViewTextBoxColumn
        //
        this->potDataGridViewTextBoxColumn->DataPropertyName     =
L"Pot";
        this->potDataGridViewTextBoxColumn->HeaderText = L"Pot";
        this->potDataGridViewTextBoxColumn->Name               =
L"potDataGridViewTextBoxColumn";
        //
        // comparativaBindingSource
        //

```

```

this->comparativaBindingSource->DataMember =
L"comparativa";
this->comparativaBindingSource->DataSource = this-
>TablaGeneral;
//
// TablaGeneral
//
this->TablaGeneral->DataSetName = L"TablaGeneral";
this->TablaGeneral->SchemaSerializationMode =
System::Data::SchemaSerializationMode::IncludeSchema;
//
// var1_cb
//
this->var1_cb->FormattingEnabled = true;
this->var1_cb->Items->AddRange(gcnew cli::array<
System::Object^ >(13) {L"Ninguno", L"SPPN", L"AlF3", L"AEs / Día",
L"Faraday",
L"% Pureza", L"Li", L"Temperatura", L"Fed / Día",
L"CEE", L"Tensión", L"Altura Baño", L"Altura Metal"});
this->var1_cb->Location = System::Drawing::Point(298,
768);

this->var1_cb->Name = L"var1_cb";
this->var1_cb->Size = System::Drawing::Size(121, 21);
this->var1_cb->TabIndex = 33;
this->var1_cb->SelectedIndexChanged += gcnew
System::EventHandler(this, &Form3::var1_cb_SelectedIndexChanged);
//
// var2_cb
//
this->var2_cb->FormattingEnabled = true;
this->var2_cb->Items->AddRange(gcnew cli::array<
System::Object^ >(13) {L"Ninguno", L"SPPN", L"AlF3", L"AEs / Día",
L"Faraday",
L"% Pureza", L"Li", L"Temperatura", L"Fed / Día",
L"CEE", L"Tensión", L"Altura Baño", L"Altura Metal"});
this->var2_cb->Location = System::Drawing::Point(526,
768);

this->var2_cb->Name = L"var2_cb";
this->var2_cb->Size = System::Drawing::Size(121, 21);
this->var2_cb->TabIndex = 34;
this->var2_cb->SelectedIndexChanged += gcnew
System::EventHandler(this, &Form3::var2_cb_SelectedIndexChanged);
//
// filtrado_gb2
//
this->filtrado_gb2->Controls->Add(this->lateral_gb2);
this->filtrado_gb2->Controls->Add(this->fase_gb2);
this->filtrado_gb2->Controls->Add(this->grupos_gb2);
this->filtrado_gb2->Controls->Add(this->serie23_rb2);
this->filtrado_gb2->Controls->Add(this->edad_lab2);
this->filtrado_gb2->Controls->Add(this->serie3_rb2);
this->filtrado_gb2->Controls->Add(this->edadhasta_lab2);
this->filtrado_gb2->Controls->Add(this->serie2_rb2);
this->filtrado_gb2->Controls->Add(this->edaddesde_lab2);
this->filtrado_gb2->Controls->Add(this->edaddesde_nud2);
this->filtrado_gb2->Controls->Add(this->edadhasta_nud2);

```

```

        this->filtrado_gb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->filtrado_gb2->Location =
System::Drawing::Point(1117, 160);
        this->filtrado_gb2->Name = L"filtrado_gb2";
        this->filtrado_gb2->Size = System::Drawing::Size(151,
601);

        this->filtrado_gb2->TabIndex = 35;
        this->filtrado_gb2->TabStop = false;
        this->filtrado_gb2->Text = L"Filtrado de Cubas";
        //
        // lateral_gb2
        //
        this->lateral_gb2->Controls->Add(this->lateralS2_lab2);
        this->lateral_gb2->Controls->Add(this->lateralS3_lab2);
        this->lateral_gb2->Controls->Add(this->s3par_cke2);
        this->lateral_gb2->Controls->Add(this->s3impar_cke2);
        this->lateral_gb2->Controls->Add(this->s2par_cke2);
        this->lateral_gb2->Controls->Add(this->s2impar_cke2);
        this->lateral_gb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->lateral_gb2->Location = System::Drawing::Point(6,
367);

        this->lateral_gb2->Name = L"lateral_gb2";
        this->lateral_gb2->Size = System::Drawing::Size(139,
101);

        this->lateral_gb2->TabIndex = 30;
        this->lateral_gb2->TabStop = false;
        this->lateral_gb2->Text = L"Lateral";
        //
        // lateralS2_lab2
        //
        this->lateralS2_lab2->AutoSize = true;
        this->lateralS2_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->lateralS2_lab2->Location =
System::Drawing::Point(6, 26);
        this->lateralS2_lab2->Name = L"lateralS2_lab2";
        this->lateralS2_lab2->Size = System::Drawing::Size(22,
13);

        this->lateralS2_lab2->TabIndex = 32;
        this->lateralS2_lab2->Text = L"S2";
        //
        // lateralS3_lab2
        //
        this->lateralS3_lab2->AutoSize = true;
        this->lateralS3_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));

```

```

        this->lateralS3_lab2->Location =
System::Drawing::Point(70, 26);
        this->lateralS3_lab2->Name = L"lateralS3_lab2";
        this->lateralS3_lab2->Size = System::Drawing::Size(22,
13);

        this->lateralS3_lab2->TabIndex = 33;
        this->lateralS3_lab2->Text = L"S3";
        //
        // s3par_ckb2
        //
        this->s3par_ckb2->AutoSize = true;
        this->s3par_ckb2->Font = (gcnw
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3par_ckb2->Location = System::Drawing::Point(73,
52);

        this->s3par_ckb2->Name = L"s3par_ckb2";
        this->s3par_ckb2->Size = System::Drawing::Size(42, 17);
        this->s3par_ckb2->TabIndex = 30;
        this->s3par_ckb2->Text = L"Par";
        this->s3par_ckb2->UseVisualStyleBackColor = true;
        this->s3par_ckb2->Click += gcnw
System::EventHandler(this, &Form3::s3par_ckb2_Click);
        //
        // s3impar_ckb2
        //
        this->s3impar_ckb2->AutoSize = true;
        this->s3impar_ckb2->Font = (gcnw
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3impar_ckb2->Location = System::Drawing::Point(73,
75);

        this->s3impar_ckb2->Name = L"s3impar_ckb2";
        this->s3impar_ckb2->Size = System::Drawing::Size(52, 17);
        this->s3impar_ckb2->TabIndex = 31;
        this->s3impar_ckb2->Text = L"Impar";
        this->s3impar_ckb2->UseVisualStyleBackColor = true;
        this->s3impar_ckb2->Click += gcnw
System::EventHandler(this, &Form3::s3impar_ckb2_Click);
        //
        // s2par_ckb2
        //
        this->s2par_ckb2->AutoSize = true;
        this->s2par_ckb2->Font = (gcnw
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2par_ckb2->Location = System::Drawing::Point(9,
52);

        this->s2par_ckb2->Name = L"s2par_ckb2";
        this->s2par_ckb2->Size = System::Drawing::Size(42, 17);
        this->s2par_ckb2->TabIndex = 28;
        this->s2par_ckb2->Text = L"Par";
        this->s2par_ckb2->UseVisualStyleBackColor = true;

```

```

        this->s2par_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s2par_ckb2_Click);
        //
        // s2impar_ckb2
        //
        this->s2impar_ckb2->AutoSize = true;
        this->s2impar_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2impar_ckb2->Location = System::Drawing::Point(9,
75);

        this->s2impar_ckb2->Name = L"s2impar_ckb2";
        this->s2impar_ckb2->Size = System::Drawing::Size(52, 17);
        this->s2impar_ckb2->TabIndex = 29;
        this->s2impar_ckb2->Text = L"Impar";
        this->s2impar_ckb2->UseVisualStyleBackColor = true;
        this->s2impar_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s2impar_ckb2_Click);
        //
        // fase_gb2
        //
        this->fase_gb2->Controls->Add(this->faseS2_lab2);
        this->fase_gb2->Controls->Add(this->faseS3_lab2);
        this->fase_gb2->Controls->Add(this->s2f1_ckb2);
        this->fase_gb2->Controls->Add(this->s2f2_ckb2);
        this->fase_gb2->Controls->Add(this->s3f2_ckb2);
        this->fase_gb2->Controls->Add(this->s3f1_ckb2);
        this->fase_gb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->fase_gb2->Location = System::Drawing::Point(6,
263);

        this->fase_gb2->Name = L"fase_gb2";
        this->fase_gb2->Size = System::Drawing::Size(139, 98);
        this->fase_gb2->TabIndex = 27;
        this->fase_gb2->TabStop = false;
        this->fase_gb2->Text = L"Fases";
        //
        // faseS2_lab2
        //
        this->faseS2_lab2->AutoSize = true;
        this->faseS2_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->faseS2_lab2->Location = System::Drawing::Point(6,
25);

        this->faseS2_lab2->Name = L"faseS2_lab2";
        this->faseS2_lab2->Size = System::Drawing::Size(22, 13);
        this->faseS2_lab2->TabIndex = 25;
        this->faseS2_lab2->Text = L"S2";
        //
        // faseS3_lab2
        //
        this->faseS3_lab2->AutoSize = true;

```

```

                this->faseS3_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(0)));
                this->faseS3_lab2->Location = System::Drawing::Point(70,
25);

                this->faseS3_lab2->Name = L"faseS3_lab2";
                this->faseS3_lab2->Size = System::Drawing::Size(22, 13);
                this->faseS3_lab2->TabIndex = 26;
                this->faseS3_lab2->Text = L"S3";
                //
                // s2f1_ckb2
                //
                this->s2f1_ckb2->AutoSize = true;
                this->s2f1_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(0)));
                this->s2f1_ckb2->Location = System::Drawing::Point(9,
50);

                this->s2f1_ckb2->Name = L"s2f1_ckb2";
                this->s2f1_ckb2->Size = System::Drawing::Size(58, 17);
                this->s2f1_ckb2->TabIndex = 21;
                this->s2f1_ckb2->Text = L"Fase 1";
                this->s2f1_ckb2->UseVisualStyleBackColor = true;
                this->s2f1_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s2f1_ckb2_Click);
                //
                // s2f2_ckb2
                //
                this->s2f2_ckb2->AutoSize = true;
                this->s2f2_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(0)));
                this->s2f2_ckb2->Location = System::Drawing::Point(9,
73);

                this->s2f2_ckb2->Name = L"s2f2_ckb2";
                this->s2f2_ckb2->Size = System::Drawing::Size(58, 17);
                this->s2f2_ckb2->TabIndex = 22;
                this->s2f2_ckb2->Text = L"Fase 2";
                this->s2f2_ckb2->UseVisualStyleBackColor = true;
                this->s2f2_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s2f2_ckb2_Click);
                //
                // s3f2_ckb2
                //
                this->s3f2_ckb2->AutoSize = true;
                this->s3f2_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(0)));
                this->s3f2_ckb2->Location = System::Drawing::Point(73,
73);

                this->s3f2_ckb2->Name = L"s3f2_ckb2";
                this->s3f2_ckb2->Size = System::Drawing::Size(58, 17);
                this->s3f2_ckb2->TabIndex = 24;

```



```

        this->s3f2_ckb2->Text = L"Fase 2";
        this->s3f2_ckb2->UseVisualStyleBackColor = true;
        this->s3f2_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s3f2_ckb2_Click);
        //
        // s3f1_ckb2
        //
        this->s3f1_ckb2->AutoSize = true;
        this->s3f1_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3f1_ckb2->Location = System::Drawing::Point(73,
50);

        this->s3f1_ckb2->Name = L"s3f1_ckb2";
        this->s3f1_ckb2->Size = System::Drawing::Size(58, 17);
        this->s3f1_ckb2->TabIndex = 23;
        this->s3f1_ckb2->Text = L"Fase 1";
        this->s3f1_ckb2->UseVisualStyleBackColor = true;
        this->s3f1_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s3f1_ckb2_Click);
        //
        // grupos_gb2
        //
        this->grupos_gb2->Controls->Add(this->grupoS2_lab2);
        this->grupos_gb2->Controls->Add(this->grupoS3_lab2);
        this->grupos_gb2->Controls->Add(this->s2a1_ckb2);
        this->grupos_gb2->Controls->Add(this->s2b1_ckb2);
        this->grupos_gb2->Controls->Add(this->s3c2_ckb2);
        this->grupos_gb2->Controls->Add(this->s2a2_ckb2);
        this->grupos_gb2->Controls->Add(this->s3b2_ckb2);
        this->grupos_gb2->Controls->Add(this->s3a1_ckb2);
        this->grupos_gb2->Controls->Add(this->s3c1_ckb2);
        this->grupos_gb2->Controls->Add(this->s3b1_ckb2);
        this->grupos_gb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->grupos_gb2->Location = System::Drawing::Point(6,
90);

        this->grupos_gb2->Name = L"grupos_gb2";
        this->grupos_gb2->Size = System::Drawing::Size(139, 167);
        this->grupos_gb2->TabIndex = 20;
        this->grupos_gb2->TabStop = false;
        this->grupos_gb2->Text = L"Grupos de Arranque";
        //
        // grupoS2_lab2
        //
        this->grupoS2_lab2->AutoSize = true;
        this->grupoS2_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->grupoS2_lab2->Location = System::Drawing::Point(6,
25);

        this->grupoS2_lab2->Name = L"grupoS2_lab2";
        this->grupoS2_lab2->Size = System::Drawing::Size(22, 13);

```

```

        this->grupoS2_lab2->TabIndex = 17;
        this->grupoS2_lab2->Text = L"S2";
        //
        // grupoS3_lab2
        //
        this->grupoS3_lab2->AutoSize = true;
        this->grupoS3_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->grupoS3_lab2->Location = System::Drawing::Point(81,
25);

        this->grupoS3_lab2->Name = L"grupoS3_lab2";
        this->grupoS3_lab2->Size = System::Drawing::Size(22, 13);
        this->grupoS3_lab2->TabIndex = 18;
        this->grupoS3_lab2->Text = L"S3";
        //
        // s2a1_ckb2
        //
        this->s2a1_ckb2->AutoSize = true;
        this->s2a1_ckb2->BackColor =
System::Drawing::Color::Gainsboro;
        this->s2a1_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2a1_ckb2->Location = System::Drawing::Point(9,
48);

        this->s2a1_ckb2->Name = L"s2a1_ckb2";
        this->s2a1_ckb2->Size = System::Drawing::Size(39, 17);
        this->s2a1_ckb2->TabIndex = 9;
        this->s2a1_ckb2->Text = L"A1";
        this->s2a1_ckb2->UseVisualStyleBackColor = false;
        this->s2a1_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s2a1_ckb2_Click);
        //
        // s2b1_ckb2
        //
        this->s2b1_ckb2->AutoSize = true;
        this->s2b1_ckb2->BackColor =
System::Drawing::Color::Yellow;
        this->s2b1_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2b1_ckb2->Location = System::Drawing::Point(9,
71);

        this->s2b1_ckb2->Name = L"s2b1_ckb2";
        this->s2b1_ckb2->Size = System::Drawing::Size(39, 17);
        this->s2b1_ckb2->TabIndex = 10;
        this->s2b1_ckb2->Text = L"B1";
        this->s2b1_ckb2->UseVisualStyleBackColor = false;
        this->s2b1_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s2b1_ckb2_Click);
        //
        // s3c2_ckb2
        //

```

```

        this->s3c2_ckb2->AutoSize = true;
        this->s3c2_ckb2->BackColor =
System::Drawing::Color::Tomato;
        this->s3c2_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3c2_ckb2->Location = System::Drawing::Point(84,
140);

        this->s3c2_ckb2->Name = L"s3c2_ckb2";
        this->s3c2_ckb2->Size = System::Drawing::Size(39, 17);
        this->s3c2_ckb2->TabIndex = 16;
        this->s3c2_ckb2->Text = L"C2";
        this->s3c2_ckb2->UseVisualStyleBackColor = false;
        this->s3c2_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s3c2_ckb2_Click);
        //
        // s2a2_ckb2
        //
        this->s2a2_ckb2->AutoSize = true;
        this->s2a2_ckb2->BackColor =
System::Drawing::Color::Yellow;
        this->s2a2_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s2a2_ckb2->Location = System::Drawing::Point(9,
94);

        this->s2a2_ckb2->Name = L"s2a2_ckb2";
        this->s2a2_ckb2->Size = System::Drawing::Size(39, 17);
        this->s2a2_ckb2->TabIndex = 11;
        this->s2a2_ckb2->Text = L"A2";
        this->s2a2_ckb2->UseVisualStyleBackColor = false;
        this->s2a2_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s2a2_ckb2_Click);
        //
        // s3b2_ckb2
        //
        this->s3b2_ckb2->AutoSize = true;
        this->s3b2_ckb2->BackColor =
System::Drawing::Color::Yellow;
        this->s3b2_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3b2_ckb2->Location = System::Drawing::Point(84,
117);

        this->s3b2_ckb2->Name = L"s3b2_ckb2";
        this->s3b2_ckb2->Size = System::Drawing::Size(39, 17);
        this->s3b2_ckb2->TabIndex = 15;
        this->s3b2_ckb2->Text = L"B2";
        this->s3b2_ckb2->UseVisualStyleBackColor = false;
        this->s3b2_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s3b2_ckb2_Click);
        //
        // s3a1_ckb2
        //

```

```

        this->s3a1_ckb2->AutoSize = true;
        this->s3a1_ckb2->BackColor =
System::Drawing::Color::Gainsboro;
        this->s3a1_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3a1_ckb2->Location = System::Drawing::Point(84,
48);

        this->s3a1_ckb2->Name = L"s3a1_ckb2";
        this->s3a1_ckb2->Size = System::Drawing::Size(39, 17);
        this->s3a1_ckb2->TabIndex = 12;
        this->s3a1_ckb2->Text = L"A1";
        this->s3a1_ckb2->UseVisualStyleBackColor = false;
        this->s3a1_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s3a1_ckb2_Click);
        //
        // s3c1_ckb2
        //
        this->s3c1_ckb2->AutoSize = true;
        this->s3c1_ckb2->BackColor =
System::Drawing::Color::Yellow;
        this->s3c1_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3c1_ckb2->Location = System::Drawing::Point(84,
94);

        this->s3c1_ckb2->Name = L"s3c1_ckb2";
        this->s3c1_ckb2->Size = System::Drawing::Size(39, 17);
        this->s3c1_ckb2->TabIndex = 14;
        this->s3c1_ckb2->Text = L"C1";
        this->s3c1_ckb2->UseVisualStyleBackColor = false;
        this->s3c1_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s3c1_ckb2_Click);
        //
        // s3b1_ckb2
        //
        this->s3b1_ckb2->AutoSize = true;
        this->s3b1_ckb2->BackColor =
System::Drawing::Color::Gainsboro;
        this->s3b1_ckb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->s3b1_ckb2->Location = System::Drawing::Point(84,
71);

        this->s3b1_ckb2->Name = L"s3b1_ckb2";
        this->s3b1_ckb2->Size = System::Drawing::Size(39, 17);
        this->s3b1_ckb2->TabIndex = 13;
        this->s3b1_ckb2->Text = L"B1";
        this->s3b1_ckb2->UseVisualStyleBackColor = false;
        this->s3b1_ckb2->Click += gcnew
System::EventHandler(this, &Form3::s3b1_ckb2_Click);
        //
        // serie23_rb2
        //

```

```

        this->serie23_rb2->AutoSize = true;
        this->serie23_rb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->serie23_rb2->Location = System::Drawing::Point(6,
67);

        this->serie23_rb2->Name = L"serie23_rb2";
        this->serie23_rb2->Size = System::Drawing::Size(86, 17);
        this->serie23_rb2->TabIndex = 6;
        this->serie23_rb2->TabStop = true;
        this->serie23_rb2->Text = L"Serie 2 y 3";
        this->serie23_rb2->UseVisualStyleBackColor = true;
        this->serie23_rb2->Click += gcnew
System::EventHandler(this, &Form3::serie23_rb2_Click);
        this->serie23_rb2->MouseClicked += gcnew
System::Windows::Forms::MouseEventHandler(this,
&Form3::serie23_rb2_MouseClick);
        //
        // edad_lab2
        //
        this->edad_lab2->AutoSize = true;
        this->edad_lab2->Location = System::Drawing::Point(34,
474);

        this->edad_lab2->Name = L"edad_lab2";
        this->edad_lab2->Size = System::Drawing::Size(89, 13);
        this->edad_lab2->TabIndex = 47;
        this->edad_lab2->Text = L"Filtro por edad";
        //
        // serie3_rb2
        //
        this->serie3_rb2->AutoSize = true;
        this->serie3_rb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->serie3_rb2->Location = System::Drawing::Point(6,
43);

        this->serie3_rb2->Name = L"serie3_rb2";
        this->serie3_rb2->Size = System::Drawing::Size(65, 17);
        this->serie3_rb2->TabIndex = 5;
        this->serie3_rb2->TabStop = true;
        this->serie3_rb2->Text = L"Serie 3";
        this->serie3_rb2->UseVisualStyleBackColor = true;
        this->serie3_rb2->Click += gcnew
System::EventHandler(this, &Form3::serie3_rb2_Click);
        this->serie3_rb2->MouseClicked += gcnew
System::Windows::Forms::MouseEventHandler(this,
&Form3::serie3_rb2_MouseClick);
        //
        // edadhasta_lab2
        //
        this->edadhasta_lab2->AutoSize = true;
        this->edadhasta_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));

```

```

        this->edadhasta_lab2->Location =
System::Drawing::Point(87, 495);
        this->edadhasta_lab2->Name = L"edadhasta_lab2";
        this->edadhasta_lab2->Size = System::Drawing::Size(38,
13);

        this->edadhasta_lab2->TabIndex = 46;
        this->edadhasta_lab2->Text = L"Hasta:";
        //
        // serie2_rb2
        //
        this->serie2_rb2->AutoSize = true;
        this->serie2_rb2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->serie2_rb2->Location = System::Drawing::Point(6,
19);

        this->serie2_rb2->Name = L"serie2_rb2";
        this->serie2_rb2->Size = System::Drawing::Size(65, 17);
        this->serie2_rb2->TabIndex = 4;
        this->serie2_rb2->TabStop = true;
        this->serie2_rb2->Text = L"Serie 2";
        this->serie2_rb2->UseVisualStyleBackColor = true;
        this->serie2_rb2->Click += gcnew
System::EventHandler(this, &Form3::serie2_rb2_Click);
        this->serie2_rb2->MouseClicked += gcnew
System::Windows::Forms::EventHandler(this,
&Form3::serie2_rb2_MouseClick);
        //
        // edaddesde_lab2
        //
        this->edaddesde_lab2->AutoSize = true;
        this->edaddesde_lab2->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->edaddesde_lab2->Location =
System::Drawing::Point(12, 495);
        this->edaddesde_lab2->Name = L"edaddesde_lab2";
        this->edaddesde_lab2->Size = System::Drawing::Size(41,
13);

        this->edaddesde_lab2->TabIndex = 45;
        this->edaddesde_lab2->Text = L"Desde:";
        //
        // edaddesde_nud2
        //
        this->edaddesde_nud2->Location =
System::Drawing::Point(13, 511);
        this->edaddesde_nud2->Maximum = System::Decimal(gcnew
cli::array< System::Int32 >(4) {9999, 0, 0, 0});
        this->edaddesde_nud2->Minimum = System::Decimal(gcnew
cli::array< System::Int32 >(4) {1, 0, 0, 0});
        this->edaddesde_nud2->Name = L"edaddesde_nud2";
        this->edaddesde_nud2->Size = System::Drawing::Size(54,
20);

        this->edaddesde_nud2->TabIndex = 43;

```

```

        this->edaddesde_nud2->Value      =      System::Decimal(gcnew
cli::array< System::Int32 >(4) {1, 0, 0, 0});
        this->edaddesde_nud2->ValueChanged      +=      gcnew
System::EventHandler(this, &Form3::edaddesde_nud2_ValueChanged);
        //
        // edadhasta_nud2
        //
        this->edadhasta_nud2->Location      =
System::Drawing::Point(90, 511);
        this->edadhasta_nud2->Maximum      =      System::Decimal(gcnew
cli::array< System::Int32 >(4) {9999, 0, 0, 0});
        this->edadhasta_nud2->Minimum      =      System::Decimal(gcnew
cli::array< System::Int32 >(4) {1, 0, 0, 0});
        this->edadhasta_nud2->Name = L"edadhasta_nud2";
        this->edadhasta_nud2->Size      =      System::Drawing::Size(54,
20);

        this->edadhasta_nud2->TabIndex = 44;
        this->edadhasta_nud2->Value      =      System::Decimal(gcnew
cli::array< System::Int32 >(4) {5000, 0, 0, 0});
        this->edadhasta_nud2->ValueChanged      +=      gcnew
System::EventHandler(this, &Form3::edadhasta_nud2_ValueChanged);
        //
        // draw_but
        //
        this->draw_but->Font      =      (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
        this->draw_but->Location      =      System::Drawing::Point(602,
136);

        this->draw_but->Name = L"draw_but";
        this->draw_but->Size = System::Drawing::Size(75, 23);
        this->draw_but->TabIndex = 36;
        this->draw_but->Text = L"Actualizar";
        this->draw_but->UseVisualStyleBackColor = true;
        this->draw_but->Click += gcnew System::EventHandler(this,
&Form3::draw_but_Click);
        //
        // alertal_lab
        //
        this->alertal_lab->AutoSize = true;
        this->alertal_lab->Font      =      (gcnew
System::Drawing::Font(L"Microsoft      Sans      Serif",      10,
static_cast<System::Drawing::FontStyle>((System::Drawing::FontStyle::Bold |
System::Drawing::FontStyle::Underline)),
System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
        this->alertal_lab->ForeColor      =
System::Drawing::Color::OrangeRed;
        this->alertal_lab->Location = System::Drawing::Point(549,
116);

        this->alertal_lab->Name = L"alertal_lab";
        this->alertal_lab->Size = System::Drawing::Size(177, 17);
        this->alertal_lab->TabIndex = 37;
        this->alertal_lab->Text = L"Gráfica NO actualizada";
        this->alertal_lab->Visible = false;
        //

```

```

// print_but
//
this->print_but->Location = System::Drawing::Point(534,
79);

this->print_but->Name = L"print_but";
this->print_but->Size = System::Drawing::Size(93, 23);
this->print_but->TabIndex = 38;
this->print_but->Text = L"Guardar gráfica";
this->print_but->UseVisualStyleBackColor = true;
this->print_but->Click += gcnew
System::EventHandler(this, &Form3::print_but_Click);
//
// print2_but
//
this->print2_but->Location = System::Drawing::Point(652,
79);

this->print2_but->Name = L"print2_but";
this->print2_but->Size = System::Drawing::Size(93, 23);
this->print2_but->TabIndex = 39;
this->print2_but->Text = L"Guardar captura";
this->print2_but->UseVisualStyleBackColor = true;
this->print2_but->Click += gcnew
System::EventHandler(this, &Form3::print2_but_Click);
//
// comparativaTableAdapter
//
this->comparativaTableAdapter->ClearBeforeFill = true;
//
// Form3
//
this->AutoScaleDimensions = System::Drawing::SizeF(6,
13);

this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
this->AutoScroll = true;
this->ClientSize = System::Drawing::Size(1270, 983);
this->Controls->Add(this->print2_but);
this->Controls->Add(this->print_but);
this->Controls->Add(this->alerta1_lab);
this->Controls->Add(this->draw_but);
this->Controls->Add(this->filtrado_gb2);
this->Controls->Add(this->var2_cb);
this->Controls->Add(this->var1_cb);
this->Controls->Add(this->interface_dgv);
this->Controls->Add(this->filtrado_gb);
this->Controls->Add(this->titulo_lab);
this->Controls->Add(this->principal_zgc);
this->Controls->Add(this->hasta_lab);
this->Controls->Add(this->desde_lab);
this->Controls->Add(this->hasta_dtp);
this->Controls->Add(this->desde_dtp);
this->Name = L"Form3";
this->Text = L"Análisis de Variables";
this->WindowState =
System::Windows::Forms::FormWindowState::Maximized;
this->Load += gcnew System::EventHandler(this,
&Form3::Form3_Load);

```



```

        this->FormClosing += gcnew
System::Windows::Forms::FormClosingEventHandler(this,
&Form3::Form3_FormClosing);
        this->grupos_gb->ResumeLayout(false);
        this->grupos_gb->PerformLayout();
        this->fase_gb->ResumeLayout(false);
        this->fase_gb->PerformLayout();
        this->lateral_gb->ResumeLayout(false);
        this->lateral_gb->PerformLayout();
        this->filtrado_gb->ResumeLayout(false);
        this->filtrado_gb->PerformLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>edadhasta_nud))->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>edadesde_nud))->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>interface_dgv))->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>comparativaBindingSource))->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>TablaGeneral))->EndInit();
        this->filtrado_gb2->ResumeLayout(false);
        this->filtrado_gb2->PerformLayout();
        this->lateral_gb2->ResumeLayout(false);
        this->lateral_gb2->PerformLayout();
        this->fase_gb2->ResumeLayout(false);
        this->fase_gb2->PerformLayout();
        this->grupos_gb2->ResumeLayout(false);
        this->grupos_gb2->PerformLayout();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>edadesde_nud2))->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>edadhasta_nud2))->EndInit();
        this->ResumeLayout(false);
        this->PerformLayout();

    }

#pragma endregion
    private: System::Void Form3_FormClosing(System::Object^ sender,
System::Windows::Forms::FormClosingEventArgs^ e) {
        e->Cancel = true; //Al cerrar la ventana de la
forma no se cierra la forma
        this->Visible = false; //En vez de eso se oculta
    }

    //Lo que se ejecuta al iniciar la forma
    private: System::Void Form3_Load(System::Object^ sender,
System::EventArgs^ e) {

```

```

        this->flagnoeventos=true; //Inicializa a false
la flag para permitir los eventos durante la ejecución del programa
        this->desde_dtp->Value=this->desde_dtp-
>Value.AddDays(-3); //Inicializa el desde dtp a 15 días antes de hoy
        this->flagnoeventos=false;
        this->comparativaTableAdapter->Fill(this-
>TablaGeneral->comparativa, this->desde_dtp->Value.Date, this->hasta_dtp-
>Value.Date); //Carga el dgv interface
        this->serie2_rb_MouseClick(nullptr, nullptr);
//Lanzo el evento de click sobre el rb de serie 2 para que el programa
inicie con todas las ckb activadas
        this->serie2_rb->Checked=1; //Lanzar el evento no
activa el rb, hay que hacerlo aparte
        this->serie3_rb2_MouseClick(nullptr, nullptr);
//Lanzo el evento de click sobre el rb de serie 2 para que el programa
inicie con todas las ckb activadas
        this->serie3_rb2->Checked=1; //Lanzar el evento no
activa el rb, hay que hacerlo aparte
        this->var1_cb->SelectedIndex=0; //Inicializo la cb
de la primera columna a 'ninguno'
        this->var2_cb->SelectedIndex=0; //Inicializo la cb
de la primera columna a 'ninguno'
        this->principal_zgc->GraphPane->Title-
>Text="Comparativa de la Evolución";
        this->principal_zgc->GraphPane->XAxis->Title-
>Text="Día del mes";
        this->principal_zgc->GraphPane->YAxis->Title-
>Text="Variable Seleccionada";
        this->principal_zgc->GraphPane->Y2Axis-
>IsVisible=true;
        this->principal_zgc->GraphPane->Fill= gcnew
Fill(Color::LightSkyBlue);
        this->principal_zgc->GraphPane->Chart->Fill= gcnew
Fill(Color::FromArgb(230, 230, 255));
        //this->principal_zgc->GraphPane->YAxis-
>MajorGrid->IsVisible= true;
        //this->principal_zgc->GraphPane->YAxis-
>MinorGrid->IsVisible= true;
    }

//Respuestas a las activaciones y desactivaciones de las
checkbox principales
    private: System::Void s2a1_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2(this->s2a1_cke->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2b1_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2(this->s2b1_cke->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2a2_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2(this->s2a2_cke->Checked);
        this->alortal_lab->Visible=1;
    }
}

```

```

private: System::Void s2f1_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2(this->s2f1_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s2f2_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2(this->s2f2_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s2par_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2(this->s2par_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s2impar_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2(this->s2impar_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3a1_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3(this->s3a1_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3b1_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3(this->s3b1_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3c1_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3(this->s3c1_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3b2_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3(this->s3b2_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3c2_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3(this->s3c2_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3f1_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3(this->s3f1_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3f2_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3(this->s3f2_cke->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3par_cke_Click(System::Object^ sender,
System::EventArgs^ e) {

```

```

        this->serie3(this->s3par_cke->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s3impar_cke_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie3(this->s3impar_cke->Checked);
        this->alortal_lab->Visible=1;
    }

    //Respuestas a las activaciones y desactivaciones de los
checkbox secundarios
    private: System::Void s2a1_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2sec(this->s2a1_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2b1_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2sec(this->s2b1_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2a2_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2sec(this->s2a2_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2f1_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2sec(this->s2f1_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2f2_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2sec(this->s2f2_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2par_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2sec(this->s2par_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s2impar_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2sec(this->s2impar_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s3a1_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie3sec(this->s3a1_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
    private: System::Void s3b1_cke2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie3sec(this->s3b1_cke2->Checked);
        this->alortal_lab->Visible=1;
    }
}

```

```

private: System::Void s3c1_ckb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3sec(this->s3c1_ckb2->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3b2_ckb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3sec(this->s3b2_ckb2->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3c2_ckb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3sec(this->s3c2_ckb2->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3f1_ckb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3sec(this->s3f1_ckb2->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3f2_ckb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3sec(this->s3f2_ckb2->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3par_ckb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3sec(this->s3par_ckb2->Checked);
    this->alortal_lab->Visible=1;
}
private: System::Void s3impar_ckb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3sec(this->s3impar_ckb2->Checked);
    this->alortal_lab->Visible=1;
}

//Lo que ocurre al activarse un rb principal
private: System::Void serie2_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3_rb->Checked=false; //Al activar rb de serie
2, desactivo los otros
    this->serie23_rb->Checked=false;
    this->flagnoeventos=true; //Si se activó manualmente, y
no por la desactivación de la última checkbox, es necesario desactivar
todas las checkbox. Para no disparar los eventos de cada checkbox al
cambiar su valor, se emplea el flag inhibidor de eventos
    this->s3a1_ckb->Checked=0;    this->s3b1_ckb->Checked=0;

    this->s3c1_ckb->Checked=0; this->s3b2_ckb->Checked=0;
    this->s3c2_ckb->Checked=0; this->s3f1_ckb->Checked=0;
    this->s3f2_ckb->Checked=0; this->s3par_ckb->Checked=0;
    this->s3impar_ckb->Checked=0;
    this->alortal_lab->Visible=1;
    this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
}

```

```

private: System::Void serie3_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2_rb->Checked=false;//Al activar rb de serie
3, desactivo los otros
    this->serie23_rb->Checked=false;
    this->flagnoeventos=true; //Si se activó manualmente, y
no por la desactivación de la última checkbox, es necesario desactivar
todas las checkbox. Para no disparar los eventos de cada checkbox al
cambiar su valor, se emplea el flag inhibidor de eventos
    this->s2a1_ckb->Checked=0; this->s2b1_ckb->Checked=0;
    this->s2a2_ckb->Checked=0; this->s2f1_ckb->Checked=0;
    this->s2f2_ckb->Checked=0; this->s2par_ckb->Checked=0;
    this->s2impar_ckb->Checked=0;
    this->alertal_lab->Visible=1;
    this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
}
private: System::Void serie23_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2_rb->Checked=false;
    this->serie3_rb->Checked=false;
    this->alertal_lab->Visible=1;
}

//Lo que ocurre al activar un rb secundario
private: System::Void serie2_rb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie3_rb2->Checked=false; //Al activar rb de
serie 2, desactivo los otros
    this->serie23_rb2->Checked=false;
    this->flagnoeventos=true; //Si se activó manualmente, y
no por la desactivación de la última checkbox, es necesario desactivar
todas las checkbox. Para no disparar los eventos de cada checkbox al
cambiar su valor, se emplea el flag inhibidor de eventos
    this->s3a1_ckb2->Checked=0; this->s3b1_ckb2->Checked=0;

    this->s3c1_ckb2->Checked=0; this->s3b2_ckb2->Checked=0;
    this->s3c2_ckb2->Checked=0; this->s3f1_ckb2->Checked=0;
    this->s3f2_ckb2->Checked=0; this->s3par_ckb2->Checked=0;
    this->s3impar_ckb2->Checked=0;
    this->alertal_lab->Visible=1;
    this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
}
private: System::Void serie3_rb2_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->serie2_rb2->Checked=false;//Al activar rb de serie
3, desactivo los otros
    this->serie23_rb2->Checked=false;
    this->flagnoeventos=true; //Si se activó manualmente, y
no por la desactivación de la última checkbox, es necesario desactivar
todas las checkbox. Para no disparar los eventos de cada checkbox al
cambiar su valor, se emplea el flag inhibidor de eventos
    this->s2a1_ckb2->Checked=0; this->s2b1_ckb2->Checked=0;
    this->s2a2_ckb2->Checked=0; this->s2f1_ckb2->Checked=0;
    this->s2f2_ckb2->Checked=0; this->s2par_ckb2->Checked=0;
    this->s2impar_ckb2->Checked=0;

```

```

        this->alerta1_lab->Visible=1;
        this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
    }
    private: System::Void serie23_rb2_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->serie2_rb2->Checked=false;
        this->serie3_rb2->Checked=false;
        this->alerta1_lab->Visible=1;
    }

    //Lo que ocurre al activarse un rb principal MANUALMENTE
    private: System::Void serie2_rb_MouseClick(System::Object^ sender,
System::Windows::Forms::MouseEventArgs^ e) {
        this->serie3_rb->Checked=false; //Al activar rb de serie
2, desactivo los otros
        this->serie23_rb->Checked=false;

        this->flagnoeventos=true; //Se desactivan las ckb de
serie 3 y se activan las de serie 2. Para no disparar los eventos de cada
checkbox al cambiar su valor, se emplea el flag inhibidor de eventos
        this->s3a1_cke->Checked=0; this->s3b1_cke->Checked=0;

        this->s3c1_cke->Checked=0; this->s3b2_cke->Checked=0;
        this->s3c2_cke->Checked=0; this->s3f1_cke->Checked=0;
        this->s3f2_cke->Checked=0; this->s3par_cke->Checked=0;
        this->s3impar_cke->Checked=0;

        this->s2a1_cke->Checked=1; this->s2b1_cke->Checked=1;
        this->s2a2_cke->Checked=1; this->s2f1_cke->Checked=1;
        this->s2f2_cke->Checked=1; this->s2par_cke->Checked=1;
        this->s2impar_cke->Checked=1;
        this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
    }
    private: System::Void serie3_rb_MouseClick(System::Object^ sender,
System::Windows::Forms::MouseEventArgs^ e) {
        this->serie2_rb->Checked=false; //Al activar rb de serie
3, desactivo los otros
        this->serie23_rb->Checked=false;

        this->flagnoeventos=true; //Se desactivan las ckb de
serie 2 y se activan las de serie 3. Para no disparar los eventos de cada
checkbox al cambiar su valor, se emplea el flag inhibidor de eventos
        this->s2a1_cke->Checked=0; this->s2b1_cke->Checked=0;
        this->s2a2_cke->Checked=0; this->s2f1_cke->Checked=0;
        this->s2f2_cke->Checked=0; this->s2par_cke->Checked=0;
        this->s2impar_cke->Checked=0;

        this->s3a1_cke->Checked=1; this->s3b1_cke->Checked=1;

        this->s3c1_cke->Checked=1; this->s3b2_cke->Checked=1;
        this->s3c2_cke->Checked=1; this->s3f1_cke->Checked=1;
        this->s3f2_cke->Checked=1; this->s3par_cke->Checked=1;
        this->s3impar_cke->Checked=1;
        this->flagnoeventos=false; //se libera el flag inhibidor
de eventos

```

```

    }
    private: System::Void serie23_rb_MouseClick(System::Object^ sender,
System::Windows::Forms::EventArgs^ e) {
        this->serie2_rb->Checked=false;//Al activar rb de serie
23, desactivo los otros
        this->serie3_rb->Checked=false;

        this->flagnoeventos=true; //Se activan todos los ckb.
Para no disparar los eventos de cada checkbox al cambiar su valor, se
emplea el flag inhibidor de eventos
        this->s2a1_cke->Checked=1; this->s2b1_cke->Checked=1;
        this->s2a2_cke->Checked=1; this->s2f1_cke->Checked=1;
        this->s2f2_cke->Checked=1; this->s2par_cke->Checked=1;
        this->s2impar_cke->Checked=1;

        this->s3a1_cke->Checked=1; this->s3b1_cke->Checked=1;
        this->s3c1_cke->Checked=1; this->s3b2_cke->Checked=1;
        this->s3c2_cke->Checked=1; this->s3f1_cke->Checked=1;
        this->s3f2_cke->Checked=1; this->s3par_cke->Checked=1;
        this->s3impar_cke->Checked=1;
        this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
    }

    //Lo que ocurre al activarse un rb secundario MANUALMENTE
    private: System::Void serie2_rb2_MouseClick(System::Object^ sender,
System::Windows::Forms::EventArgs^ e) {
        this->serie3_rb2->Checked=false; //Al activar rb de
serie 2, desactivo los otros
        this->serie23_rb2->Checked=false;

        this->flagnoeventos=true; //Se desactivan las ckb de
serie 3 y se activan las de serie 2. Para no disparar los eventos de cada
checkbox al cambiar su valor, se emplea el flag inhibidor de eventos
        this->s3a1_cke2->Checked=0; this->s3b1_cke2->Checked=0;

        this->s3c1_cke2->Checked=0; this->s3b2_cke2->Checked=0;
        this->s3c2_cke2->Checked=0; this->s3f1_cke2->Checked=0;
        this->s3f2_cke2->Checked=0; this->s3par_cke2->Checked=0;
        this->s3impar_cke2->Checked=0;

        this->s2a1_cke2->Checked=1; this->s2b1_cke2->Checked=1;
        this->s2a2_cke2->Checked=1; this->s2f1_cke2->Checked=1;
        this->s2f2_cke2->Checked=1; this->s2par_cke2->Checked=1;
        this->s2impar_cke2->Checked=1;
        this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
    }

    private: System::Void serie3_rb2_MouseClick(System::Object^ sender,
System::Windows::Forms::EventArgs^ e) {
        this->serie2_rb2->Checked=false;//Al activar rb de serie
3, desactivo los otros
        this->serie23_rb2->Checked=false;

        this->flagnoeventos=true; //Se desactivan las ckb de
serie 2 y se activan las de serie 3. Para no disparar los eventos de cada
checkbox al cambiar su valor, se emplea el flag inhibidor de eventos

```



```

        this->s2a1_ckb2->Checked=0; this->s2b1_ckb2->Checked=0;
        this->s2a2_ckb2->Checked=0; this->s2f1_ckb2->Checked=0;
        this->s2f2_ckb2->Checked=0; this->s2par_ckb2->Checked=0;
        this->s2impar_ckb2->Checked=0;

        this->s3a1_ckb2->Checked=1;   this->s3b1_ckb2->Checked=1;

        this->s3c1_ckb2->Checked=1; this->s3b2_ckb2->Checked=1;
        this->s3c2_ckb2->Checked=1; this->s3f1_ckb2->Checked=1;
        this->s3f2_ckb2->Checked=1; this->s3par_ckb2->Checked=1;
        this->s3impar_ckb2->Checked=1;
        this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
    }
    private: System::Void serie23_rb2_MouseClick(System::Object^ sender,
System::Windows::Forms::EventArgs^ e) {
        this->serie2_rb2->Checked=false; //Al activar rb de serie
23, desactivo los otros
        this->serie3_rb2->Checked=false;

        this->flagnoeventos=true; //Se activan todos los ckb.
Para no disparar los eventos de cada checkbox al cambiar su valor, se
emplea el flag inhibidor de eventos
        this->s2a1_ckb2->Checked=1; this->s2b1_ckb2->Checked=1;
        this->s2a2_ckb2->Checked=1; this->s2f1_ckb2->Checked=1;
        this->s2f2_ckb2->Checked=1; this->s2par_ckb2->Checked=1;
        this->s2impar_ckb2->Checked=1;

        this->s3a1_ckb2->Checked=1; this->s3b1_ckb2->Checked=1;
        this->s3c1_ckb2->Checked=1; this->s3b2_ckb2->Checked=1;
        this->s3c2_ckb2->Checked=1; this->s3f1_ckb2->Checked=1;
        this->s3f2_ckb2->Checked=1; this->s3par_ckb2->Checked=1;
        this->s3impar_ckb2->Checked=1;
        this->flagnoeventos=false; //se libera el flag inhibidor
de eventos
    }
    //Lo que ocurre al modificar la fecha mínima
    private: System::Void desde_dtp_ValueChanged(System::Object^ sender,
System::EventArgs^ e) {
        if (flagnoeventos==false) {
            this->comparativaTableAdapter->Fill(this-
>TablaGeneral->comparativa, this->desde_dtp->Value.Date, this->hasta_dtp-
>Value.Date); //Carga el dgv interface
            this->alertal_lab->Visible=1;
        }
    }
    //Lo que ocurre al modificar la fecha máxima
    private: System::Void hasta_dtp_ValueChanged(System::Object^ sender,
System::EventArgs^ e) {
        if (flagnoeventos==false) {
            this->comparativaTableAdapter->Fill(this-
>TablaGeneral->comparativa, this->desde_dtp->Value.Date, this->hasta_dtp-
>Value.Date); //Carga el dgv interface
            this->alertal_lab->Visible=1;
        }
    }
    //Lo que ocurre al pulsar el botón de actualizar gráfica

```

```

private: System::Void draw_but_Click(System::Object^ sender,
System::EventArgs^ e) {
    getarray();
    getarray2();
    ejes();
    this->alortal_lab->Visible=0;
}

//Lo que ocurre al modificar la edad mínima en el grupo 1
private: System::Void edaddesde_nud_ValueChanged(System::Object^
sender, System::EventArgs^ e) {
    this->alortal_lab->Visible=1;
}
//Lo que ocurre al modificar la edad máxima en el grupo 1
private: System::Void edadhasta_nud_ValueChanged(System::Object^
sender, System::EventArgs^ e) {
    this->alortal_lab->Visible=1;
}
//Lo que ocurre al modificar la edad mínima en el grupo 2
private: System::Void edaddesde_nud2_ValueChanged(System::Object^
sender, System::EventArgs^ e) {
    this->alortal_lab->Visible=1;
}
//Lo que ocurre al modificar la edad máxima en el grupo 2
private: System::Void edadhasta_nud2_ValueChanged(System::Object^
sender, System::EventArgs^ e) {
    this->alortal_lab->Visible=1;
}
private: System::Void var1_cb_SelectedIndexChanged(System::Object^
sender, System::EventArgs^ e) {
    this->alortal_lab->Visible=1;
}
private: System::Void var2_cb_SelectedIndexChanged(System::Object^
sender, System::EventArgs^ e) {
    this->alortal_lab->Visible=1;
}

//Guardar la gráfica
private: System::Void print_but_Click(System::Object^ sender,
System::EventArgs^ e) {
    this->principal_zgc->SaveAs();
}
//Guardar captura de pantalla
private: System::Void print2_but_Click(System::Object^ sender,
System::EventArgs^ e) {
    SaveFileDialog^ dialog = gcnew SaveFileDialog(); //Se
genera un diálogo de guardado
    dialog->Title = "Guardar captura de pantalla"; //título
del diálogo de guardado
    dialog->ShowDialog(); //Se abre un diálogo de guardado
    if (dialog->FileName != "") //Si se ha escrito un nombre
    {
        Bitmap^ captura; //Declaro el objeto captura para
almacenar el pantallazo
        Rectangle^ rectangulo; //Declaro el objeto
rectángulo para el pantallazo
    }
}

```

```

        captura = gcnew Bitmap(this->Width, this->Height);
        //Inicializo el bitmap al tamaño de la form
        rectangulo = gcnew Rectangle(0, 0, this->Width,
this->Height); //Inicializo el rectángulo del pantallazo
        this->DrawToBitmap(captura, *rectangulo); //Saco
el pantallazo y lo almaceno en captura
        captura->Save(String::Concat(dialog->FileName,
".png"), System::Drawing::Imaging::ImageFormat::Bmp); //Guardo el archivo
con el nombre que se le puso en el diálogo añadiendo .png al final. Además
asigno ese tipo de archivo
    }
};
}

```

## 2.4 Form4.h – Específico

Contiene el código de la ventana de análisis específico.

```

#pragma once

using namespace ZedGraph;
using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;

namespace menu {

public ref class Form4 : public System::Windows::Forms::Form
{
public:
    Form4(void)
    {
        InitializeComponent();
        //
        //TODO: Add the constructor code here
        //
    }

protected:
    ~Form4()
    {
        if (components)
        {
            delete components;
        }
    }

private: System::Windows::Forms::ListBox^ cubaspre_lb;
protected:

```

```

protected:

private: System::Windows::Forms::Button^ change_but;
private: System::Windows::Forms::Button^ changeback_but;
private: System::Windows::Forms::ListBox^ cubaspost_lb;
private: System::Windows::Forms::DataGridView^ total_dgv;
private: System::Windows::Forms::Button^ changebackall_but;
private: System::Windows::Forms::Button^ changeall_but;
private: System::Windows::Forms::Button^ changeall2_but;
private: System::Windows::Forms::Button^ changebackall2_but;
private: System::Windows::Forms::ListBox^ cubaspost2_lb;
private: System::Windows::Forms::Button^ changeback2_but;
private: System::Windows::Forms::Button^ change2_but;
private: System::Windows::Forms::ListBox^ cubaspre2_lb;
private: ZedGraph::ZedGraphControl^ graficocentral_zg;
private: System::Windows::Forms::BindingSource^
comparativa1BindingSource;
private: menu::TablaGeneralTableAdapters::comparativa1TableAdapter^
comparativa1TableAdapter;
private: menu::TablaGeneral^ TablaGeneral;
private: System::Windows::Forms::Label^ titulo_lab;
private: System::Windows::Forms::CheckBox^ media_cb;
private: System::Windows::Forms::CheckBox^ cubas_cb;
private: System::Windows::Forms::Button^ calcular_but;

private: System::Windows::Forms::Label^ notupdated_lab;
private: System::Windows::Forms::Label^ ok_lab;
private: menu::TablaGeneralTableAdapters::Pot_Day_Hist1TableAdapter^
pot_Day_Hist1TableAdapter1;
private: System::Windows::Forms::DateTimePicker^ from_dtp;
private: System::Windows::Forms::DateTimePicker^ to_dtp;
private: System::Windows::Forms::ComboBox^ variable_cb;
private: System::Windows::Forms::Label^ variable_lab;
private: System::Windows::Forms::RadioButton^ diarios_rb;
private: System::Windows::Forms::RadioButton^ semanales_rb;
private: System::Windows::Forms::TextBox^ flag;

private: System::Windows::Forms::Label^ selected_lab;

private: System::Windows::Forms::Label^ selected2_lab;
private: System::Windows::Forms::TextBox^ flag2;

```

```

private: System::Windows::Forms::Button^ printchart_but;
private: System::Windows::Forms::Button^ printscreen_but;
private: System::Windows::Forms::Label^ variable2_lab;

private: System::Windows::Forms::ComboBox^ variable2_cb;

private: System::Windows::Forms::CheckBox^ leyenda_ckb;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
dateTimeDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
potDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
ageDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
mEASSTPBATHXSALF3DataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalIronFeDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Bath_Calcium_Fluoride;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Metal_Silicon_Si;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Metal_Copper_Cu;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
measMetalLithiumLiDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Metal_Manganese_Mn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
measBathTemperatureControlDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
MEAS_STP_BATH_SUPERHEAT;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
noisSPNVoltsDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
AE_Num_Total;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
aETimeAbove8VoltsDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
fedFailuresDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Bath_Level_Target;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
measBathLevelDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Metal_Pad_Target;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Metal_DAS_Real;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
resCycleAvgVoltsDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
tappingWeightScheduleDataGridViewTextBoxColumn;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Meas_Add_Fluoride_Actual;
private: System::Windows::Forms::DataGridViewTextBoxColumn^
Nois_relnoise_FFT;

```

Aplicación de escritorio para visualización de datos de análisis del  
proceso electrolítico del aluminio – David González Aguado

Fecha: Febrero 2015

Aplicación de escritorio para visualización de datos de análisis del  
proceso electrolítico del aluminio – David González Aguado

Fecha: Febrero 2015

```
private: System::ComponentModel::IContainer^ components;
protected:
protected:
private:

        //Ajusta la fecha para que el tramo se divida en semanas
naturales
        void fija_fecha()
        {
                while (this->from_dtp-
>Value.DayOfWeek!=DayOfWeek::Monday)
                        this->from_dtp->Value=this->from_dtp-
>Value.AddDays(-1);
                while (this->to_dtp->Value.DayOfWeek!=DayOfWeek::Sunday)
                        this->to_dtp->Value=this->to_dtp-
>Value.AddDays(+1);
        }

        //Enciende mensaje ok y apaga el resto
        void luz_verde()
        {
                this->notupdated_lab->Visible=0;
                this->ok_lab->Visible=1;
        }

        //Enciende mensaje not updated y apaga el resto
        void luz_roja()
        {
```



```

        this->notupdated_lab->Visible=1;
        this->ok_lab->Visible=0;
    }

    //Función que cambia items de izda a dcha
    void add()
    {
        if(this->cubaspre_lb->SelectedItem != nullptr)
        {
            this->cubaspost_lb->Items->Add(this->cubaspre_lb-
>SelectedItem->ToString());
            this->cubaspre_lb->Items->Remove(this->cubaspre_lb-
>SelectedItem);
        }
    }

    //Función que elimina items de la derecha
    void remove()
    {
        if(this->cubaspost_lb->SelectedItem != nullptr)
        {
            this->cubaspre_lb->Items->Add(this->cubaspost_lb-
>SelectedItem->ToString());
            this->cubaspost_lb->Items->Remove(this-
>cubaspost_lb->SelectedItem);
        }
    }

    //Función que añade TODOS los items de la izquierda a la
derecha
    void add_all()
    {
        for(int i=0; i<this->cubaspre_lb->Items->Count;i++)
            this->cubaspost_lb->Items->Add(this->cubaspre_lb-
>Items[i]->ToString());
        this->cubaspre_lb->Items->Clear();
    }

    //Función que elimina TODOS los items de la derecha
    void remove_all()
    {
        for(int i=0; i<this->cubaspost_lb->Items->Count;i++)
            this->cubaspre_lb->Items->Add(this->cubaspost_lb-
>Items[i]->ToString());
        this->cubaspost_lb->Items->Clear();
    }

    //Función que cambia items de izda a dcha
    void add2()
    {
        if(this->cubaspre2_lb->SelectedItem != nullptr)
        {
            this->cubaspost2_lb->Items->Add(this->cubaspre2_lb-
>SelectedItem->ToString());

```

```

        this->cubaspre2_lb->Items->Remove(this->
>cubaspre2_lb->SelectedItem);
    }
}

//Función que elimina items de la derecha
void remove2()
{
    if(this->cubaspost2_lb->SelectedItem != nullptr)
    {
        this->cubaspre2_lb->Items->Add(this->cubaspost2_lb->
>SelectedItem->ToString());
        this->cubaspost2_lb->Items->Remove(this->
>cubaspost2_lb->SelectedItem);
    }
}

//Función que añade TODOS los items de la izquierda a la
derecha
void add_all2()
{
    for(int i=0; i<this->cubaspre2_lb->Items->Count;i++)
        this->cubaspost2_lb->Items->Add(this->cubaspre2_lb->
>Items[i]->ToString());
    this->cubaspre2_lb->Items->Clear();
}

//Función que elimina TODOS los items de la derecha
void remove_all2()
{
    for(int i=0; i<this->cubaspost2_lb->Items->Count;i++)
        this->cubaspre2_lb->Items->Add(this->cubaspost2_lb->
>Items[i]->ToString());
    this->cubaspost2_lb->Items->Clear();
}

//Función que calcula los datos para representar las curvas
void curvas_dias()
{
    int i=0;
    int cubas=1; //columna de cubas del dgv
    int selected = variable_cb->SelectedIndex+3; //Columna
seleccionada en el cb
    int numero=0; //para la flag de pruebas
    LineItem ^myCurve1; //Para la representación de las cubas
individuales
    LineItem ^myCurve2; //Para la media
    Random ^rand = gcnew Random(); //Objeto necesario para
generar números aleatorios
    //Inicializa parámetros de zedgraph
    this->graficocentral_zg->GraphPane->CurveList->Clear();
    this->graficocentral_zg->Invalidate();

    //Arrays para cuba individual y para media de cubas y
contador para avanzar por ellos

```

```

        array<double>^ media_arr = gcnew array<double>((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La longitud del array es la diferencia de días entre desde y hasta +1
        array<double>^ mediaaux_arr = gcnew array<double>((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La longitud del array es la diferencia de días entre desde y hasta +1
        array<double>^ cuba_arr = gcnew array<double>((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La longitud del array es la diferencia de días entre desde y hasta +1
        int contador=0;
        //Arrays para datos semanales
        array<double>^ cubasem_arr = gcnew array<double>(((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La longitud del array es la diferencia de días entre desde y hasta +1
        array<double>^ cubaauxsem_arr = gcnew array<double>(((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La longitud del array es la diferencia de días entre desde y hasta +1
        array<double>^ mediasem_arr = gcnew array<double>(((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La longitud del array es la diferencia de días entre desde y hasta +1
        array<double>^ mediaauxsem_arr = gcnew array<double>(((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La longitud del array es la diferencia de días entre desde y hasta +1
        //Arrays de leyenda
        array<String^>^ labels = gcnew array<String^>((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La longitud del array es la diferencia de días entre desde y hasta +1
        array<String^>^ labelssem = gcnew array<String^>(((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La longitud del array es la diferencia de días entre desde y hasta +1

        //Se inicializan los arrays de datos diarios (siempre, porque se usan para todo)
        for (int k=0;k<((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1);k++)
        {
            media_arr[k]=Double::NaN;
            mediaaux_arr[k]=0.0;
            labels[k]= this->from_dtp->Value.Date.AddDays(k).Day.ToString(); //A la fecha desde se le suma la posición del vector y se almacena
        }

        //Si vamos a usar datos semanales se inicializan los arrays y se fija el eje X
        if (this->semanales_rb->Checked==1)
        {
            for (int k=0;k<(((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7);k++)
            {
                mediasem_arr[k]=Double::NaN;
                mediaauxsem_arr[k]=0.0;
                labelssem[k]= (k+1).ToString();
            }
        }

```

```

        this->graficocentral_zg->GraphPane->XAxis->Scale-
>TextLabel=labelssem; //Coloco el array labels en el eje x
        this->graficocentral_zg->GraphPane->XAxis->Type=
AxisType::Text; //Y cambio el tipo de eje a eje de texto
    }

    //Si vamos a usar datos diarios se fija el eje X
    if (this->diarios_rb->Checked==1)
    {
        this->graficocentral_zg->GraphPane->XAxis->Scale-
>TextLabel=labels; //Coloco el array labels en el eje x
        this->graficocentral_zg->GraphPane->XAxis->Type=
AxisType::Text; //Y cambio el tipo de eje a eje de texto
    }

    //Inicio del algoritmo de cálculo y representación
    for (int j=0; j<cubaspost_lb->Items->Count;j++) //Desde
la primera a la última cuba seleccionada
    {
        //Al comenzar con cada cuba se inicializa a NaN el
array
        for (int k=0;k<((to_dtp->Value.Date-from_dtp-
>Value.Date).TotalDays+1);k++)
        {
            cuba_arr[k]=Double::NaN;
            contador=0;
        }
        //Al comenzar con cada cuba se inicializa a Nan el
array de semana si está activada la casilla semana
        if (this->semanales_rb->Checked==1)
        {
            for (int k=0;k<(((this->to_dtp->Value.Date-
this->from_dtp->Value.Date).TotalDays+1)/7);k++)
            {
                cubasem_arr[k]=Double::NaN;
                cubauxsem_arr[k]=0.0;
            }
        }
        for (i; i<total_dgv->Rows->Count-1;i++) //Se avanza
por el dgv buscando coincidencia de cuba en cada linea
        {
            //Si coincide el número de cuba
            if (total_dgv->Rows[i]->Cells[cubas]->Value-
>ToString()==cubaspost_lb->Items[j]->ToString())
            {
                //Rellenado del array de cuba individual
                if(total_dgv->Rows[i]->Cells[selected]-
>Value!= DBNull::Value)//Si la variable seleccionada tiene valor no nulo
                {
                    cuba_arr[contador]=Convert::ToDouble(total_dgv->Rows[i]-
>Cells[selected]->Value); //Se guarda en el array
                }
                else if(contador>0) //Si es nulo, pero
al menos no es el primero

```

```

        cuba_arr[contador]=cuba_arr[contador-1]; //Se guarda el anterior
valor
//Rellenado del array de media (si
procede)
        if(this->media_cb->Checked==1)
        {
                if(Double::IsNaN(media_arr[contador])!=0           &&
Double::IsNaN(cuba_arr[contador])!=0)
                {
                        media_arr[contador]=media_arr[contador]+cuba_arr[contador];
                                                                mediaaux_arr[contador]++;
                }
                else
if(Double::IsNaN(cuba_arr[contador])!=0)
                {
                        media_arr[contador]=cuba_arr[contador];
                                                                mediaaux_arr[contador]++;
                }
                }
        contador++; //Una vez rellena la casilla
(o no), se mueve a la siguiente
//Si es el último día de la cuba
        if (total_dgv->Rows[i]->Cells[cubas]-
>Value->Equals(total_dgv->Rows[i+1]->Cells[cubas]->Value)!=0) //Si además
de coincidir el número de cuba, la siguiente línea ya no coincide, se han
terminado los datos para ella
        {
                //Representación de cubas
individuales (si procede)
                if (this->cubas_cb->Checked==1 &&
this->diarios_rb->Checked==1)
                {
                        myCurve1 = this-
>graficocentral_zg->GraphPane->AddCurve(total_dgv->Rows[i]->Cells[cubas]-
>Value->ToString(), nullptr, cuba_arr, Color::FromArgb(rand->Next(0, 255),
rand->Next(0, 255), rand->Next(0, 255)), SymbolType::None); //El color de
cada línea de cuba es aleatorio
                                                                myCurve1->Line->Width=2;
                                                                this->graficocentral_zg-
>AxisChange();
                }
                //Cálculo de cubas individuales
para datos semanales
                if (this->semanales_rb-
>Checked==1) //Cálculo de datos de cuba por semanas si está activada la
casilla semanas
                {
                        for (int k=0;k<((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1);k++)
                {

```

```

                                                                    if
(Double::IsNaN(cubasem_arr[k/7])==0 && Double::IsNaN(cuba_arr[k])==0)
{
    cubasem_arr[k/7]=cubasem_arr[k/7]+cuba_arr[k];
    cubaauxsem_arr[k/7]++;
}
else if
(! (Double::IsNaN(cuba_arr[k])))
{
    cubasem_arr[k/7]=cuba_arr[k];
    cubaauxsem_arr[k/7]++;
}
}
for (int k=0;k<(((this->to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7);k++)
{
    if
(cubaauxsem_arr[k]>0)
    cubasem_arr[k]=cubasem_arr[k]/cubaauxsem_arr[k];
}
if (this->cubas_cb-
>Checked==1) //Se representa la cuba por semanas si está activada la
casilla cubas
{
    myCurve1 = this-
>graficocentral_zg->GraphPane->AddCurve(total_dgv->Rows[i]->Cells[cubas]-
>Value->ToString(), nullptr, cubasem_arr, Color::FromArgb(rand->Next(0,
255), rand->Next(0, 255), rand->Next(0, 255)), SymbolType::None); //El
color de cada linea de cuba es aleatorio
    myCurve1->Line-
>Width=2;
    this-
>graficocentral_zg->AxisChange();
}
//Cálculo de media semanal
if (this->media_cb-
>Checked==1)
{
    for (int k=0;
k<(((this->to_dtp->Value.Date-this->from_dtp-
>Value.Date).TotalDays+1)/7);k++)
    {
        if(Double::IsNaN(mediasem_arr[k])==0 &&
Double::IsNaN(cubasem_arr[k])==0)
        {
            mediasem_arr[k]=mediasem_arr[k]+cubasem_arr[k];
            mediaauxsem_arr[k]++;
        }
    }
}

```

```

else
if(Double::IsNaN(cubasem_arr[k])==0)
{
    mediasem_arr[k]=cubasem_arr[k];
    mediaauxsem_arr[k]++;
}
}
}
}
numero++; //Recuento de cubas
seleccionadas
break; //Se deja de recorrer el
dgv porque ya no va a haber más coincidencias. Al no reiniciarse i, la
siguiente cuba seleccionada comenzará a buscar desde donde terminó esta
}
}
}
}
//Calculos finales y representación del array de la media
(si procede)
if(this->media_cb->Checked==1 && this->diarios_rb-
>Checked==1)
{
    //Se realiza la división de la suma acumulada de
    los valores de las cubas para cada día entre el número de cubas
    for (int k=0; k<((this->to_dtp->Value.Date-this-
>from_dtp->Value.Date).TotalDays+1);k++)
    {
        if (mediaaux_arr[k]>0) //Solo si hay al menos
un valor de una cuba
            media_arr[k]=
media_arr[k]/mediaaux_arr[k];
    }
    //Se representa la linea de media
myCurve2 = this->graficocentral_zg->GraphPane-
>AddCurve("Media A", nullptr, media_arr, Color::Green, SymbolType::Circle);
myCurve2->Line->Width=4; //Aumento el grosor de la
linea de media para que destaque
myCurve2->Symbol->Fill= gcnew Fill(Color::Black);
//Fijo el color de símbolo
myCurve2->Symbol->Size=6; //Fijo el tamaño de
símbolo
this->graficocentral_zg->AxisChange();
}
//Cálculos finales y representación del array de la media
semanal (si procede)
if(this->media_cb->Checked==1 && this->semanales_rb-
>Checked==1)
{

```

```

        for(int k=0;k<(((this->to_dtp->Value.Date-this-
>from_dtp->Value.Date).TotalDays+1)/7);k++)
        {
            if(mediaauxsem_arr[k]>0)

                mediasem_arr[k]=mediasem_arr[k]/mediaauxsem_arr[k];
            }
            //Se representa la linea de media semanal
            myCurve2=this->graficocentral_zg->GraphPane-
>AddCurve("Media A", nullptr, mediasem_arr, Color::Green,
SymbolType::Circle);
            myCurve2->Line->Width=4; //Aumento el grosor de la
linea de media para que destaque
            myCurve2->Symbol->Fill= gcnew Fill(Color::Black);
//Fijo el color de símbolo
            this->graficocentral_zg->AxisChange();
        }
        //Cambio títulos y nombres de ejes
        this->graficocentral_zg->GraphPane->Title-
>Text=String::Concat("Gráfico de ",this->variable_cb->Text);
        if (this->semanales_rb->Checked==1)
            this->graficocentral_zg->GraphPane->XAxis->Title-
>Text="Semana";
        else if(this->diarios_rb->Checked==1)
            this->graficocentral_zg->GraphPane->XAxis->Title-
>Text="Día";
            this->graficocentral_zg->GraphPane->YAxis->Title-
>Text=this->variable_cb->Text;
            this->graficocentral_zg->GraphPane->Y2Axis->Title-
>Text=this->variable2_cb->Text;

            //Muestro el numero de cubas por la flag
            this->flag->Text=System::Convert::ToString(numero);

            //Fin de la función
        }

        //Función que calcula los datos para representar las curvas de
la selección B
        void curvas_dias2()
        {
            int i=0;
            int cubas=1; //columna de cubas del dgv
            int selected = variable2_cb->SelectedIndex+3; //Columna
seleccionada en el cb
            int numero=0; //para la flag de pruebas
            LineItem ^myCurve3; //Para la representación de las cubas
individuales
            LineItem ^myCurve4; //Para la media
            Random ^rand2 = gcnew Random(); //Objeto necesario para
generar números aleatorios
            //Inicializa parámetros de zedgraph
            //this->graficocentral_zg->GraphPane->CurveList->Clear();
            //this->graficocentral_zg->Invalidate();

```



```

//Arrays para cuba individual y para media de cubas y
contador para avanzar por ellos
array<double>^ media_arr = gcnw array<double>((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La
longitud del array es la diferencia de días entre desde y hasta +1
array<double>^ mediaaux_arr = gcnw array<double>((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La
longitud del array es la diferencia de días entre desde y hasta +1
array<double>^ cuba_arr = gcnw array<double>((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La
longitud del array es la diferencia de días entre desde y hasta +1
int contador=0;
//Arrays para datos semanales
array<double>^ cubasem_arr = gcnw array<double>(((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La
longitud del array es la diferencia de días entre desde y hasta +1
array<double>^ cubauxsem_arr = gcnw
array<double>(((this->to_dtp->Value.Date-this->from_dtp-
>Value.Date).TotalDays+1)/7); //La longitud del array es la diferencia de
días entre desde y hasta +1
array<double>^ mediasem_arr = gcnw array<double>(((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La
longitud del array es la diferencia de días entre desde y hasta +1
array<double>^ mediaauxsem_arr = gcnw
array<double>(((this->to_dtp->Value.Date-this->from_dtp-
>Value.Date).TotalDays+1)/7); //La longitud del array es la diferencia de
días entre desde y hasta +1
//Arrays de leyenda
array<String>^ labels = gcnw array<String>((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1); //La
longitud del array es la diferencia de días entre desde y hasta +1
array<String>^ labelssem = gcnw array<String>(((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7); //La
longitud del array es la diferencia de días entre desde y hasta +1

//Se inicializan los arrays de datos diarios (siempre,
porque se usan para todo)
for (int k=0;k<((this->to_dtp->Value.Date-this->from_dtp-
>Value.Date).TotalDays+1);k++)
{
    media_arr[k]=Double::NaN;
    mediaaux_arr[k]=0.0;
    labels[k]= this->from_dtp-
>Value.Date.AddDays(k).Day.ToString(); //A la fecha desde se le suma la
posición del vector y se almacena
}

//Si vamos a usar datos semanales se inicializan los
arrays y se fija el eje X
if (this->semanales_rb->Checked==1)
{
    for (int k=0;k<(((this->to_dtp->Value.Date-this-
>from_dtp->Value.Date).TotalDays+1)/7);k++)
    {
        mediasem_arr[k]=Double::NaN;
        mediaauxsem_arr[k]=0.0;
        labelssem[k]= (k+1).ToString();
    }
}

```

```

    }

    this->graficocentral_zg->GraphPane->XAxis->Scale-
>TextLabels=labelssem; //Coloco el array labels en el eje x
    this->graficocentral_zg->GraphPane->XAxis->Type=
AxisType::Text; //Y cambio el tipo de eje a eje de texto
    }

    //Si vamos a usar datos diarios se fija el eje X
    if (this->diarios_rb->Checked==1)
    {
        this->graficocentral_zg->GraphPane->XAxis->Scale-
>TextLabels=labels; //Coloco el array labels en el eje x
        this->graficocentral_zg->GraphPane->XAxis->Type=
AxisType::Text; //Y cambio el tipo de eje a eje de texto
    }

    //Inicio del algoritmo de cálculo y representación
    for (int j=0; j<cubaspost2_lb->Items->Count;j++) //Desde
la primera a la última cuba seleccionada
    {
        //Al comenzar con cada cuba se inicializa a NaN el
array
        for (int k=0;k<((to_dtp->Value.Date-from_dtp-
>Value.Date).TotalDays+1);k++)
        {
            cuba_arr[k]=Double::NaN;
            contador=0;
        }
        //Al comenzar con cada cuba se inicializa a Nan el
array de semana si está activada la casilla semana
        if (this->semanales_rb->Checked==1)
        {
            for (int k=0;k<(((this->to_dtp->Value.Date-
this->from_dtp->Value.Date).TotalDays+1)/7);k++)
            {
                cubasem_arr[k]=Double::NaN;
                cubauxsem_arr[k]=0.0;
            }
        }
        for (i; i<total_dgv->Rows->Count-1;i++) //Se avanza
por el dgv buscando coincidencia de cuba en cada linea
        {
            //Si coincide el número de cuba
            if (total_dgv->Rows[i]->Cells[cubas]->Value-
>ToString()==cubaspost2_lb->Items[j]->ToString())
            {
                //Rellenado del array de cuba individual
                if (total_dgv->Rows[i]->Cells[selected]-
>Value!= DBNull::Value)//Si la variable seleccionada tiene valor no nulo
                {
                    cuba_arr[contador]=Convert::ToDouble(total_dgv->Rows[i]-
>Cells[selected]->Value); //Se guarda en el array
                }
                else if (contador>0) //Si es nulo, pero
al menos no es el primero

```

```

        cuba_arr[contador]=cuba_arr[contador-1]; //Se guarda el anterior
valor
//Rellenado del array de media (si
procede)
        if(this->media_cb->Checked==1)
        {
                if(Double::IsNaN(media_arr[contador])!=0           &&
Double::IsNaN(cuba_arr[contador])!=0)
                {
                        media_arr[contador]=media_arr[contador]+cuba_arr[contador];
                                                                mediaaux_arr[contador]++;
                }
                else
if(Double::IsNaN(cuba_arr[contador])!=0)
                {
                        media_arr[contador]=cuba_arr[contador];
                                                                mediaaux_arr[contador]++;
                }
                contador++; //Una vez rellena la casilla
(o no), se mueve a la siguiente
//Si es el último día de la cuba
        if (total_dgv->Rows[i]->Cells[cubas]-
>Value->Equals(total_dgv->Rows[i+1]->Cells[cubas]->Value)!=0) //Si además
de coincidir el número de cuba, la siguiente línea ya no coincide, se han
terminado los datos para ella
        {
                //Representación de cubas
individuales (si procede)
                if (this->cubas_cb->Checked==1 &&
this->diarios_rb->Checked==1)
                {
                        myCurve3 = this-
>graficocentral_zg->GraphPane->AddCurve(total_dgv->Rows[i]->Cells[cubas]-
>Value->ToString(), nullptr, cuba_arr, Color::FromArgb(255-rand2->Next(0,
255), 255-rand2->Next(0, 255), 255-rand2->Next(0, 255)), SymbolType::None);
//El color de cada línea de cuba es aleatorio
                        myCurve3->Line->Width=2;
                        if (this->variable_cb-
>SelectedIndex==this->variable2_cb->SelectedIndex)
                        {
                                this-
>graficocentral_zg->GraphPane->Y2Axis->IsVisible=0;
                                                                myCurve3->IsY2Axis=0;
                        }
                        else
                        {
                                this-
>graficocentral_zg->GraphPane->Y2Axis->IsVisible=1;
                                                                myCurve3->IsY2Axis=1;

```

```

}
this->graficocentral_zg-
>AxisChange();
}
//Cálculo de cubas individuales
para datos semanales
if (this->semanales_rb-
>Checked==1) //Cálculo de datos de cuba por semanas si está activada la
casilla semanas
{
for (int k=0;k<((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1);k++)
{
if
(Double::IsNaN(cubasem_arr[k/7])==0 && Double::IsNaN(cuba_arr[k])==0)
{
cubasem_arr[k/7]=cubasem_arr[k/7]+cuba_arr[k];
cubaauxsem_arr[k/7]++;
}
else if
(! (Double::IsNaN(cuba_arr[k])))
{
cubasem_arr[k/7]=cuba_arr[k];
cubaauxsem_arr[k/7]++;
}
}
for (int k=0;k<(((this-
>to_dtp->Value.Date-this->from_dtp->Value.Date).TotalDays+1)/7);k++)
{
if
(cubaauxsem_arr[k]>0)
{
cubasem_arr[k]=cubasem_arr[k]/cubaauxsem_arr[k];
}
if (this->cubas_cb-
>Checked==1) //Se representa la cuba por semanas si está activada la
casilla cubas
{
myCurve3 = this-
>graficocentral_zg->GraphPane->AddCurve(total_dgv->Rows[i]->Cells[cubas]-
>Value->ToString(), nullptr, cubasem_arr, Color::FromArgb(rand2->Next(0,
255), rand2->Next(0, 255), rand2->Next(0, 255)), SymbolType::None); //El
color de cada linea de cuba es aleatorio
myCurve3->Line-
Width=2;
if (this->variable_cb-
>SelectedIndex==this->variable2_cb->SelectedIndex)
{
this-
myCurve3-
}
}
}
}

```

```

else
{
    this-
    myCurve3-
}
this-
}
//Cálculo de media semanal
if (this->media_cb-
{
    for (int k=0;
    {
        if(Double::IsNaN(mediasem_arr[k])==0 &&
        Double::IsNaN(cubasem_arr[k])==0)
        {
            mediasem_arr[k]=mediasem_arr[k]+cubasem_arr[k];
            mediaauxsem_arr[k]++;
        }
        else
        {
            mediasem_arr[k]=cubasem_arr[k];
            mediaauxsem_arr[k]++;
        }
    }
}
numero++; //Recuento de cubas
seleccionadas
break; //Se deja de recorrer el
dgv porque ya no va a haber más coincidencias. Al no reiniciarse i, la
siguiente cuba seleccionada comenzará a buscar desde donde terminó esta
}
}
}
//Calculos finales y representación del array de la media
(si procede)
if(this->media_cb->Checked==1 && this->diarios_rb-
>Checked==1)
{

```

```

//Se realiza la división de la suma acumulada de
los valores de las cubas para cada día entre el número de cubas
for (int k=0; k<((this->to_dtp->Value.Date-this-
>from_dtp->Value.Date).TotalDays+1);k++)
{
    if (mediaaux_arr[k]>0) //Solo si hay al menos
un valor de una cuba
        media_arr[k]=
media_arr[k]/mediaaux_arr[k];
}
//Se representa la linea de media
myCurve4 = this->graficocentral_zg->GraphPane-
>AddCurve("Media B", nullptr, media_arr, Color::Blue, SymbolType::Circle);
myCurve4->Line->Width=4; //Aumento el grosor de la
linea de media para que destaque
if (this->variable_cb->SelectedIndex==this-
>variable2_cb->SelectedIndex)
{
    this-
>graficocentral_zg->GraphPane->Y2Axis->IsVisible=0;
    myCurve4->IsY2Axis=0;
}
else
{
    this-
>graficocentral_zg->GraphPane->Y2Axis->IsVisible=1;
    myCurve4->IsY2Axis=1;
}
myCurve4->Symbol->Fill= gcnew Fill(Color::Black);
//Fijo el color de símbolo
myCurve4->Symbol->Size=6; //Fijo el tamaño de
símbolo
this->graficocentral_zg->AxisChange();
}

//Cálculos finales y representación del array de la media
semanal (si procede)
if(this->media_cb->Checked==1 && this->semanales_rb-
>Checked==1)
{
    for(int k=0;k<(((this->to_dtp->Value.Date-this-
>from_dtp->Value.Date).TotalDays+1)/7);k++)
    {
        if(mediaauxsem_arr[k]>0)

        mediasem_arr[k]=mediasem_arr[k]/mediaauxsem_arr[k];
    }
    //Se representa la linea de media semanal
    myCurve4=this->graficocentral_zg->GraphPane-
>AddCurve("Media", nullptr, mediasem_arr, Color::Blue, SymbolType::Circle);
    myCurve4->Line->Width=4; //Aumento el grosor de la
linea de media para que destaque
    if (this->variable_cb->SelectedIndex==this-
>variable2_cb->SelectedIndex)
        {

```

```

>graficocentral_zg->GraphPane->Y2Axis->IsVisible=0;
this-
myCurve4->IsY2Axis=0;
}
else
{
this-
>graficocentral_zg->GraphPane->Y2Axis->IsVisible=1;
myCurve4->IsY2Axis=1;
}
myCurve4->Symbol->Fill= gcnew Fill(Color::Black);
//Fijo el color de símbolo
this->graficocentral_zg->AxisChange();
}
/*
//Cambio títulos y nombres de ejes
this->graficocentral_zg->GraphPane->Title-
>Text=String::Concat("Gráfico de ",this->variable_cb->Text);
if (this->semanales_rb->Checked==1)
this->graficocentral_zg->GraphPane->XAxis->Title-
>Text="Semana";
else if(this->diarios_rb->Checked==1)
this->graficocentral_zg->GraphPane->XAxis->Title-
>Text="Día";
this->graficocentral_zg->GraphPane->YAxis->Title-
>Text=this->variable_cb->Text;
*/
//Muestro el numero de cubas por la flag
this->flag2->Text=System::Convert::ToString(numero);

//Fin de la función
}
#pragma region Windows Form Designer generated code

void InitializeComponent(void)
{
    this->components = (gcnew
System::ComponentModel::Container());
    this->cubaspre_lb = (gcnew
System::Windows::Forms::ListBox());
    this->change_but = (gcnew
System::Windows::Forms::Button());
    this->changeback_but = (gcnew
System::Windows::Forms::Button());
    this->cubaspost_lb = (gcnew
System::Windows::Forms::ListBox());
    this->total_dgv = (gcnew
System::Windows::Forms::DataGridView());
    this->changebackall_but = (gcnew
System::Windows::Forms::Button());
    this->changeall_but = (gcnew
System::Windows::Forms::Button());
    this->changeall2_but = (gcnew
System::Windows::Forms::Button());
    this->changebackall2_but = (gcnew
System::Windows::Forms::Button());
}

```

```

        this->cubaspost2_lb = (gcnew
System::Windows::Forms::ListBox());
        this->changeback2_but = (gcnew
System::Windows::Forms::Button());
        this->change2_but = (gcnew
System::Windows::Forms::Button());
        this->cubaspre2_lb = (gcnew
System::Windows::Forms::ListBox());
        this->graficocentral_zg = (gcnew
ZedGraph::ZedGraphControl());
        this->titulo_lab = (gcnew
System::Windows::Forms::Label());
        this->media_cb = (gcnew
System::Windows::Forms::CheckBox());
        this->cubas_cb = (gcnew
System::Windows::Forms::CheckBox());
        this->calcular_but = (gcnew
System::Windows::Forms::Button());
        this->notupdated_lab = (gcnew
System::Windows::Forms::Label());
        this->ok_lab = (gcnew System::Windows::Forms::Label());
        this->from_dtp = (gcnew
System::Windows::Forms::DateTimePicker());
        this->to_dtp = (gcnew
System::Windows::Forms::DateTimePicker());
        this->variable_cb = (gcnew
System::Windows::Forms::ComboBox());
        this->variable_lab = (gcnew
System::Windows::Forms::Label());
        this->diarios_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->semanales_rb = (gcnew
System::Windows::Forms::RadioButton());
        this->flag = (gcnew System::Windows::Forms::TextBox());
        this->selected_lab = (gcnew
System::Windows::Forms::Label());
        this->selected2_lab = (gcnew
System::Windows::Forms::Label());
        this->flag2 = (gcnew System::Windows::Forms::TextBox());
        this->printchart_but = (gcnew
System::Windows::Forms::Button());
        this->printscreen_but = (gcnew
System::Windows::Forms::Button());
        this->variable2_lab = (gcnew
System::Windows::Forms::Label());
        this->variable2_cb = (gcnew
System::Windows::Forms::ComboBox());
        this->leyenda_ckb = (gcnew
System::Windows::Forms::CheckBox());
        this->comparativa1BindingSource = (gcnew
System::Windows::Forms::BindingSource(this->components));
        this->TablaGeneral = (gcnew menu::TablaGeneral());
        this->comparativa1TableAdapter = (gcnew
menu::TablaGeneralTableAdapters::comparativa1TableAdapter());
        this->pot_Day_Hist1TableAdapter1 = (gcnew
menu::TablaGeneralTableAdapters::Pot_Day_Hist1TableAdapter());

```



```

        this->dateTimeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->potDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->ageDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalIronFeDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Bath_Calcium_Fluoride = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Metal_Silicon_Si = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Metal_Copper_Cu = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measMetalLithiumLiDataGridViewTextBoxColumn =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Metal_Manganese_Mn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measBathTemperatureControlDataGridViewTextBoxColumn
= (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->MEAS_STP_BATH_SUPERHEAT = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->noisSPPNVoltsDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->AE_Num_Total = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->fedFailuresDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Bath_Level_Target = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->measBathLevelDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Metal_Pad_Target = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Metal_DAS_Real = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->resCycleAvgVoltsDataGridViewTextBoxColumn = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->tappingWeightScheduleDataGridViewTextBoxColumn =
(gcnew System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Meas_Add_Fluoride_Actual = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());
        this->Nois_relnoise_FFT = (gcnew
System::Windows::Forms::DataGridViewTextBoxColumn());

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>(this->total_dgv))->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>(this->comparativa1BindingSource))->BeginInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^
>(this->TablaGeneral))->BeginInit();

```

```

        this->SuspendLayout();
        //
        // cubaspre_lb
        //
        this->cubaspre_lb->FormattingEnabled = true;
        this->cubaspre_lb->Location = System::Drawing::Point(13,
164);

        this->cubaspre_lb->Name = L"cubaspre_lb";
        this->cubaspre_lb->Size = System::Drawing::Size(41, 810);
        this->cubaspre_lb->Sorted = true;
        this->cubaspre_lb->TabIndex = 0;
        this->cubaspre_lb->DoubleClick += gcnew
System::EventHandler(this, &Form4::cubaspre_lb_DoubleClick);
        //
        // change_but
        //
        this->change_but->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->change_but->Location = System::Drawing::Point(60,
200);

        this->change_but->Name = L"change_but";
        this->change_but->Size = System::Drawing::Size(40, 23);
        this->change_but->TabIndex = 2;
        this->change_but->Text = L">";
        this->change_but->UseVisualStyleBackColor = true;
        this->change_but->Click += gcnew
System::EventHandler(this, &Form4::change_but_Click);
        //
        // changeback_but
        //
        this->changeback_but->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->changeback_but->Location =
System::Drawing::Point(60, 229);
        this->changeback_but->Name = L"changeback_but";
        this->changeback_but->Size = System::Drawing::Size(40,
23);

        this->changeback_but->TabIndex = 3;
        this->changeback_but->Text = L"<";
        this->changeback_but->UseVisualStyleBackColor = true;
        this->changeback_but->Click += gcnew
System::EventHandler(this, &Form4::changeback_but_Click);
        //
        // cubaspost_lb
        //
        this->cubaspost_lb->FormattingEnabled = true;
        this->cubaspost_lb->Location =
System::Drawing::Point(106, 164);
        this->cubaspost_lb->Name = L"cubaspost_lb";
        this->cubaspost_lb->Size = System::Drawing::Size(41,
810);

        this->cubaspost_lb->Sorted = true;
        this->cubaspost_lb->TabIndex = 4;

```

```

        this->cubaspost_lb->DoubleClick += gcnew
System::EventHandler(this, &Form4::cubaspost_lb_DoubleClick);
        //
        // total_dgv
        //
        this->total_dgv->AutoGenerateColumns = false;
        this->total_dgv->ColumnHeadersHeightSizeMode =
System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;
        this->total_dgv->Columns->AddRange(gcnew cli::array<
System::Windows::Forms::DataGridViewColumn^ >(24) {this-
>dateTimeDataGridViewTextBoxColumn,
        this->potDataGridViewTextBoxColumn, this-
>ageDataGridViewTextBoxColumn, this-
>mEASSTPBATHXSALF3DataGridViewTextBoxColumn, this-
>measMetalIronFeDataGridViewTextBoxColumn,
        this->Meas_Bath_Calcium_Fluoride, this-
>Meas_Metal_Silicon_Si, this->Meas_Metal_Copper_Cu, this-
>measMetalLithiumLiDataGridViewTextBoxColumn,
        this->Meas_Metal_Manganese_Mn, this-
>measBathTemperatureControlDataGridViewTextBoxColumn, this-
>MEAS_STP_BATH_SUPERHEAT, this->noisSPPNVoltsDataGridViewTextBoxColumn,
        this->AE_Num_Total, this-
>aETimeAbove8VoltsDataGridViewTextBoxColumn, this-
>fedFailuresDataGridViewTextBoxColumn, this->Meas_Bath_Level_Target,
        this->measBathLevelDataGridViewTextBoxColumn, this-
>Meas_Metal_Pad_Target, this->Meas_Metal_DAS_Real, this-
>resCycleAvgVoltsDataGridViewTextBoxColumn,
        this-
>tappingWeightScheduleDataGridViewTextBoxColumn, this-
>Meas_Add_Fluoride_Actual, this->Nois_relnoise_FFT});
        this->total_dgv->DataMember = L"comparativa1";
        this->total_dgv->DataSource = this-
>comparativa1BindingSource;
        this->total_dgv->Location = System::Drawing::Point(12,
821);

        this->total_dgv->Name = L"total_dgv";
        this->total_dgv->Size = System::Drawing::Size(1039, 150);
        this->total_dgv->TabIndex = 5;
        this->total_dgv->Visible = false;
        //
        // changebackall_but
        //
        this->changebackall_but->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->changebackall_but->Location =
System::Drawing::Point(60, 287);
        this->changebackall_but->Name = L"changebackall_but";
        this->changebackall_but->Size = System::Drawing::Size(40,
23);

        this->changebackall_but->TabIndex = 6;
        this->changebackall_but->Text = L"<<";
        this->changebackall_but->UseVisualStyleBackColor = true;
        this->changebackall_but->Click += gcnew
System::EventHandler(this, &Form4::button1_Click);
        //

```

```

        // changeall_but
        //
        this->changeall_but->Font = (gcnw
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->changeall_but->Location =
System::Drawing::Point(60, 258);
        this->changeall_but->Name = L"changeall_but";
        this->changeall_but->Size = System::Drawing::Size(40,
23);

        this->changeall_but->TabIndex = 7;
        this->changeall_but->Text = L">>";
        this->changeall_but->UseVisualStyleBackColor = true;
        this->changeall_but->Click += gcnw
System::EventHandler(this, &Form4::button2_Click);
        //
        // changeall2_but
        //
        this->changeall2_but->Font = (gcnw
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->changeall2_but->Location =
System::Drawing::Point(1171, 254);
        this->changeall2_but->Name = L"changeall2_but";
        this->changeall2_but->Size = System::Drawing::Size(40,
23);

        this->changeall2_but->TabIndex = 13;
        this->changeall2_but->Text = L">>";
        this->changeall2_but->UseVisualStyleBackColor = true;
        this->changeall2_but->Click += gcnw
System::EventHandler(this, &Form4::changeall2_but_Click);
        //
        // changebackall2_but
        //
        this->changebackall2_but->Font = (gcnw
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->changebackall2_but->Location =
System::Drawing::Point(1171, 283);
        this->changebackall2_but->Name = L"changebackall2_but";
        this->changebackall2_but->Size =
System::Drawing::Size(40, 23);
        this->changebackall2_but->TabIndex = 12;
        this->changebackall2_but->Text = L"<<";
        this->changebackall2_but->UseVisualStyleBackColor = true;
        this->changebackall2_but->Click += gcnw
System::EventHandler(this, &Form4::changebackall2_but_Click);
        //
        // cubaspost2_lb
        //
        this->cubaspost2_lb->FormattingEnabled = true;
        this->cubaspost2_lb->Location =
System::Drawing::Point(1217, 160);
        this->cubaspost2_lb->Name = L"cubaspost2_lb";

```

```

        this->cubaspost2_lb->Size    =    System::Drawing::Size(41,
810);

        this->cubaspost2_lb->Sorted = true;
        this->cubaspost2_lb->TabIndex = 11;
        this->cubaspost2_lb->DoubleClick    +=                gcnnew
System::EventHandler(this, &Form4::cubaspost2_lb_DoubleClick);
        //
        // changeback2_but
        //
        this->changeback2_but->Font                =                (gcnnew
System::Drawing::Font(L"Microsoft                Sans                Serif",                8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(0)));
        this->changeback2_but->Location                =
System::Drawing::Point(1171, 225);
        this->changeback2_but->Name = L"changeback2_but";
        this->changeback2_but->Size    =    System::Drawing::Size(40,
23);

        this->changeback2_but->TabIndex = 10;
        this->changeback2_but->Text = L"<";
        this->changeback2_but->UseVisualStyleBackColor = true;
        this->changeback2_but->Click    +=                gcnnew
System::EventHandler(this, &Form4::changeback2_but_Click);
        //
        // change2_but
        //
        this->change2_but->Font                =                (gcnnew
System::Drawing::Font(L"Microsoft                Sans                Serif",                8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
                static_cast<System::Byte>(0)));
        this->change2_but->Location                =
System::Drawing::Point(1171, 196);
        this->change2_but->Name = L"change2_but";
        this->change2_but->Size = System::Drawing::Size(40, 23);
        this->change2_but->TabIndex = 9;
        this->change2_but->Text = L">";
        this->change2_but->UseVisualStyleBackColor = true;
        this->change2_but->Click    +=                gcnnew
System::EventHandler(this, &Form4::change2_but_Click);
        //
        // cubaspre2_lb
        //
        this->cubaspre2_lb->FormattingEnabled = true;
        this->cubaspre2_lb->Location                =
System::Drawing::Point(1124, 160);
        this->cubaspre2_lb->Name = L"cubaspre2_lb";
        this->cubaspre2_lb->Size    =    System::Drawing::Size(41,
810);

        this->cubaspre2_lb->Sorted = true;
        this->cubaspre2_lb->TabIndex = 8;
        this->cubaspre2_lb->DoubleClick    +=                gcnnew
System::EventHandler(this, &Form4::cubaspre2_lb_DoubleClick);
        //
        // graficocentral_zg
        //

```

```

        this->graficocentral_zg->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Regular,
        System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
        this->graficocentral_zg->Location =
System::Drawing::Point(165, 137);
        this->graficocentral_zg->Name = L"graficocentral_zg";
        this->graficocentral_zg->ScrollGrace = 0;
        this->graficocentral_zg->ScrollMaxX = 0;
        this->graficocentral_zg->ScrollMaxY = 0;
        this->graficocentral_zg->ScrollMaxY2 = 0;
        this->graficocentral_zg->ScrollMinX = 0;
        this->graficocentral_zg->ScrollMinY = 0;
        this->graficocentral_zg->ScrollMinY2 = 0;
        this->graficocentral_zg->Size =
System::Drawing::Size(941, 834);
        this->graficocentral_zg->TabIndex = 14;
        //
        // titulo_lab
        //
        this->titulo_lab->AutoSize = true;
        this->titulo_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 16,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->titulo_lab->Location = System::Drawing::Point(488,
22);
        this->titulo_lab->Name = L"titulo_lab";
        this->titulo_lab->Size = System::Drawing::Size(292, 26);
        this->titulo_lab->TabIndex = 15;
        this->titulo_lab->Text = L"Análisis Versátil de Cubas";
        //
        // media_cb
        //
        this->media_cb->AutoSize = true;
        this->media_cb->Checked = true;
        this->media_cb->CheckState =
System::Windows::Forms::CheckState::Checked;
        this->media_cb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 9,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->media_cb->Location = System::Drawing::Point(622,
85);
        this->media_cb->Name = L"media_cb";
        this->media_cb->Size = System::Drawing::Size(126, 19);
        this->media_cb->TabIndex = 18;
        this->media_cb->Text = L"Linea de media";
        this->media_cb->UseVisualStyleBackColor = true;
        this->media_cb->Click += gcnew System::EventHandler(this,
&Form4::media_cb_Click);
        //
        // cubas_cb
        //
        this->cubas_cb->AutoSize = true;
        this->cubas_cb->Checked = true;

```

```

        this->cubas_cb->CheckState =
System::Windows::Forms::CheckState::Checked;
        this->cubas_cb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 9,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->cubas_cb->Location = System::Drawing::Point(622,
112);

        this->cubas_cb->Name = L"cubas_cb";
        this->cubas_cb->Size = System::Drawing::Size(131, 19);
        this->cubas_cb->TabIndex = 19;
        this->cubas_cb->Text = L"Lineas de cubas";
        this->cubas_cb->UseVisualStyleBackColor = true;
        this->cubas_cb->Click += gcnew System::EventHandler(this,
&Form4::cubas_cb_Click);
        //
        // calcular_but
        //
        this->calcular_but->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->calcular_but->Location =
System::Drawing::Point(759, 86);
        this->calcular_but->Name = L"calcular_but";
        this->calcular_but->Size = System::Drawing::Size(90, 46);
        this->calcular_but->TabIndex = 20;
        this->calcular_but->Text = L"Actualizar";
        this->calcular_but->UseVisualStyleBackColor = true;
        this->calcular_but->Click += gcnew
System::EventHandler(this, &Form4::calcular_but_Click);
        //
        // notupdated_lab
        //
        this->notupdated_lab->AutoSize = true;
        this->notupdated_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 13,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->notupdated_lab->ForeColor =
System::Drawing::Color::Red;
        this->notupdated_lab->Location =
System::Drawing::Point(861, 87);
        this->notupdated_lab->Name = L"notupdated_lab";
        this->notupdated_lab->Size = System::Drawing::Size(146,
44);

        this->notupdated_lab->TabIndex = 22;
        this->notupdated_lab->Text = L"          Gráfica \r\nNO
actualizada";
        //
        // ok_lab
        //
        this->ok_lab->AutoSize = true;
        this->ok_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 13,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));

```

```

        this->ok_lab->ForeColor =
System::Drawing::Color::ForestGreen;
        this->ok_lab->Location = System::Drawing::Point(910, 98);
        this->ok_lab->Name = L"ok_lab";
        this->ok_lab->Size = System::Drawing::Size(38, 22);
        this->ok_lab->TabIndex = 23;
        this->ok_lab->Text = L"OK";
        this->ok_lab->Visible = false;
        //
        // from_dtp
        //
        this->from_dtp->Location = System::Drawing::Point(278,
85);

        this->from_dtp->Name = L"from_dtp";
        this->from_dtp->Size = System::Drawing::Size(200, 20);
        this->from_dtp->TabIndex = 24;
        this->from_dtp->CloseUp += gcnew
System::EventHandler(this, &Form4::from_dtp_CloseUp);
        //
        // to_dtp
        //
        this->to_dtp->Location = System::Drawing::Point(278,
112);

        this->to_dtp->Name = L"to_dtp";
        this->to_dtp->Size = System::Drawing::Size(200, 20);
        this->to_dtp->TabIndex = 25;
        this->to_dtp->CloseUp += gcnew System::EventHandler(this,
&Form4::to_dtp_CloseUp);
        //
        // variable_cb
        //
        this->variable_cb->FormattingEnabled = true;
        this->variable_cb->Items->AddRange(gcnew cli::array<
System::Object^ >(21) {L"AlF3", L"Fe", L"Ca", L"Si", L"Cu", L"Li", L"Mn",
        L"Temperatura", L"SuperHeat", L"SPPN", L"AE Num",
L"AE Time", L"Fed", L"Bath target", L"Bath level", L"Metal Target", L"Metal
DAS",
        L"Voltaje", L"Colada Real", L"AlF3 Add", L"FFT"});
        this->variable_cb->Location = System::Drawing::Point(165,
110);

        this->variable_cb->Name = L"variable_cb";
        this->variable_cb->Size = System::Drawing::Size(93, 21);
        this->variable_cb->TabIndex = 26;
        this->variable_cb->SelectedIndexChanged += gcnew
System::EventHandler(this, &Form4::variable_cb_SelectedIndexChanged);
        //
        // variable_lab
        //
        this->variable_lab->AutoSize = true;
        this->variable_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->variable_lab->Location =
System::Drawing::Point(165, 91);
        this->variable_lab->Name = L"variable_lab";
        this->variable_lab->Size = System::Drawing::Size(57, 13);

```



```

this->variable_lab->TabIndex = 27;
this->variable_lab->Text = L"Variable:";
//
// diarios_rb
//
this->diarios_rb->AutoSize = true;
this->diarios_rb->Checked = true;
this->diarios_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->diarios_rb->Location = System::Drawing::Point(493,
89);

this->diarios_rb->Name = L"diarios_rb";
this->diarios_rb->Size = System::Drawing::Size(99, 17);
this->diarios_rb->TabIndex = 28;
this->diarios_rb->TabStop = true;
this->diarios_rb->Text = L"Datos diarios";
this->diarios_rb->UseVisualStyleBackColor = true;
this->diarios_rb->Click += gcnew
System::EventHandler(this, &Form4::diarios_rb_Click);
//
// semanales_rb
//
this->semanales_rb->AutoSize = true;
this->semanales_rb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->semanales_rb->Location =
System::Drawing::Point(493, 112);
this->semanales_rb->Name = L"semanales_rb";
this->semanales_rb->Size = System::Drawing::Size(121,
17);

this->semanales_rb->TabIndex = 29;
this->semanales_rb->TabStop = true;
this->semanales_rb->Text = L"Datos semanales";
this->semanales_rb->UseVisualStyleBackColor = true;
this->semanales_rb->Click += gcnew
System::EventHandler(this, &Form4::semanales_rb_Click);
//
// flag
//
this->flag->Location = System::Drawing::Point(106, 137);
this->flag->Name = L"flag";
this->flag->Size = System::Drawing::Size(41, 20);
this->flag->TabIndex = 30;
//
// selected_lab
//
this->selected_lab->AutoSize = true;
this->selected_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
static_cast<System::Byte>(0)));
this->selected_lab->Location = System::Drawing::Point(18,
140);

```

```

        this->selected_lab->Name = L"selected_lab";
        this->selected_lab->Size = System::Drawing::Size(82, 13);
        this->selected_lab->TabIndex = 31;
        this->selected_lab->Text = L"N° de Cubas:";
        //
        // selected2_lab
        //
        this->selected2_lab->AutoSize = true;
        this->selected2_lab->Font = gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0));
        this->selected2_lab->Location =
System::Drawing::Point(1129, 137);
        this->selected2_lab->Name = L"selected2_lab";
        this->selected2_lab->Size = System::Drawing::Size(82,
13);

        this->selected2_lab->TabIndex = 32;
        this->selected2_lab->Text = L"N° de Cubas:";
        //
        // flag2
        //
        this->flag2->Location = System::Drawing::Point(1217,
133);

        this->flag2->Name = L"flag2";
        this->flag2->Size = System::Drawing::Size(41, 20);
        this->flag2->TabIndex = 33;
        //
        // printchart_but
        //
        this->printchart_but->Location =
System::Drawing::Point(129, 22);
        this->printchart_but->Name = L"printchart_but";
        this->printchart_but->Size = System::Drawing::Size(93,
23);

        this->printchart_but->TabIndex = 34;
        this->printchart_but->Text = L"Guardar Gráfica";
        this->printchart_but->UseVisualStyleBackColor = true;
        this->printchart_but->Click += gcnew
System::EventHandler(this, &Form4::printchart_but_Click);
        //
        // printscreen_but
        //
        this->printscreen_but->Location =
System::Drawing::Point(12, 22);
        this->printscreen_but->Name = L"printscreen_but";
        this->printscreen_but->Size = System::Drawing::Size(93,
23);

        this->printscreen_but->TabIndex = 35;
        this->printscreen_but->Text = L"Guardar Captura";
        this->printscreen_but->UseVisualStyleBackColor = true;
        this->printscreen_but->Click += gcnew
System::EventHandler(this, &Form4::printscreen_but_Click);
        //
        // variable2_lab
        //
        this->variable2_lab->AutoSize = true;

```

```

        this->variable2_lab->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->variable2_lab->Location =
System::Drawing::Point(1013, 91);
        this->variable2_lab->Name = L"variable2_lab";
        this->variable2_lab->Size = System::Drawing::Size(57,
13);

        this->variable2_lab->TabIndex = 37;
        this->variable2_lab->Text = L"Variable:";
        //
        // variable2_cb
        //
        this->variable2_cb->FormattingEnabled = true;
        this->variable2_cb->Items->AddRange(gcnew cli::array<
System::Object^ >(21) {L"AlF3", L"Fe", L"Ca", L"Si", L"Cu", L"Li",
        L"Mn", L"Temperatura", L"SuperHeat", L"SPPN", L"AE
Num", L"AE Time", L"Fed", L"Bath target", L"Bath level", L"Metal Target",
        L"Metal DAS", L"Voltaje", L"Colada Real", L"AlF3
Add", L"FFT"});
        this->variable2_cb->Location =
System::Drawing::Point(1013, 110);
        this->variable2_cb->Name = L"variable2_cb";
        this->variable2_cb->Size = System::Drawing::Size(93, 21);
        this->variable2_cb->TabIndex = 36;
        //
        // leyenda_ckb
        //
        this->leyenda_ckb->AutoSize = true;
        this->leyenda_ckb->BackColor =
System::Drawing::Color::LightSkyBlue;
        this->leyenda_ckb->Font = (gcnew
System::Drawing::Font(L"Microsoft Sans Serif", 8.25F,
System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,
        static_cast<System::Byte>(0)));
        this->leyenda_ckb->Location = System::Drawing::Point(278,
149);

        this->leyenda_ckb->Name = L"leyenda_ckb";
        this->leyenda_ckb->Size = System::Drawing::Size(74, 17);
        this->leyenda_ckb->TabIndex = 39;
        this->leyenda_ckb->Text = L"Leyenda";
        this->leyenda_ckb->UseVisualStyleBackColor = false;
        this->leyenda_ckb->Click += gcnew
System::EventHandler(this, &Form4::leyenda_ckb_Click);
        //
        // comparativa1BindingSource
        //
        this->comparativa1BindingSource->DataSource = this-
>TablaGeneral;
        this->comparativa1BindingSource->Position = 0;
        //
        // TablaGeneral
        //
        this->TablaGeneral->DataSetName = L"TablaGeneral";
        this->TablaGeneral->SchemaSerializationMode =
System::Data::SchemaSerializationMode::IncludeSchema;

```

```

//
// comparativaTableAdapter
//
this->comparativaTableAdapter->ClearBeforeFill = true;
//
// pot_Day_Hist1TableAdapter1
//
this->pot_Day_Hist1TableAdapter1->ClearBeforeFill = true;
//
// dateTimeDataGridViewTextBoxColumn
//
this->dateTimeDataGridViewTextBoxColumn->DataPropertyName
= L"Date_Time";
this->dateTimeDataGridViewTextBoxColumn->HeaderText
= L"Date_Time";
this->dateTimeDataGridViewTextBoxColumn->Name
= L"dateTimeDataGridViewTextBoxColumn";
//
// potDataGridViewTextBoxColumn
//
this->potDataGridViewTextBoxColumn->DataPropertyName
= L"Pot";
this->potDataGridViewTextBoxColumn->HeaderText = L"Pot";
this->potDataGridViewTextBoxColumn->Name
= L"potDataGridViewTextBoxColumn";
//
// ageDataGridViewTextBoxColumn
//
this->ageDataGridViewTextBoxColumn->DataPropertyName
= L"Age";
this->ageDataGridViewTextBoxColumn->HeaderText = L"Age";
this->ageDataGridViewTextBoxColumn->Name
= L"ageDataGridViewTextBoxColumn";
//
// mEASSTPBATHXSALF3DataGridViewTextBoxColumn
//
this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn-
>DataPropertyName = L"MEAS_STP_BATH_XSALF3";
this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn-
>HeaderText = L"MEAS_STP_BATH_XSALF3";
this->mEASSTPBATHXSALF3DataGridViewTextBoxColumn->Name
= L"mEASSTPBATHXSALF3DataGridViewTextBoxColumn";
//
// measMetalIronFeDataGridViewTextBoxColumn
//
this->measMetalIronFeDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Iron_Fe";
this->measMetalIronFeDataGridViewTextBoxColumn-
>HeaderText = L"Meas_Metal_Iron_Fe";
this->measMetalIronFeDataGridViewTextBoxColumn->Name
= L"measMetalIronFeDataGridViewTextBoxColumn";
//
// Meas_Bath_Calcium_Fluoride
//
this->Meas_Bath_Calcium_Fluoride->DataPropertyName
= L"Meas_Bath_Calcium_Fluoride";

```

```

        this->Meas_Bath_Calcium_Fluoride->HeaderText =
L"Meas_Bath_Calcium_Fluoride";
        this->Meas_Bath_Calcium_Fluoride->Name =
L"Meas_Bath_Calcium_Fluoride";
        //
        // Meas_Metal_Silicon_Si
        //
        this->Meas_Metal_Silicon_Si->DataPropertyName =
L"Meas_Metal_Silicon_Si";
        this->Meas_Metal_Silicon_Si->HeaderText =
L"Meas_Metal_Silicon_Si";
        this->Meas_Metal_Silicon_Si->Name =
L"Meas_Metal_Silicon_Si";
        //
        // Meas_Metal_Copper_Cu
        //
        this->Meas_Metal_Copper_Cu->DataPropertyName =
L"Meas_Metal_Copper_Cu";
        this->Meas_Metal_Copper_Cu->HeaderText =
L"Meas_Metal_Copper_Cu";
        this->Meas_Metal_Copper_Cu->Name =
L"Meas_Metal_Copper_Cu";
        //
        // measMetalLithiumLiDataGridViewTextBoxColumn
        //
        this->measMetalLithiumLiDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Metal_Lithium_Li";
        this->measMetalLithiumLiDataGridViewTextBoxColumn-
>HeaderText = L"Meas_Metal_Lithium_Li";
        this->measMetalLithiumLiDataGridViewTextBoxColumn->Name =
L"measMetalLithiumLiDataGridViewTextBoxColumn";
        //
        // Meas_Metal_Manganese_Mn
        //
        this->Meas_Metal_Manganese_Mn->DataPropertyName =
L"Meas_Metal_Manganese_Mn";
        this->Meas_Metal_Manganese_Mn->HeaderText =
L"Meas_Metal_Manganese_Mn";
        this->Meas_Metal_Manganese_Mn->Name =
L"Meas_Metal_Manganese_Mn";
        //
        // measBathTemperatureControlDataGridViewTextBoxColumn
        //
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->DataPropertyName =
L"Meas_Bath_Temperature_Control";
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->HeaderText =
L"Meas_Bath_Temperature_Control";
        this-
>measBathTemperatureControlDataGridViewTextBoxColumn->Name =
L"measBathTemperatureControlDataGridViewTextBoxColumn";
        //
        // MEAS_STP_BATH_SUPERHEAT
        //
        this->MEAS_STP_BATH_SUPERHEAT->DataPropertyName =
L"MEAS_STP_BATH_SUPERHEAT";

```

```

        this->MEAS_STP_BATH_SUPERHEAT->HeaderText =
L"MEAS_STP_BATH_SUPERHEAT";
        this->MEAS_STP_BATH_SUPERHEAT->Name =
L"MEAS_STP_BATH_SUPERHEAT";
        //
        // noisSPPNVoltsDataGridViewTextBoxColumn
        //
        this->noisSPPNVoltsDataGridViewTextBoxColumn-
>DataPropertyName = L"Nois_SPPN_Volts";
        this->noisSPPNVoltsDataGridViewTextBoxColumn->HeaderText
= L"Nois_SPPN_Volts";
        this->noisSPPNVoltsDataGridViewTextBoxColumn->Name =
L"noisSPPNVoltsDataGridViewTextBoxColumn";
        //
        // AE_Num_Total
        //
        this->AE_Num_Total->DataPropertyName = L"AE_Num_Total";
        this->AE_Num_Total->HeaderText = L"AE_Num_Total";
        this->AE_Num_Total->Name = L"AE_Num_Total";
        //
        // aETimeAbove8VoltsDataGridViewTextBoxColumn
        //
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn-
>DataPropertyName = L"AE_Time_Above_8_Volts";
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn-
>HeaderText = L"AE_Time_Above_8_Volts";
        this->aETimeAbove8VoltsDataGridViewTextBoxColumn->Name =
L"aETimeAbove8VoltsDataGridViewTextBoxColumn";
        //
        // fedFailuresDataGridViewTextBoxColumn
        //
        this->fedFailuresDataGridViewTextBoxColumn-
>DataPropertyName = L"Fed_Failures";
        this->fedFailuresDataGridViewTextBoxColumn->HeaderText =
L"Fed_Failures";
        this->fedFailuresDataGridViewTextBoxColumn->Name =
L"fedFailuresDataGridViewTextBoxColumn";
        this->fedFailuresDataGridViewTextBoxColumn->ReadOnly =
true;
        //
        // Meas_Bath_Level_Target
        //
        this->Meas_Bath_Level_Target->DataPropertyName =
L"Meas_Bath_Level_Target";
        this->Meas_Bath_Level_Target->HeaderText =
L"Meas_Bath_Level_Target";
        this->Meas_Bath_Level_Target->Name =
L"Meas_Bath_Level_Target";
        //
        // measBathLevelDataGridViewTextBoxColumn
        //
        this->measBathLevelDataGridViewTextBoxColumn-
>DataPropertyName = L"Meas_Bath_Level";
        this->measBathLevelDataGridViewTextBoxColumn->HeaderText
= L"Meas_Bath_Level";
        this->measBathLevelDataGridViewTextBoxColumn->Name =
L"measBathLevelDataGridViewTextBoxColumn";

```

```

//
// Meas_Metal_Pad_Target
//
this->Meas_Metal_Pad_Target->DataPropertyName =
L"Meas_Metal_Pad_Target";
this->Meas_Metal_Pad_Target->HeaderText =
L"Meas_Metal_Pad_Target";
this->Meas_Metal_Pad_Target->Name =
L"Meas_Metal_Pad_Target";
//
// Meas_Metal_DAS_Real
//
this->Meas_Metal_DAS_Real->DataPropertyName =
L"Meas_Metal_DAS_Real";
this->Meas_Metal_DAS_Real->HeaderText =
L"Meas_Metal_DAS_Real";
this->Meas_Metal_DAS_Real->Name = L"Meas_Metal_DAS_Real";
//
// resCycleAvgVoltsDataGridViewTextBoxColumn
//
this->resCycleAvgVoltsDataGridViewTextBoxColumn-
>DataPropertyName = L"Res_Cycle_Avg_Volts";
this->resCycleAvgVoltsDataGridViewTextBoxColumn-
>HeaderText = L"Res_Cycle_Avg_Volts";
this->resCycleAvgVoltsDataGridViewTextBoxColumn->Name =
L"resCycleAvgVoltsDataGridViewTextBoxColumn";
//
// tappingWeightScheduleDataGridViewTextBoxColumn
//
this->tappingWeightScheduleDataGridViewTextBoxColumn-
>DataPropertyName = L"Tapping_Weight_Schedule";
this->tappingWeightScheduleDataGridViewTextBoxColumn-
>HeaderText = L"Tapping_Weight_Schedule";
this->tappingWeightScheduleDataGridViewTextBoxColumn-
>Name = L"tappingWeightScheduleDataGridViewTextBoxColumn";
//
// Meas_Add_Fluoride_Actual
//
this->Meas_Add_Fluoride_Actual->DataPropertyName =
L"Meas_Add_Fluoride_Actual";
this->Meas_Add_Fluoride_Actual->HeaderText =
L"Meas_Add_Fluoride_Actual";
this->Meas_Add_Fluoride_Actual->Name =
L"Meas_Add_Fluoride_Actual";
//
// Nois_relnoise_FFT
//
this->Nois_relnoise_FFT->DataPropertyName =
L"Nois_relnoise_FFT";
this->Nois_relnoise_FFT->HeaderText =
L"Nois_relnoise_FFT";
this->Nois_relnoise_FFT->Name = L"Nois_relnoise_FFT";
//
// Form4
//
this->AutoScaleDimensions = System::Drawing::SizeF(6,
13);

```

```

        this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
        this->AutoScroll = true;
        this->ClientSize = System::Drawing::Size(1028, 753);
        this->Controls->Add(this->leyenda_ckb);
        this->Controls->Add(this->variable2_lab);
        this->Controls->Add(this->variable2_cb);
        this->Controls->Add(this->printscreens_but);
        this->Controls->Add(this->printchart_but);
        this->Controls->Add(this->flag2);
        this->Controls->Add(this->selected2_lab);
        this->Controls->Add(this->selected_lab);
        this->Controls->Add(this->flag);
        this->Controls->Add(this->semanales_rb);
        this->Controls->Add(this->diarios_rb);
        this->Controls->Add(this->variable_lab);
        this->Controls->Add(this->variable_cb);
        this->Controls->Add(this->to_dtp);
        this->Controls->Add(this->from_dtp);
        this->Controls->Add(this->ok_lab);
        this->Controls->Add(this->notupdated_lab);
        this->Controls->Add(this->calcular_but);
        this->Controls->Add(this->cubas_cb);
        this->Controls->Add(this->media_cb);
        this->Controls->Add(this->titulo_lab);
        this->Controls->Add(this->graficocentral_zg);
        this->Controls->Add(this->changeall2_but);
        this->Controls->Add(this->changebackall2_but);
        this->Controls->Add(this->cubaspost2_lb);
        this->Controls->Add(this->changeback2_but);
        this->Controls->Add(this->change2_but);
        this->Controls->Add(this->cubaspre2_lb);
        this->Controls->Add(this->changeall_but);
        this->Controls->Add(this->changebackall_but);
        this->Controls->Add(this->total_dgv);
        this->Controls->Add(this->cubaspost_lb);
        this->Controls->Add(this->changeback_but);
        this->Controls->Add(this->change_but);
        this->Controls->Add(this->cubaspre_lb);
        this->Name = L"Form4";
        this->Text = L"Análisis versátil de cubas";
        this->WindowState =
System::Windows::Forms::FormWindowState::Maximized;
        this->Load += gcnew System::EventHandler(this,
&Form4::Form4_Load);
        this->FormClosing += gcnew
System::Windows::Forms::FormClosingEventHandler(this,
&Form4::Form4_FormClosing);

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>total_dgv) )->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>comparativa1BindingSource) )->EndInit();

        (cli::safe_cast<System::ComponentModel::ISupportInitialize^ >(this-
>TablaGeneral) )->EndInit();

```



```

        this->ResumeLayout(false);
        this->PerformLayout();

    }
    #pragma endregion
    private: System::Void Form4_FormClosing(System::Object^ sender,
System::Windows::Forms::FormClosingEventArgs^ e) {
        e->Cancel = true;
        this->Visible = false;
    }

    private: System::Void Form4_Load(System::Object^ sender,
System::EventArgs^ e) {
        this->from_dtp->Value=this->to_dtp->Value.AddDays(-15);
//Inicializo la fecha inicial a 15 días antes de la actual
        this->variable_cb->SelectedIndex=0; //Inicializo la cb a
la primera
        this->variable2_cb->SelectedIndex=0; //Inicializo la cb
a la primera
        this->leyenda_cke->Checked=1;
//Formato del zedgraph
        this->graficocentral_zg->GraphPane->Fill = gcnew
Fill(Color::LightSkyBlue);
        this->graficocentral_zg->GraphPane->Title->Text="Gráfico
de Temperatura";
        this->graficocentral_zg->GraphPane->Title->FontSpec-
>Size=10;
        this->graficocentral_zg->GraphPane->Legend->FontSpec-
>Size=8;
        this->graficocentral_zg->GraphPane->XAxis->Title-
>FontSpec->Size=10;
        this->graficocentral_zg->GraphPane->YAxis->Title-
>FontSpec->Size=10;
        this->graficocentral_zg->GraphPane->XAxis->Scale-
>FontSpec->Size=8;
        this->graficocentral_zg->GraphPane->YAxis->Scale-
>FontSpec->Size=8;
        this->graficocentral_zg->GraphPane->Y2Axis->IsVisible=1;
        this->graficocentral_zg->GraphPane->Y2Axis->Scale-
>FontSpec->Size=8;
        this->graficocentral_zg->GraphPane->Y2Axis->Title-
>FontSpec->Size=10;
        this->graficocentral_zg->GraphPane->Y2Axis->MajorTic-
>IsOpposite=0;
        this->graficocentral_zg->GraphPane->Y2Axis->MinorTic-
>IsOpposite=0;
        this->graficocentral_zg->GraphPane->YAxis->MajorTic-
>IsOpposite=0;
        this->graficocentral_zg->GraphPane->YAxis->MinorTic-
>IsOpposite=0;
        this->graficocentral_zg->GraphPane->YAxis->MajorGrid-
>IsVisible=true;
        this->graficocentral_zg->GraphPane->Y2Axis->MajorGrid-
>IsVisible=true;
        this->graficocentral_zg->GraphPane->XAxis->MajorGrid-
>IsVisible=true;

```

```

        this->graficocentral_zg->GraphPane->XAxis->MajorGrid-
>Color=System::Drawing::Color::LightGray;
        //Carga datos desde la BD
        this->comparativaTableAdapter->Fill(this->TablaGeneral-
>comparativa, this->from_dtp->Value.Date, this->to_dtp->Value.Date);
//Carga el dgv interface
        //Relleno la lista de cubas posibles para cubaspre y
cubaspre2
        for (int i=0; i<this->total_dgv->Rows->Count-1;i++)
        {
            if (i>0 && this->total_dgv->Rows[i]->Cells[1]-
>Value->Equals(this->total_dgv->Rows[i+1]->Cells[1]->Value)==0)
            {
                this->cubaspre_lb->Items->Add(this-
>total_dgv->Rows[i]->Cells[1]->Value->ToString());
                this->cubaspre2_lb->Items->Add(this-
>total_dgv->Rows[i]->Cells[1]->Value->ToString());
            }
        }
        luz_verde(); //Gráfica vacía y 0 cubas seleccionadas
    }

    //Botón >
    private: System::Void change_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        add();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

    //Botón <
    private: System::Void changeback_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        remove();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

    //Botón >>
    private: System::Void button2_Click(System::Object^ sender,
System::EventArgs^ e) {
        add_all();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

    //Botón <<
    private: System::Void button1_Click(System::Object^ sender,
System::EventArgs^ e) {
        remove_all();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

    //Botón > 2
    private: System::Void change2_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        add2();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

```

```

    }
    //Botón < 2
    private: System::Void changeback2_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        remove2();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }
    //Botón >> 2
    private: System::Void changeall2_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        add_all2();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }
    //Botón << 2
    private: System::Void changebackall2_but_Click(System::Object^
sender, System::EventArgs^ e) {
        remove_all2();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

    //Agregado y eliminación de elementos de las listbox mediante
dobles clicks
    private: System::Void cubaspre_lb_DoubleClick(System::Object^ sender,
System::EventArgs^ e) {
        add();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }
    private: System::Void cubaspost_lb_DoubleClick(System::Object^
sender, System::EventArgs^ e) {
        remove();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }
    private: System::Void cubaspre2_lb_DoubleClick(System::Object^
sender, System::EventArgs^ e) {
        add2();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }
    private: System::Void cubaspost2_lb_DoubleClick(System::Object^
sender, System::EventArgs^ e) {
        remove2();
        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

    //Botón de actualizado de datos y gráfica
    private: System::Void calcular_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        curvas_dias(); //Funcionalidad
        curvas_dias2();
        luz_verde(); //Cambia aviso visual
    }

```

```

        //Botón de selección de datos diarios
        private: System::Void diarios_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
            this->semanales_rb->Checked=0; //Desactiva los datos
semanales
            luz_roja(); //Avisa de que la gráfica ya no está
actualizada
        }
        //Botón de selección de datos semanales
        private: System::Void semanales_rb_Click(System::Object^ sender,
System::EventArgs^ e) {
            this->diarios_rb->Checked=0; //Desactiva los datos
diarios
            fija_fecha();
            this->comparativaTableAdapter->Fill(this->TablaGeneral-
>comparativa, this->from_dtp->Value.Date, this->to_dtp->Value.Date);
//Carga el dgv interface
            luz_roja(); //Avisa de que la gráfica ya no está
actualizada
        }

        //Cuando se hace click en el checkbox de media
        private: System::Void media_cb_Click(System::Object^ sender,
System::EventArgs^ e) {
            luz_roja();
        }
        //Cuando se hace click en el checkbox de cubas
        private: System::Void cubas_cb_Click(System::Object^ sender,
System::EventArgs^ e) {
            luz_roja();
        }
        //Cuando se cambia la selección de la listbox de variables
        private: System::Void variable_cb_SelectedIndexChanged(System::Object^
sender, System::EventArgs^ e) {
            luz_roja();
        }
        //Cuando el usuario selecciona una fecha "desde"
        private: System::Void from_dtp_CloseUp(System::Object^ sender,
System::EventArgs^ e) {
            if (this->semanales_rb->Checked==1)
                fija_fecha();
            this->comparativaTableAdapter->Fill(this->TablaGeneral-
>comparativa, this->from_dtp->Value.Date, this->to_dtp->Value.Date);
//Carga el dgv interface
            luz_roja(); //Avisa de que la gráfica ya no está
actualizada
        }
        //Cuando el usuario selecciona una fecha "hasta"
        private: System::Void to_dtp_CloseUp(System::Object^ sender,
System::EventArgs^ e) {
            if (this->semanales_rb->Checked==1)
                fija_fecha();
            this->comparativaTableAdapter->Fill(this->TablaGeneral-
>comparativa, this->from_dtp->Value.Date, this->to_dtp->Value.Date);
//Carga el dgv interface
    
```

```

        luz_roja(); //Avisa de que la gráfica ya no está
actualizada
    }

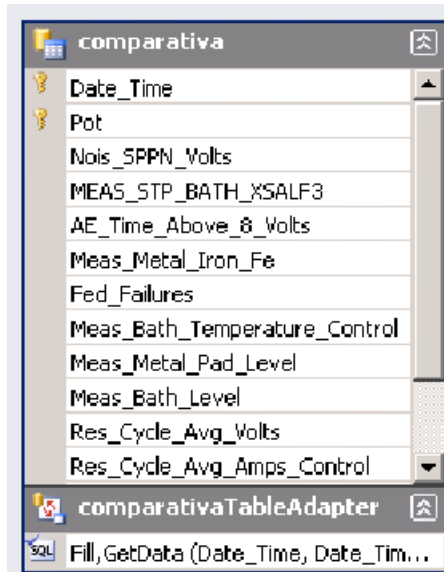
    //Botón de guardar gráfico
    private: System::Void printchart_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        this->graficocentral_zg->SaveAs();
    }
    //Botón de guardar captura
    private: System::Void printscreen_but_Click(System::Object^ sender,
System::EventArgs^ e) {
        SaveFileDialog^ dialog = gcnew SaveFileDialog(); //Se
genera un diálogo de guardado
        dialog->Title = "Guardar captura de pantalla"; //título
del diálogo de guardado
        dialog->ShowDialog(); //Se abre un diálogo de guardado
        if (dialog->FileName != "") //Si se ha escrito un nombre
        {
            Bitmap^ captura; //Declaro el objeto captura para
almacenar el pantallazo
            Rectangle^ rectangulo; //Declaro el objeto
rectángulo para el pantallazo
            captura = gcnew Bitmap(this->Width, this->Height);
            //Inicializo el bitmap al tamaño de la form
            rectangulo = gcnew Rectangle(0, 0, this->Width,
this->Height); //Inicializo el rectángulo del pantallazo
            this->DrawToBitmap(captura, *rectangulo); //Saco
el pantallazo y lo almaceno en captura
            captura->Save(String::Concat(dialog->FileName,
".png"), System::Drawing::Imaging::ImageFormat::Bmp); //Guardo el archivo
con el nombre que se le puso en el diálogo añadiendo .png al final. Además
asigno ese tipo de archivo
        }
    }
    //Activa o desactiva la leyenda
    private: System::Void leyenda_ckb_Click(System::Object^ sender,
System::EventArgs^ e) {
        if (this->leyenda_ckb->Checked==1)
            this->graficocentral_zg->GraphPane->Legend-
>IsVisible=1;
        else
            this->graficocentral_zg->GraphPane->Legend-
>IsVisible=0;
        this->graficocentral_zg->Refresh();
    }
};
}

```

## 2.5 Querys

El código autogenerated recogido en el archivo TablaGeneral.xds no resulta de fácil comprensión, de modo que para una mayor legibilidad y entendimiento, se incluyen en su lugar imágenes de las tablas de datos, así como el código SQL de sus querys.

### 2.5.1 Comparativa

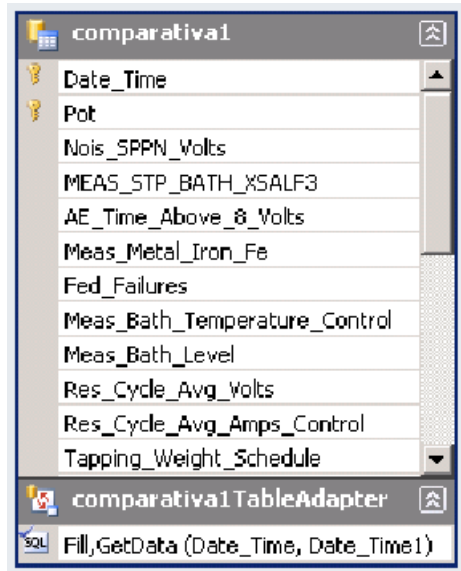


```

SELECT  Pot_Day_Hist.Date_Time, Pot_Day_Hist.Pot, Pot_Day_Hist.Nois_SPPN_Volts * 1000
AS Nois_SPPN_Volts, Pot_Day_Hist.MEAS_STP_BATH_XSALF3,
        Pot_Day_Hist.AE_Time_Above_8_Volts / 60 AS AE_Time_Above_8_Volts,
Pot_Day_Hist.Meas_Metal_Iron_Fe,
        L_Pot_Day_Hist.Fed_Failure_Int_Day + L_Pot_Day_Hist.Fed_Failure_Ext_Day AS
Fed_Failures,
        L_Pot_Day_Hist.Meas_Bath_Lithium_Fluoride AS Meas_Metal_Lithium_Li,
Pot_Day_Hist.Meas_Bath_Temperature_Control,
        Pot_Day_Hist.Meas_Metal_Pad_Level, Pot_Day_Hist.Meas_Bath_Level,
Pot_Day_Hist.Res_Cycle_Avg_Volts,
        Pot_Day_Hist.Res_Cycle_Avg_Amps_Control,
L_Pot_Day_Hist.Tapping_Weight_Schedule, Pot_Day_Hist.Age
FROM    Pot_Day_Hist INNER JOIN
        L_Pot_Day_Hist ON Pot_Day_Hist.Date_Time = L_Pot_Day_Hist.Date_Time AND
Pot_Day_Hist.Pot = L_Pot_Day_Hist.Pot
WHERE   (Pot_Day_Hist.Date_Time >= ?) AND (Pot_Day_Hist.Date_Time <= ?) AND
(Pot_Day_Hist.Res_Cycle_Avg_Volts > 0)

```

## 2.5.2 Comparativa 1

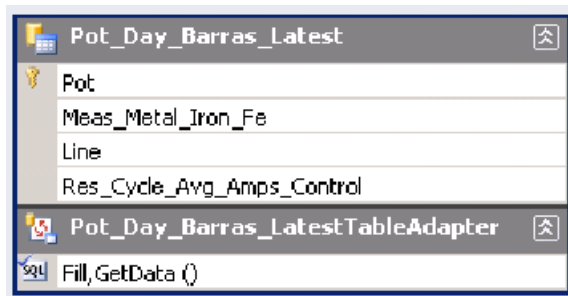


```

SELECT    Pot_Day_Hist.Date_Time, Pot_Day_Hist.Pot, Pot_Day_Hist.Nois_SPPN_Volts * 1000
AS Nois_SPPN_Volts, Pot_Day_Hist.MEAS_STP_BATH_XSALF3,
          Pot_Day_Hist.AE_Num_Total, Pot_Day_Hist.AE_Time_Above_8_Volts / 60 AS
AE_Time_Above_8_Volts,
          Pot_Day_Hist.Meas_Metal_Iron_Fe * 100 AS Meas_Metal_Iron_Fe,
          L_Pot_Day_Hist.Fed_Failure_Int_Day + L_Pot_Day_Hist.Fed_Failure_Ext_Day AS
Fed_Failures,
          L_Pot_Day_Hist.Meas_Bath_Lithium_Fluoride AS Meas_Metal_Lithium_Li,
Pot_Day_Hist.Meas_Bath_Temperature_Control,
          Pot_Day_Hist.Res_Cycle_Avg_Volts, Pot_Day_Hist.Res_Cycle_Avg_Amps_Control,
L_Pot_Day_Hist.Tapping_Weight_Schedule, Pot_Day_Hist.Age,
          Pot_Day_Hist.Meas_Metal_Lithium_Li AS Expr1,
Pot_Day_Hist.Meas_Bath_Calcium_Fluoride,
          Pot_Day_Hist.Meas_Metal_Silicon_Si * 100 AS Meas_Metal_Silicon_Si,
Pot_Day_Hist.Meas_Metal_Copper_Cu * 10000 AS Meas_Metal_Copper_Cu,
          Pot_Day_Hist.Meas_Metal_Manganese_Mn * 10000 AS
Meas_Metal_Manganese_Mn, Pot_Day_Hist.Meas_Metal_Pad_Target,
          L_Pot_Day_Hist.Meas_Metal_DAS_Real, Pot_Day_Hist.Meas_Bath_Level,
Pot_Day_Hist.Meas_Bath_Level_Target,
          Pot_Day_Hist.Meas_Add_Fluoride_Actual,
Pot_Day_Hist.MEAS_STP_BATH_SUPERHEAT, L_Pot_Day_Hist.Nois_relnoise_FFT
FROM      Pot_Day_Hist INNER JOIN
          L_Pot_Day_Hist ON Pot_Day_Hist.Date_Time = L_Pot_Day_Hist.Date_Time AND
Pot_Day_Hist.Pot = L_Pot_Day_Hist.Pot
WHERE     (Pot_Day_Hist.Date_Time >= ?) AND (Pot_Day_Hist.Date_Time <= ?) AND
(Pot_Day_Hist.Res_Cycle_Avg_Volts > 0)
ORDER BY Pot_Day_Hist.Pot, Pot_Day_Hist.Date_Time

```

### 2.5.3 Pot\_Day\_Barras\_Latest

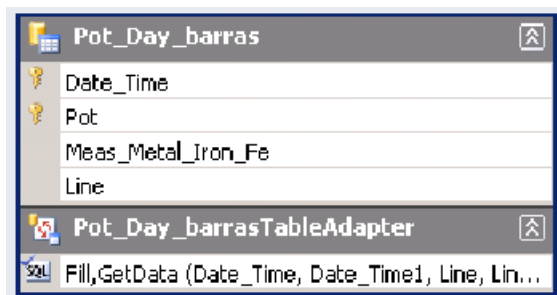


```

SELECT      Pot, Meas_Metal_Iron_Fe * 100 AS Meas_Metal_Iron_Fe, Line,
Res_Cycle_Avg_Amps_Control
FROM        Pot_Day_Latest
WHERE       (Res_Cycle_Avg_Amps_Control > 0)

```

### 2.5.4 Pot\_Day\_Barras



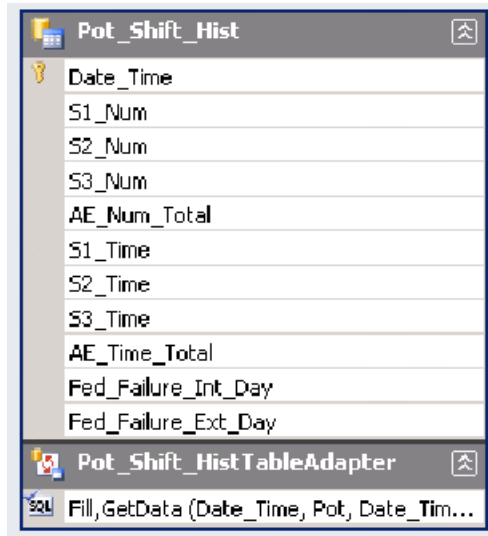
```

SELECT  Date_Time, Pot, Meas_Metal_Iron_Fe * 100 AS Meas_Metal_Iron_Fe, Line
FROM    Pot_Day_Hist
WHERE   (Date_Time > ?) AND (Date_Time <= ?) AND (Meas_Metal_Iron_Fe >= 0) AND (Line =
? OR
        Line = ?)

```



## 2.5.5 Pot\_Shift\_Hist

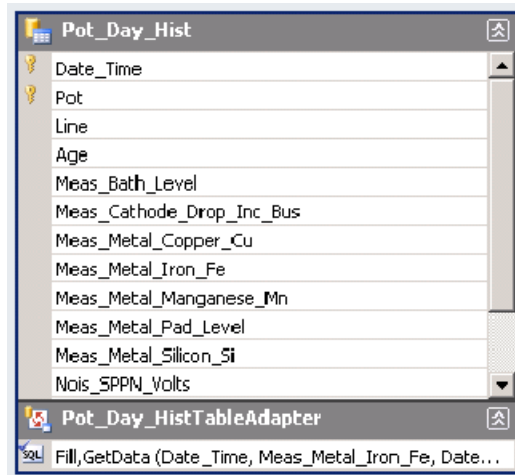


```

SELECT Pot_Shift_Hist.Date_Time, MAX(CASE Shift WHEN 1 THEN AE_Num_Total ELSE NULL
END) AS S1_Num,
      MAX(CASE Shift WHEN 2 THEN AE_Num_Total ELSE NULL END) - MAX(CASE Shift
WHEN 1 THEN AE_Num_Total ELSE NULL END) AS S2_Num,
      MAX(CASE Shift WHEN 3 THEN AE_Num_Total ELSE NULL END) - MAX(CASE Shift
WHEN 2 THEN AE_Num_Total ELSE NULL END) AS S3_Num,
      MAX(CASE Shift WHEN 3 THEN AE_Num_Total ELSE NULL END) AS AE_Num_Total,
CONVERT(VARCHAR, DATEADD(second,
      MAX(CASE Shift WHEN 1 THEN AE_time_above_8_volts ELSE NULL END), 0), 108)
AS S1_Time, CONVERT(VARCHAR, DATEADD(second,
      MAX(CASE Shift WHEN 2 THEN AE_time_above_8_volts ELSE NULL END), 0), 108)
AS S2_Time, CONVERT(VARCHAR, DATEADD(second,
      MAX(CASE Shift WHEN 3 THEN AE_time_above_8_volts ELSE NULL END), 0), 108)
AS S3_Time, CONVERT(VARCHAR, DATEADD(second,
      MAX(CASE Shift WHEN 1 THEN AE_time_above_8_volts ELSE NULL END) +
MAX(CASE Shift WHEN 2 THEN AE_time_above_8_volts ELSE NULL END)
      + MAX(CASE Shift WHEN 3 THEN AE_time_above_8_volts ELSE NULL END), 0), 108)
AS AE_Time_Total, L_Pot_Day_Hist.Fed_Failure_Int_Day,
      L_Pot_Day_Hist.Fed_Failure_Ext_Day
FROM Pot_Shift_Hist INNER JOIN
      L_Pot_Day_Hist ON Pot_Shift_Hist.Date_Time = L_Pot_Day_Hist.Date_Time AND
Pot_Shift_Hist.Pot = L_Pot_Day_Hist.Pot
WHERE (Pot_Shift_Hist.Date_Time >= ?) AND (Pot_Shift_Hist.Pot = ?) AND
(Pot_Shift_Hist.Date_Time <= ?)
GROUP BY Pot_Shift_Hist.Date_Time, L_Pot_Day_Hist.Fed_Failure_Int_Day,
L_Pot_Day_Hist.Fed_Failure_Ext_Day
ORDER BY Pot_Shift_Hist.Date_Time DESC

```

## 2.5.6 Pot\_Day\_Hist

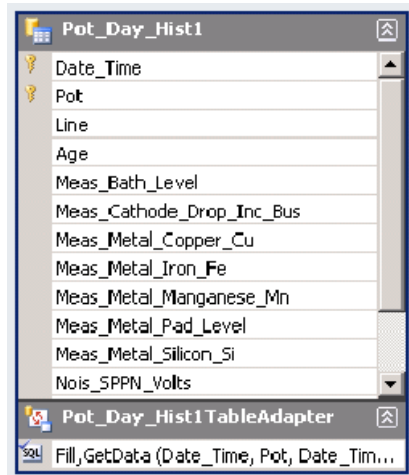


```

SELECT    Pot_Day_Hist.Date_Time, Pot_Day_Hist.Pot, Pot_Day_Hist.Line, Pot_Day_Hist.Age,
Pot_Day_Hist.Meas_Bath_Level,
          Pot_Day_Hist.Meas_Cathode_Drop_Inc_Bus,
Pot_Day_Hist.Meas_Metal_Copper_Cu * 10000 AS Meas_Metal_Copper_Cu,
          Pot_Day_Hist.Meas_Metal_Iron_Fe * 100 AS Meas_Metal_Iron_Fe,
          Pot_Day_Hist.Meas_Metal_Manganese_Mn      *      1000      AS
Meas_Metal_Manganese_Mn,
          Pot_Day_Hist.Meas_Metal_Pad_Target        AS      Meas_Metal_Pad_Level,
Pot_Day_Hist.Meas_Metal_Silicon_Si * 100 AS Meas_Metal_Silicon_Si,
          Pot_Day_Hist.Nois_SPPN_Volts      *      1000      AS      Nois_SPPN_Volts,
Pot_Day_Hist.MEAS_STP_BATH_XSALF3,
          Pot_Day_Hist.Meas_Bath_Temperature_Control,
L_Pot_Day_Hist.Meas_Metal_DAS_Real
FROM      Pot_Day_Hist INNER JOIN
          L_Pot_Day_Hist ON Pot_Day_Hist.Date_Time = L_Pot_Day_Hist.Date_Time AND
Pot_Day_Hist.Pot = L_Pot_Day_Hist.Pot
WHERE     (Pot_Day_Hist.Line >= 2) AND (Pot_Day_Hist.Date_Time = ?) AND
(Pot_Day_Hist.Meas_Metal_Iron_Fe >= ? / 10) OR
          (Pot_Day_Hist.Line >= 2) AND (Pot_Day_Hist.Date_Time = ?) AND
(Pot_Day_Hist.Pot = ?) OR
          (Pot_Day_Hist.Line >= 2) AND (Pot_Day_Hist.Date_Time = ?) AND
(Pot_Day_Hist.Pot = ?) OR
          (Pot_Day_Hist.Line >= 2) AND (Pot_Day_Hist.Date_Time = ?) AND
(Pot_Day_Hist.Pot = ?) OR
          (Pot_Day_Hist.Line >= 2) AND (Pot_Day_Hist.Date_Time = ?) AND
(Pot_Day_Hist.Pot = ?) OR
          (Pot_Day_Hist.Line >= 2) AND (Pot_Day_Hist.Date_Time = ?) AND
(Pot_Day_Hist.Pot = ?)

```

## 2.5.7 Pot\_Day\_Hist1

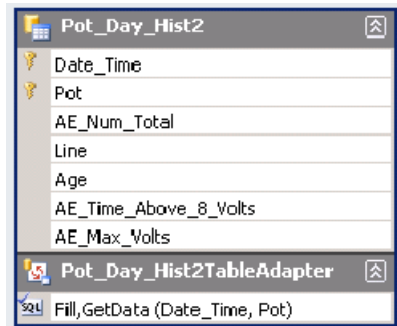


```

SELECT    Pot_Day_Hist.Date_Time, Pot_Day_Hist.Pot, Pot_Day_Hist.Line, Pot_Day_Hist.Age,
Pot_Day_Hist.Meas_Bath_Level,
          Pot_Day_Hist.Meas_Cathode_Drop_Inc_Bus,
Pot_Day_Hist.Meas_Metal_Copper_Cu * 10000 AS Meas_Metal_Copper_Cu,
          Pot_Day_Hist.Meas_Metal_Iron_Fe * 100 AS Meas_Metal_Iron_Fe,
          Pot_Day_Hist.Meas_Metal_Manganese_Mn      *      1000      AS
Meas_Metal_Manganese_Mn,
          Pot_Day_Hist.Meas_Metal_Pad_Target      AS      Meas_Metal_Pad_Level,
Pot_Day_Hist.Meas_Metal_Silicon_Si * 100 AS Meas_Metal_Silicon_Si,
          Pot_Day_Hist.Nois_SPPN_Volts      *      1000      AS      Nois_SPPN_Volts,
Pot_Day_Hist.MEAS_STP_BATH_XSALF3,
          Pot_Day_Hist.Meas_Bath_Temperature_Control,
L_Pot_Day_Hist.Meas_Metal_DAS_Real AS ELDAS
FROM      Pot_Day_Hist INNER JOIN
          L_Pot_Day_Hist ON Pot_Day_Hist.Date_Time = L_Pot_Day_Hist.Date_Time AND
Pot_Day_Hist.Pot = L_Pot_Day_Hist.Pot
WHERE     (Pot_Day_Hist.Line > 1) AND (Pot_Day_Hist.Date_Time >= ?) AND (Pot_Day_Hist.Pot
= ?) AND (Pot_Day_Hist.Date_Time <= ?)
ORDER BY Pot_Day_Hist.Date_Time DESC

```

## 2.5.8 Pot\_Day\_Hist2

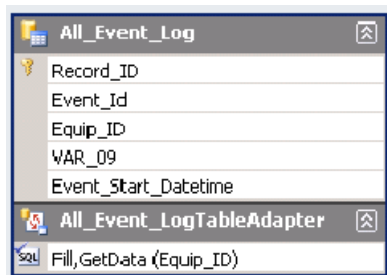


```

SELECT  Pot_Day_Hist.Date_Time, Pot_Day_Hist.Pot, CONVERT(VARCHAR, DATEADD(second,
Pot_Day_Hist.AE_Time_Above_8_Volts, 0), 108)
        AS AE_Time_Above_8_Volts, Pot_Day_Hist.AE_Num_Total, Pot_Day_Hist.Line,
Pot_Day_Hist.Age, L_Pot_Day_Hist.AE_Max_Volts
FROM    Pot_Day_Hist INNER JOIN
        L_Pot_Day_Hist ON Pot_Day_Hist.Date_Time = L_Pot_Day_Hist.Date_Time AND
Pot_Day_Hist.Pot = L_Pot_Day_Hist.Pot
WHERE   (Pot_Day_Hist.Date_Time > ?) AND (Pot_Day_Hist.Line > 1) AND (Pot_Day_Hist.Pot =
?)
ORDER BY Pot_Day_Hist.Date_Time DESC

```

## 2.5.9 All\_Event\_Log

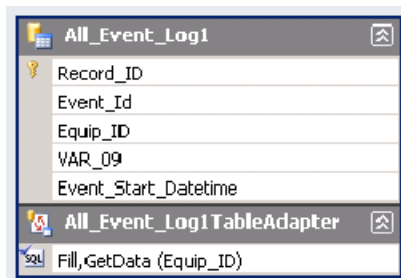


```

SELECT  TOP (1) Record_ID, Event_Id, Equip_ID, VAR_09, Event_Start_Datetime
FROM    (SELECT  TOP (2) Record_ID, Event_Id, Equip_ID, VAR_09, Event_Start_Datetime
        FROM    All_Event_Log
        WHERE   (Event_Id = 1005) AND (Equip_ID = ?)
        ORDER BY Record_ID DESC) AS derivedtbl_1
ORDER BY Record_ID

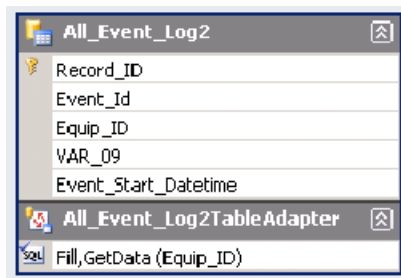
```

### 2.5.10 All\_Event\_Log1



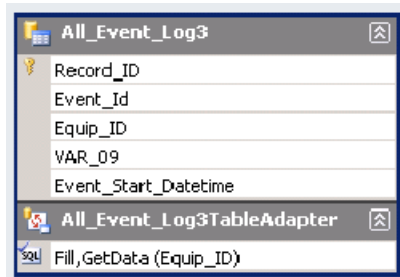
```
SELECT TOP (1) Record_ID, Event_Id, Equip_ID, VAR_09, Event_Start_Datetime
FROM All_Event_Log
WHERE (Event_Id = 1005) AND (Equip_ID = ?)
ORDER BY Record_ID DESC
```

### 2.5.11 All\_Event\_Log2



```
SELECT TOP (1) Record_ID, Event_Id, Equip_ID, VAR_09, Event_Start_Datetime
FROM (SELECT TOP (3) Record_ID, Event_Id, Equip_ID, VAR_09, Event_Start_Datetime
FROM All_Event_Log
WHERE (Event_Id = 1005) AND (Equip_ID = ?)
ORDER BY Record_ID DESC) AS derivedtbl_1
ORDER BY Record_ID
```

### 2.5.12 All\_Event\_Log3

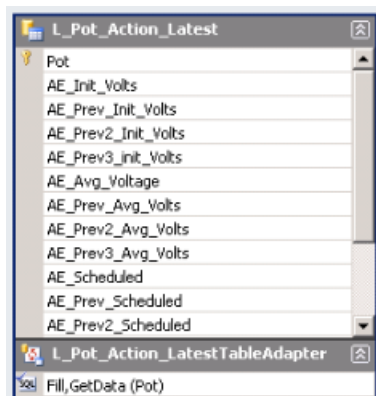


```

SELECT TOP (1) Record_ID, Event_Id, Equip_ID, VAR_09, Event_Start_Datetime
FROM (SELECT TOP (4) Record_ID, Event_Id, Equip_ID, VAR_09, Event_Start_Datetime
      FROM All_Event_Log
      WHERE (Event_Id = 1005) AND (Equip_ID = ?)
      ORDER BY Record_ID DESC) AS derivedtbl_1
ORDER BY Record_ID

```

### 2.5.13 L\_Pot\_Action\_Latest



```

SELECT AE_Init_Volts, AE_Prev_Init_Volts, AE_Prev2_Init_Volts, AE_Prev3_init_Volts,
AE_Avg_Voltage, AE_Prev_Avg_Volts, AE_Prev2_Avg_Volts,
      AE_Prev3_Avg_Volts, CONVERT(varchar(8), DATEADD(second,
AE_Time_Above_8V, 0), 108) AS AE_Time_Above_8V, CONVERT(varchar(8), DATEADD(second,
AE_Prev_Time_Above_8V, 0), 108) AS AE_Prev_Time_Above_8V,
      CONVERT(varchar(8), DATEADD(second, AE_Prev2_Time_Above_8V, 0), 108) AS
AE_Prev2_Time_Above_8V, CONVERT(varchar(8), DATEADD(second, AE_Prev3_Time_Above_8V, 0),
108) AS AE_Prev3_Time_Above_8V, AE_Scheduled, AE_Prev_Scheduled, AE_Prev2_Scheduled,
AE_Prev3_Scheduled, Pot
FROM L_Pot_Action_Latest
WHERE (Pot = ?)

```