



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

CUADRO DE MANDOS PARA UN PROCESO DE DESARROLLO DE SOFTWARE

Autor:

Agustí Pellicer Nácher

Director del Proyecto:

Patricio Letelier Torres

AGRADECIMIENTOS

A Patricio Letelier por su apoyo y a todos los amigos que me han acompañado en toda la carrera.

Dedicado a mi familia y Mirka.

INDICE

AGRADECIMIENTOS	3
1. Introducción.....	7
1.1. Motivación	7
1.2. Objetivos.....	7
1.3. Organización de la memoria	7
2. Cuadro de Mandos Integral	9
2.1. Metodología Kaplan & Norton.....	9
2.2. Beneficios de la implantación de un Cuadro de Mandos Integral	10
2.3. Riesgos de la implantación de un Cuadro de Mandos Integral.....	10
3. Microsoft Report Builder	11
3.1. Introducción.....	11
3.2. Conceptos básicos	11
3.2.1. Interfaz de Microsoft Report Builder 3.0	11
3.2.2. Data Sources	12
3.2.3. Datasets	13
3.2.4. Parámetros	15
3.3. Creación del informe	16
3.3.1. Esqueleto del informe.....	16
4. Estado del arte	26
4.1. Introducción.....	26
4.2. Pivotal Tracker	26
4.3. Scrumpy	27
4.4. Icescrum.....	28
4.5. Scrumdesk.....	30
4.6. ScrumWorks.....	31
4.7. TeamPulse.....	33
5. Tune-Up	36
5.1. ¿Qué es TUNE-UP?.....	36
5.2. TUNE-UP Process Tool	38
5.2.1. Planificador Personal	38

5.2.2. Gestor de Unidades de Trabajo	39
5.2.3. Planificador de Versiones	40
5.2.4. Cuadro de Mandos en TUNE-UP	40
6. Creación del cuadro de mandos	44
6.1. Introducción.....	44
6.2. Porcentaje de WUs por versión	45
6.2.1. Explicación	45
6.2.2. Parte técnica	46
6.3. Número de eventos por versión	46
6.3.1. Explicación	46
6.3.2. Parte técnica	47
6.4. Velocidad de la versión	48
6.4.1. Explicación	48
6.4.2. Parte técnica	49
7. Conclusiones.....	50
Referencias.....	52

1. Introducción

1.1. Motivación

En la actualidad existen diversas metodologías para abordar el desarrollo de software y una inmensa mayoría de ellas se basan en un proceso iterativo hasta conseguir que los requisitos y metas que se habían marcado en el proceso de análisis se cumplan y sean correctos.

La planificación y seguimiento son actividades claves para el éxito de un proyecto. También es importante dentro de un marco de mejora continua realizar frecuentemente evaluaciones de los procesos empleados, evaluando su efectividad y estableciendo acciones de mejora.

Para cualquier iniciativa de mejora es indispensable poder medir el rendimiento de los procesos actuales teniendo así un punto de referencia. Esto se corresponde con el nivel de madurez CMMI (Cuadro de Mandos Integral) asociado a la gestión cuantitativa del proceso.

Para presentar de forma efectiva, concisa y clara la información necesaria para controlar el proceso de desarrollo y mantenimiento de software toma relevancia el concepto de Cuadro de Mandos. Un Cuadro de Mandos permite mostrar de forma resumida indicadores del proceso permitiendo observar tendencias y observar desviaciones con respecto de los umbrales establecidos para dichos indicadores. Así, un cuadro de mandos es una herramienta para medir actividades y aportar una visión estratégica de un conjunto de datos que puedan ser importantes y relevantes en la empresa y en su devenir.

1.2. Objetivos

El objetivo de este proyecto se basa en la creación de un Cuadro de Mandos para el desarrollo de software basado en la metodología TUNE-UP, que sea básico, útil y muestre información que ayude a tomar decisiones a la hora de plantear siguientes versiones o nuevos productos viendo tendencias pasadas.

Haremos un repaso al concepto Cuadro de Mandos, también investigaremos sobre como otras herramientas basadas en diferentes metodologías ágiles, han integrado el concepto de Cuadro de Mandos en su aplicación haciendo una pequeña reseña de cada una. Además ofreceremos un pequeño recorrido por la herramienta, Microsoft Report Builder 3.0, que usaremos para crear dicho Cuadro de Mandos.

1.3. Organización de la memoria

A continuación se presenta la estructura del proyecto:

1. Introducción, explicaremos la motivación y objetivos del proyecto dentro del área Cuadro de Mandos.
2. Cuadro de Mandos Integral, repasaremos el concepto de CMI y veremos su evolución en el tiempo.
3. Guía Microsoft Report Builder 3.0, donde haremos un repaso a su interfaz y a las características más importantes, además haremos un informe ejemplo para ver su funcionamiento.

4. Estado del arte sobre el tema a desarrollar, veremos las distintas soluciones que se puedan encontrar sobre cuadros de mandos para el desarrollo de software en diferentes productos enfocados a metodologías ágiles.
5. Tune-Up, explicaremos los aspectos básicos de su metodología.
6. Parte práctica, la implementación del cuadro de mandos usando todo lo visto anteriormente y explicación de las gráficas creadas.
7. Conclusiones, haremos un repaso a todo lo visto anteriormente y resumiremos el trabajo realizado.

2. Cuadro de Mandos Integral

El cuadro de mandos integral (CMI), también conocido como Balanced Scorecard (BSC) o dashboard, es una herramienta de control empresarial que permite establecer y monitorizar los objetivos de una empresa y sus diferentes áreas o unidades.

También puede ser considerado como una aplicación que ayuda a una organización a establecer sus objetivos estratégicos e iniciativas necesarias para cumplirlos, mostrando en cada instante cuan cerca o lejos se está de cumplirlos. Para ello es necesaria la elaboración de un modelo de negocio que refleje las interrelaciones entre los diferentes componentes de la empresa (mediante un plan estratégico).

A diferencia de otras herramientas de explotación de información, los cuadros de mando están más enfocados al seguimiento de los indicadores (financieros o no financieros) de objetivos estratégicos de la organización que al análisis minucioso de la información. Por ello, es de vital importancia que los indicadores estén bien definidos y que estos tengan un alto grado de relación con el objetivo estratégico a cumplir.

La aportación que ha convertido al CMI en una de las herramientas más significativas de los últimos años es que se cimenta en un modelo de negocio. El éxito de su implantación radica en que el equipo de dirección se involucre y dedique tiempo al desarrollo de su propio modelo de negocio.

Este apartado de la memoria se ha desarrollado consultando las siguiente referencia [1].

2.1. Metodología Kaplan & Norton

A pesar que hay un buen número de metodologías a la hora de elaborar cuadros de mando, la mayoría de éstos siguen la elaborada por Kaplan y Norton durante los años 90, materializada en el libro “The Balanced Scorecard”.

Kaplan y Norton consideraron que los antiguos modelos de gestión empresarial basados únicamente en indicadores financieros estaban completamente obsoletos, ya que no se tenían en cuenta los factores no financieros y la influencia que podían tener sobre los objetivos estratégicos de la organización.

Para la definición de estos objetivos, se tuvo en cuenta que los mismos pueden organizarse en cuatro distintas áreas o perspectivas:

- **Perspectiva Financiera:** serán aquellos objetivos que incorpora la visión de los accionistas, midiendo la creación de valor para la organización. Un indicador ejemplo de los objetivos estratégicos incluidos en esta perspectiva sería el beneficio de la organización.

- **Perspectiva de Cliente:** son los objetivos que responden al posicionamiento de la organización dentro del mercado o el segmento de mercado donde esta se mueve. Un indicador ejemplo de los objetivos estratégicos incluidos en esta perspectiva sería la cuota de mercado.

- **Perspectiva Interna:** objetivos derivados de los procesos internos de la organización que son críticos para el posicionamiento de mercado. Un indicador ejemplo de los objetivos

estratégicos incluidos en esta perspectiva sería la productividad de la planta de producción de una organización.

- **Perspectiva de Aprendizaje y Crecimiento:** teniendo en cuenta que los recursos materiales y las personas son las claves del éxito de cualquier organización, esta perspectiva incluye a todos aquellos objetivos que reflejan el éxito de la organización a la hora de la inversión en recursos materiales y el aprendizaje de los trabajadores. Un indicador ejemplo de los objetivos estratégicos incluidos en esta perspectiva sería el porcentaje de trabajadores que dominan una determinada tecnología.

Pese a que estas son las cuatro perspectivas más genéricas, no son obligatorias, y en determinados tipos de empresas surgirán nuevas perspectivas. En la Figura 2-1 se puede observar cómo interactúan las perspectivas entre sí.



Figura 2-1: Diagrama de interacción de las distintas perspectivas

2.2. Beneficios de la implantación de un Cuadro de Mandos Integral

- La traducción de un modelo de negocio en indicadores clave facilita el consenso en toda la empresa, no sólo en la dirección.
- Da una visión más amplia de cómo las acciones realizadas día a día afectan no solo al corto plazo, sino al largo plazo.
- Una vez que el cuadro de mando integral está en marcha, se pueden comunicar los planes a toda la organización, aunando esfuerzos y remando toda ella en la misma dirección.
- Permite detectar anomalías en el desarrollo del plan estratégico u operativo.

2.3. Riesgos de la implantación de un Cuadro de Mandos Integral

- Un modelo de negocio poco elaborado y sin la colaboración de todos los componentes de la organización es inútil.
- Los indicadores deben ser escogidos con cuidado para no distorsionar el cumplimiento de los objetivos.

3. Microsoft Report Builder

En este capítulo presentaremos la herramienta que usaremos para hacer el Cuadro de Mandos, para ello haremos un repaso a sus características más importantes y haremos un informe de ejemplo básico para ilustrar el funcionamiento de Microsoft Report Builder 3.0.

Se ha utilizado la referencia [2] para la redacción de este capítulo.

3.1. Introducción

Microsoft Report Builder 3.0 es una herramienta para la creación de informes que forma parte de la gama de productos relacionados con Microsoft SQL Server 2008 R2. Esta gama de productos se basa en la explotación de datos de una manera visual.

El flujo de trabajo para diseñar un informe en Microsoft Report Builder es el siguiente: se escoge la fuente de los datos, se crean consultas que nos devuelvan la información deseada, se diseña el informe (se añaden tablas, gráficas, etc.) y por último si hay que procesar esos datos de alguna manera. Cuando se ejecuta, hay un motor de informes que coge toda la información especificada en el informe y la combina con el diseño para generar el informe. Se puede ver el informe después en el mismo programa, el Report Server o integrado en un entorno SharePoint.

Microsoft Report Builder nos ofrece controles que para hacer informes de una manera fácil y potente, estos incluyen: gráficas, tablas, mapas, indicadores, etc. También tenemos a disposición la creación de parámetros del informe para aportar cierta interactividad si se necesita. Los informes se guardan en formato *.rdl (Report Description Language) que al fin y al cabo es un fichero XML.

En resumen, Microsoft Report Builder nos ofrece un punto de partida para la creación de informes en un entorno Microsoft y que se integra con otras herramientas diseñadas por Microsoft, de fácil implantación en la infraestructura de la empresa.

Microsoft Report Builder 3.0 se ofrece de forma gratuita desde la siguiente URL:

<http://www.microsoft.com/downloads/en/details.aspx?FamilyID=d3173a87-7c0d-40cc-a408-3d1a43ae4e33&displaylang=en>

3.2. Conceptos básicos

Antes de la creación del informe ejemplo repasaremos la interfaz del programa y los elementos fundamentales en cualquier informe creado en Microsoft Report Builder.

3.2.1. Interfaz de Microsoft Report Builder 3.0

La interfaz de Microsoft Report Builder es muy parecida a los productos de la suite ofimática Microsoft Office. En la figura 2-1 podemos ver la ventana principal del programa dividida en cuatro grandes áreas.

La primera y más importante al principio es Report Data, aquí es donde se listan y se crean todos los datasources, datasets y parámetros del informe, es el punto de partida. Como bien dice el nombre es donde pondremos todos los datos que usaremos después en el diseño del informe.

La segunda área es la barra de herramientas dividida en tres pestañas, que incluyen las típicas herramientas de formateo para texto, para insertar los elementos de presentación (gráficas, tablas, etc.), visualización del informe y el botón 'Run' que nos permite ejecutar el informe directamente desde la interfaz de Microsoft Report Builder y ver el resultado.

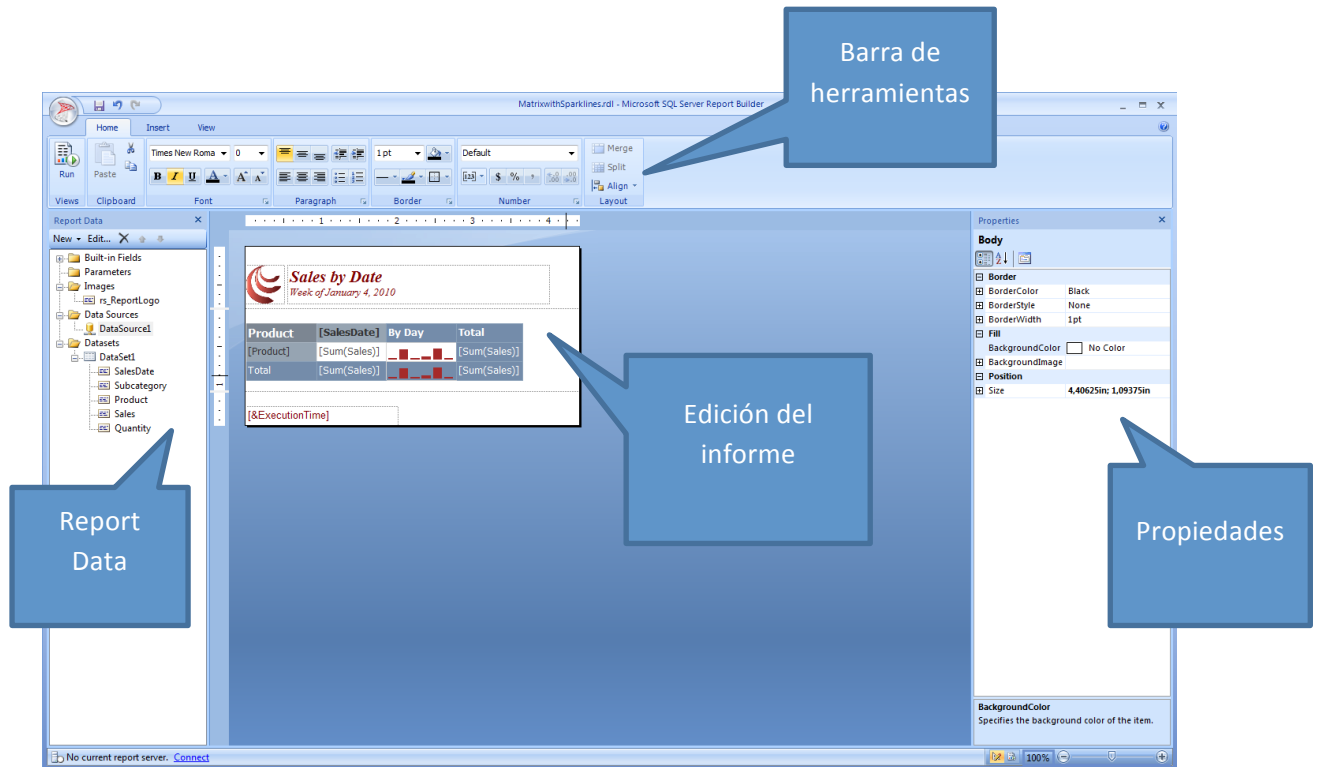


Figura 3-1: Ventana principal de Microsoft Report Builder 3.0

La tercera es la edición del informe, donde iremos arrastrando y creando los elementos que queremos en nuestro informe.

Por último, tenemos la ventana de propiedades que según el elemento seleccionado en la ventana de edición del informe nos mostrará las diferentes propiedades. Hay que añadir que normalmente no es habitual el uso de esta ventana ya que las opciones más comunes para un elemento tienen su propia ventana donde es más fácil cambiarlas, pero a veces para un uso más específico podemos encontrar opciones que no se muestran en las ventanas de propiedades de los elementos.

3.2.2. Data Sources

Los data sources son el primer paso en la creación de un informe. Es la fuente dónde conseguiremos los datos, típicamente será una conexión a un motor de base de datos, Microsoft SQL Server en nuestro caso, aunque nos ofrece otras opciones más avanzadas como Analysis Services. En la figura 2-2 podemos ver el formulario para crear uno, y las diferentes opciones que soporta para elegir el tipo de conexión a la fuente de datos.

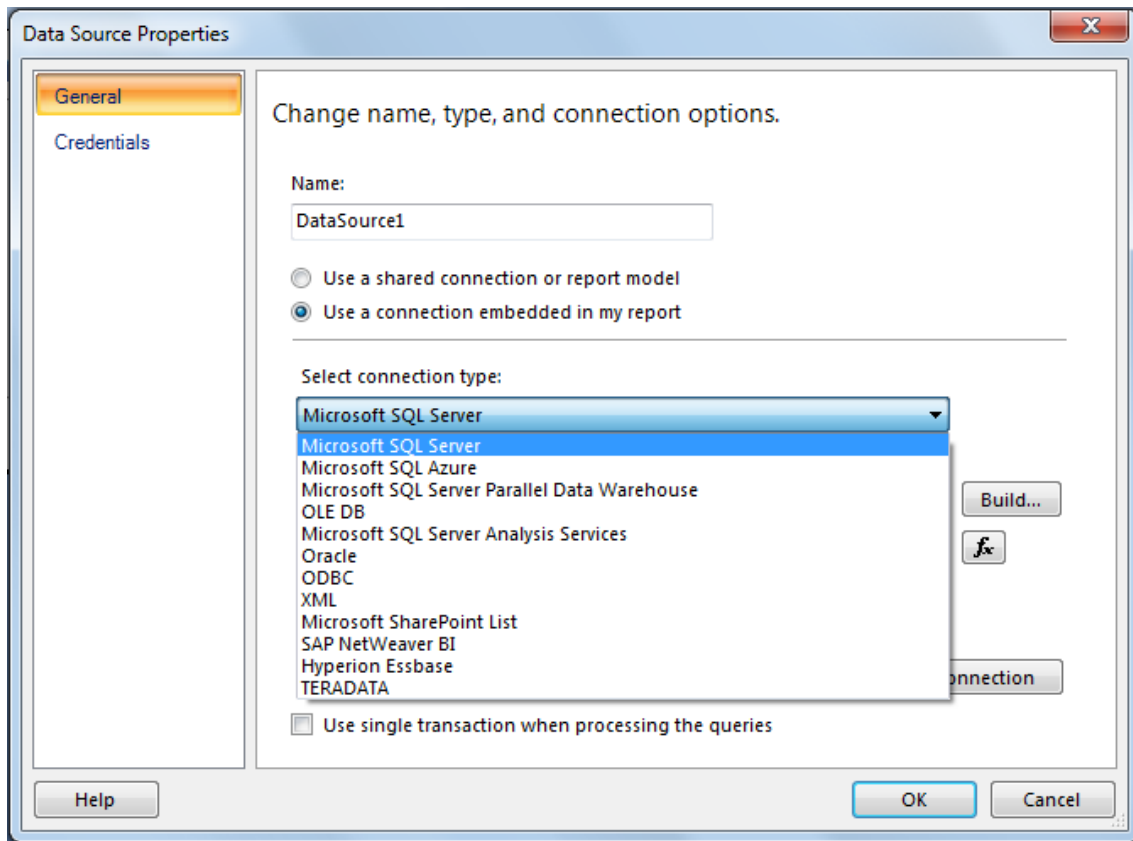


Figura 3-2: Creación de un Data Source

Después de seleccionar el tipo de conexión pasaremos a hacer la conexión rellenando los parámetros más importantes como nombre del servidor y la base de datos en dicho servidor, además de cualquier login necesario para tener permisos.

Adicionalmente, se puede elegir un dataset compartido (que tiene que haber sido creado anteriormente y normalmente alojado en el Report Server), que nos ofrece un datasource con todos los parámetros ya configurados.

3.2.3. Datasets

Los datasets son el conjunto de datos que queremos mostrar o manipular en nuestro informe. Típicamente es una consulta SQL cuando se trabaja sobre una conexión a un servidor de base de datos SQL. En la Figura 2-3 podemos ver el formulario de creación de un dataset.

Primero se escoge el datasource (definido anteriormente) y luego se hace la consulta para traer los datos que necesitamos. El dataset nos ofrecerá los diferentes campos de la consulta para poder usarlos en los informes posteriormente de forma automática.

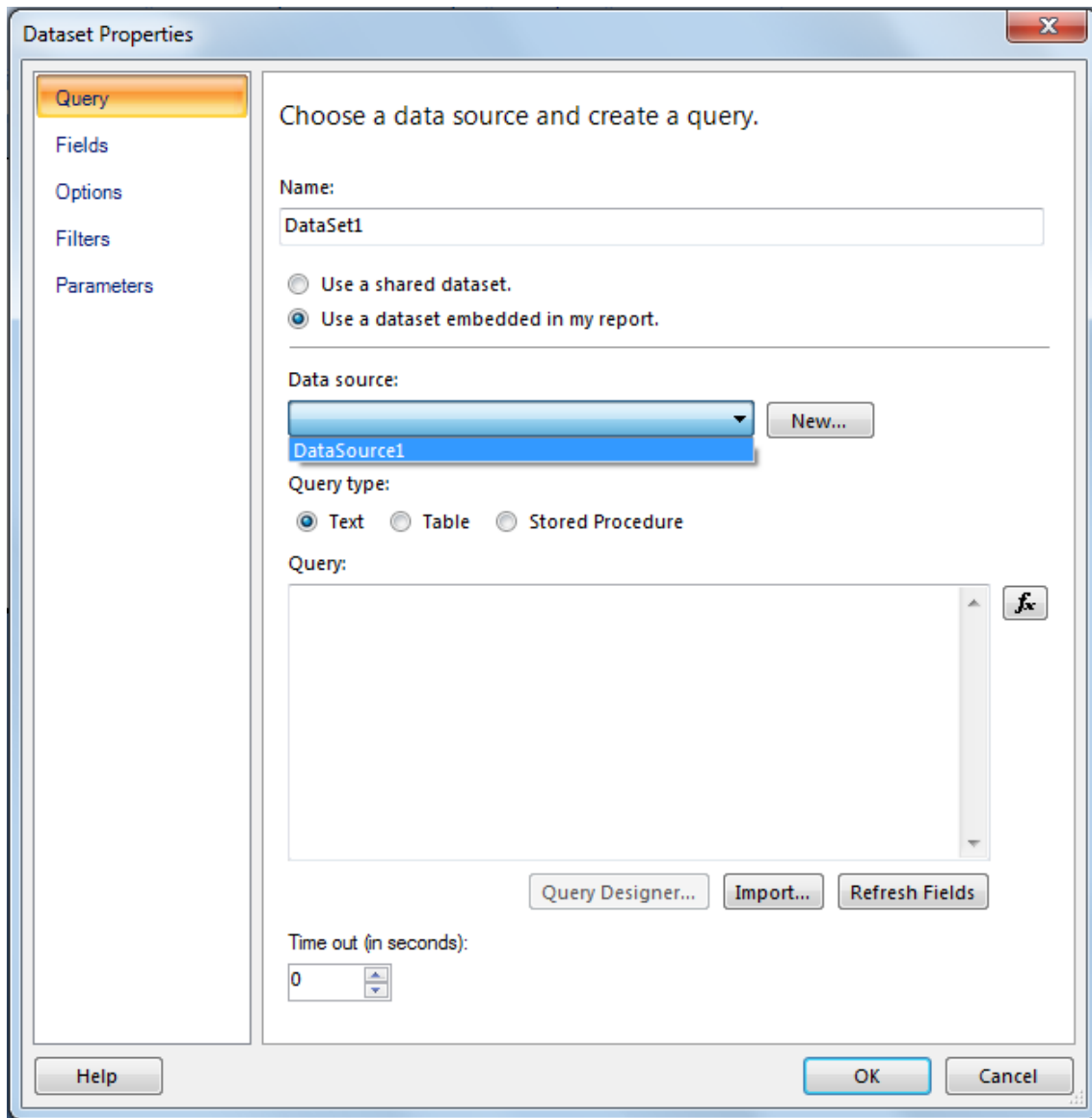


Figura 3-3: Creación de un dataset

Para definir la consulta tenemos diferentes opciones según el tipo de conexión del data source, pero normalmente habrá que usar el 'Query Designer...' (figura 2-4) que nos ofrece de una manera visual organizar nuestra consulta SQL, pudiendo aplicar filtros, seleccionar los campos que queramos de las tablas y realizar diferentes agregaciones para más de una tabla. También tenemos la posibilidad de crear la consulta directamente en modo texto para consultas más complejas.

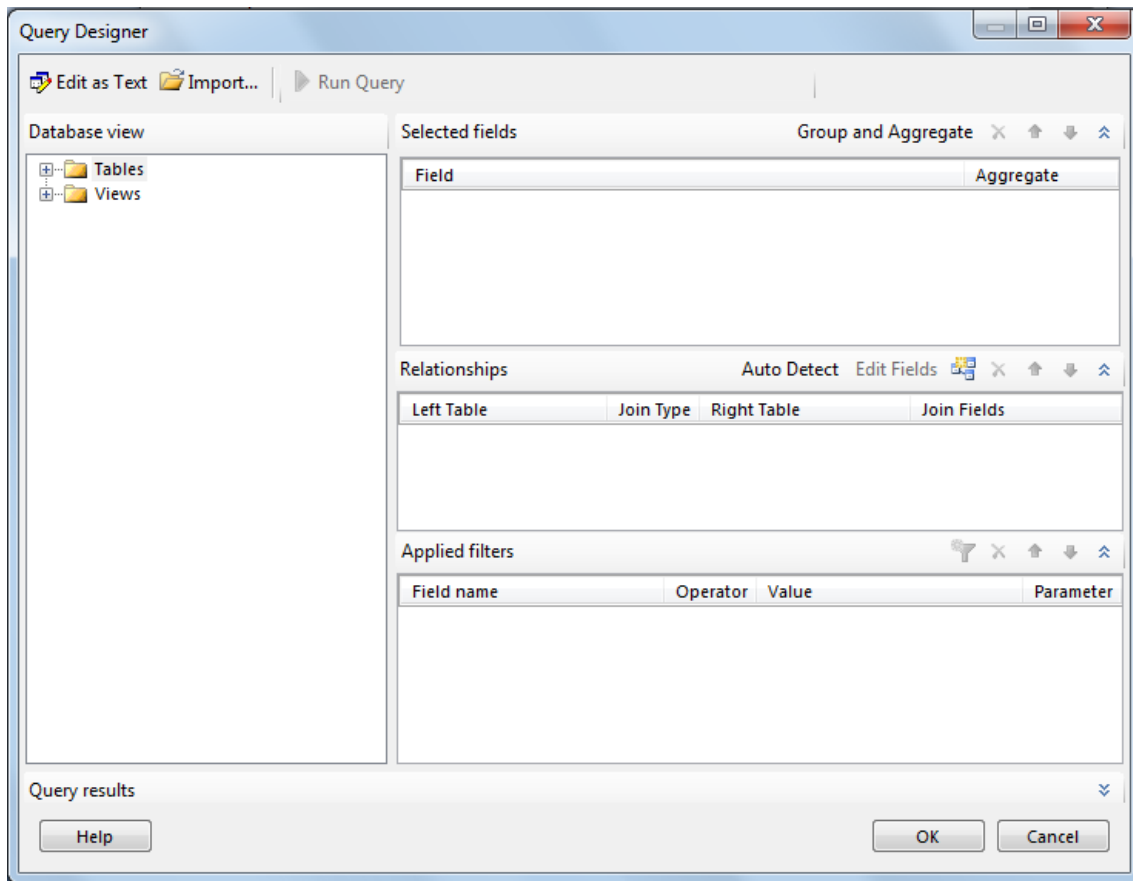


Figura 3-4: Query Designer

En el caso de los datasets también pueden haber sido creados datasets compartido alojados en el Report Server y por tanto utilizarlos sin hacer ninguna configuración, incluida la selección de datasource.

3.2.4. Parámetros

Los parámetros son una parte importante ya que da al usuario que ejecuta el informe la opción de mostrar los datos que considere relevantes aplicando un filtro. Puede ser una variable definida en tiempo de ejecución, ya sea por el usuario o de una manera programada. En Microsoft Report Builder pueden ser usados para controlar los datos que se muestran o variar el diseño del informe. Podemos definir los valores del parámetro disponibles para que el usuario seleccione uno o tener uno o varios valores predeterminados al inicio de la ejecución del informe.

En la figura 2-5 podemos ver el formulario para la creación de un parámetro.

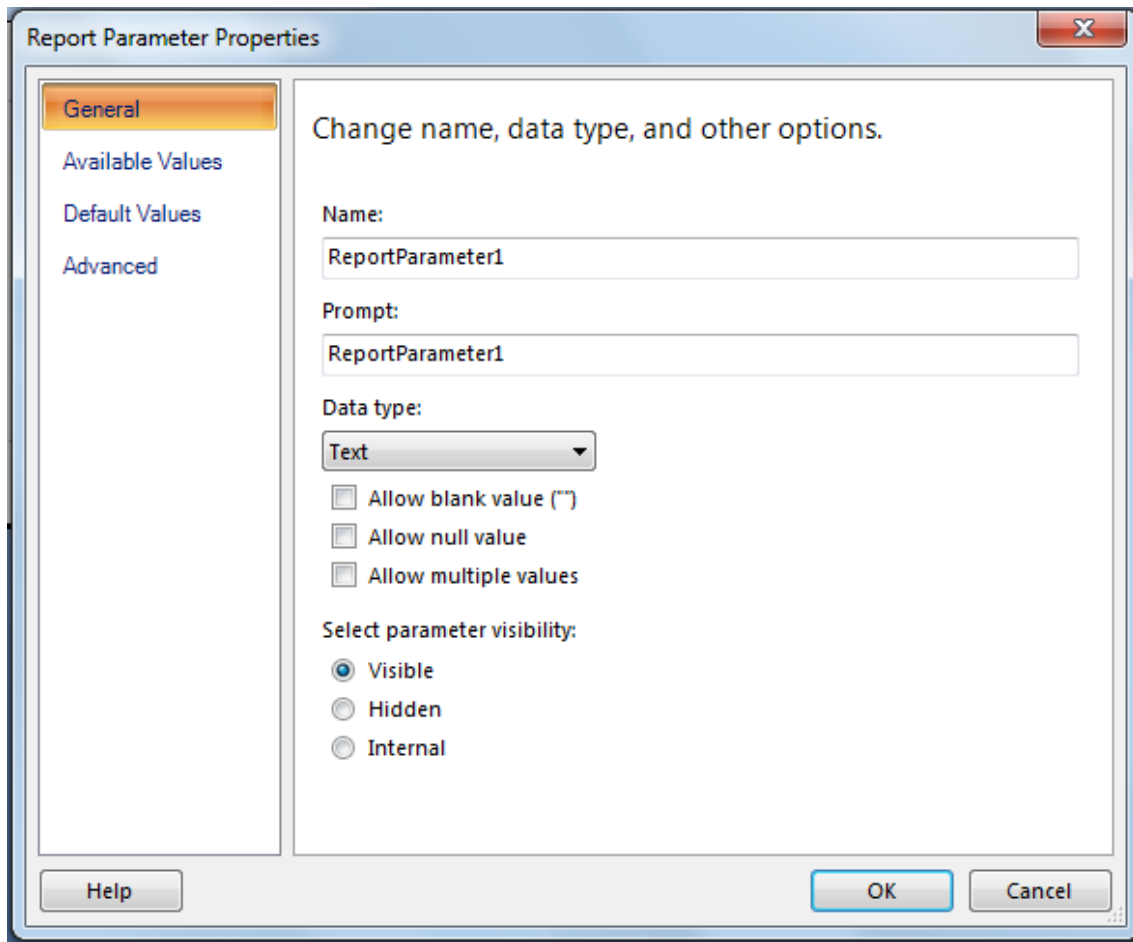


Figura 3-5: Creación de un parámetro

Los parámetros serán utilizados después en la definición de los data sources para que hagan efecto.

3.3. Creación del informe

Una vez vistos los elementos fundamentales, que sería el esqueleto del informe, introduciremos los elementos básicos de presentación (tablas y gráficas) y que conformarían un informe simple, pero completo.

Para la simplificación de la creación del informe, la consulta del dataset incluye ya los datos necesarios y por lo tanto no se necesita ninguna conexión a ningún motor de base de datos, pero aun así crearemos un data source a nuestro servidor de base de datos ya que el dataset necesita siempre uno.

En este caso haremos un informe que refleja las diferentes ventas de productos electrónicos de diferentes tiendas mediante una tabla y una gráfica que muestre las cantidades vendidas de los productos, también crearemos un parámetro que nos permitirá elegir las tiendas de las cuales queremos mostrar datos.

3.3.1. Esqueleto del informe

El primer paso es crear el data source con pero como hemos dicho en nuestro ejemplo no hará falta realmente un servidor de base de datos, por lo tanto crearemos un data source vacío.

Ahora toca crear el dataset que contendrá la información que después explotaremos en el informe, como hemos comentado la consulta incluirá todos los datos y no nos hace falta ningún motor de base de datos. Su nombre será Tiendas y elegiremos la opción 'Use a dataset embedded in my report', seleccionando el datasource creado anteriormente.

Para insertar la consulta usaremos el 'Query Designer' y presionaremos el botón 'Edit as Text' (ver figura 2-6) y pegaremos la siguiente consulta:

```
;WITH CTE (StoreID, Subcategory, Quantity)  
  
AS (  
  
SELECT 200 AS StoreID, 'Digital SLR Cameras' AS Subcategory, 2002 AS Quantity  
  
UNION SELECT 200 AS StoreID, 'Camcorders' AS Subcategory, 1954 AS Quantity  
  
UNION SELECT 200 AS StoreID, 'Accessories' AS Subcategory, 1895 AS Quantity  
  
UNION SELECT 199 AS StoreID, 'Digital Cameras' AS Subcategory, 1849 AS  
Quantity  
  
UNION SELECT 306 AS StoreID, 'Digital SLR Cameras' AS Subcategory, 1579 AS  
Quantity  
  
UNION SELECT 306 AS StoreID, 'Camcorders' AS Subcategory, 1561 AS Quantity  
  
UNION SELECT 306 AS StoreID, 'Digital Cameras' AS Subcategory, 1553 AS  
Quantity  
  
UNION SELECT 306 AS StoreID, 'Accessories' AS Subcategory, 1534 AS Quantity  
  
UNION SELECT 307 AS StoreID, 'Accessories' AS Subcategory, 1755 AS Quantity  
  
UNION SELECT 307 AS StoreID, 'Camcorders' AS Subcategory, 1631 AS Quantity  
  
UNION SELECT 307 AS StoreID, 'Digital SLR Cameras' AS Subcategory, 1772 AS  
Quantity)  
  
SELECT StoreID, Subcategory, Quantity  
  
FROM CTE
```

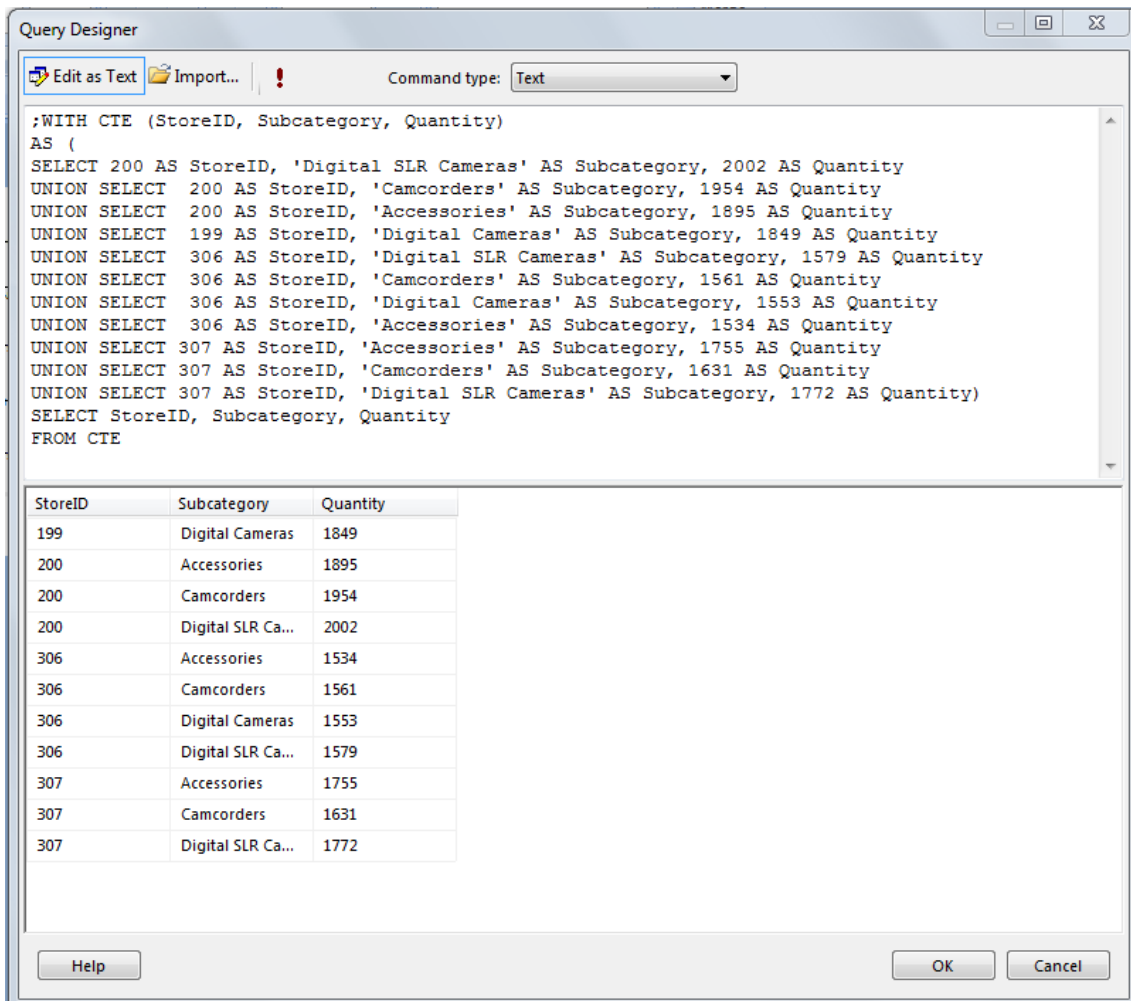


Figura 3-6: Query Designer en modo texto para insertar nuestra consulta

Podemos ejecutar la consulta para ver los campos de los que disponemos después en nuestro informe y ver que no tenemos ningún error con la consulta.

Ya tenemos definido el dataset y como vemos en la ventana 'Report Data' también podemos ver los campos que podremos usar en nuestro informe (ver figura 2-7), en nuestro caso los parámetros StoreID, Subcategory y Quantity.

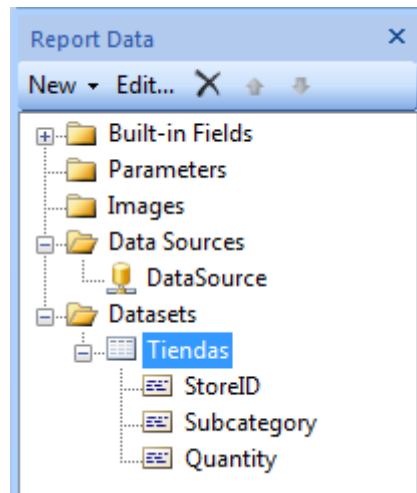


Figura 3-7: Dataset definido con sus respectivos campos

Ahora crearemos el parámetro que nos dejará elegir la tienda de la cual queremos ver los datos. Como podemos ver la consulta sólo tiene ID's de las tiendas y no sus nombre por lo tanto crearemos otro dataset que nos de esa información y podamos mostrar el nombre de la tienda en vez de su ID. El dataset se llamará 'NombreTiendas' y los campos 'Store ID' y 'Store Name'. Se crea de la misma manera que el anterior dataset pero ahora la consulta será la siguiente:

```
SELECT 200 AS StoreID, 'Contoso Catalog Store' as StoreName
```

```
UNION SELECT 199 AS StoreID, 'Contoso North America Online Store' as StoreName
```

```
UNION SELECT 307 AS StoreID, 'Contoso Asia Online Store' as StoreName
```

```
UNION SELECT 306 AS StoreID, 'Contoso Europe Online Store' as StoreName
```

Ejecuta la consulta para ver los campos disponibles (StoreID y StoreName) y ver que no hay ningún error.

Vamos a crear ahora el parámetro desde la carpeta 'Parameters' de la ventana 'Report Data', por lo tanto sobre la carpeta pulsamos botón derecho y seleccionamos la opción 'Add Parameter...'. Rellenamos la primera pestaña con el nombre del parámetro Tienda y en la casilla 'Prompt' 'Tienda:' (este es el texto que aparecerá a la izquierda del parámetro cuando se ejecute el informe y el texto que vera el usuario). Seleccionamos como 'Data Type' 'Integer' y seleccionamos que el parámetro puede tener múltiples valores, así podremos seleccionar más de una tienda para nuestros informes.

Ahora sólo falta definir los valores que queremos mostrar en los parámetros para que el usuario pueda seleccionarlos, accedemos a la pestaña 'Available Values' (ver figura 2-8) y seleccionamos la opción 'Get values from a query' donde seleccionaremos el dataset creado anteriormente, NombreTiendas. Sólo falta rellenar el valor que tendrá el parámetro, en nuestro caso será el ID dela tienda (Campo StoreID), y la etiqueta del parámetro (el texto que

usuario verá al usar el desplegable de los parámetros) que será el nombre de la tienda (campo StoreName).

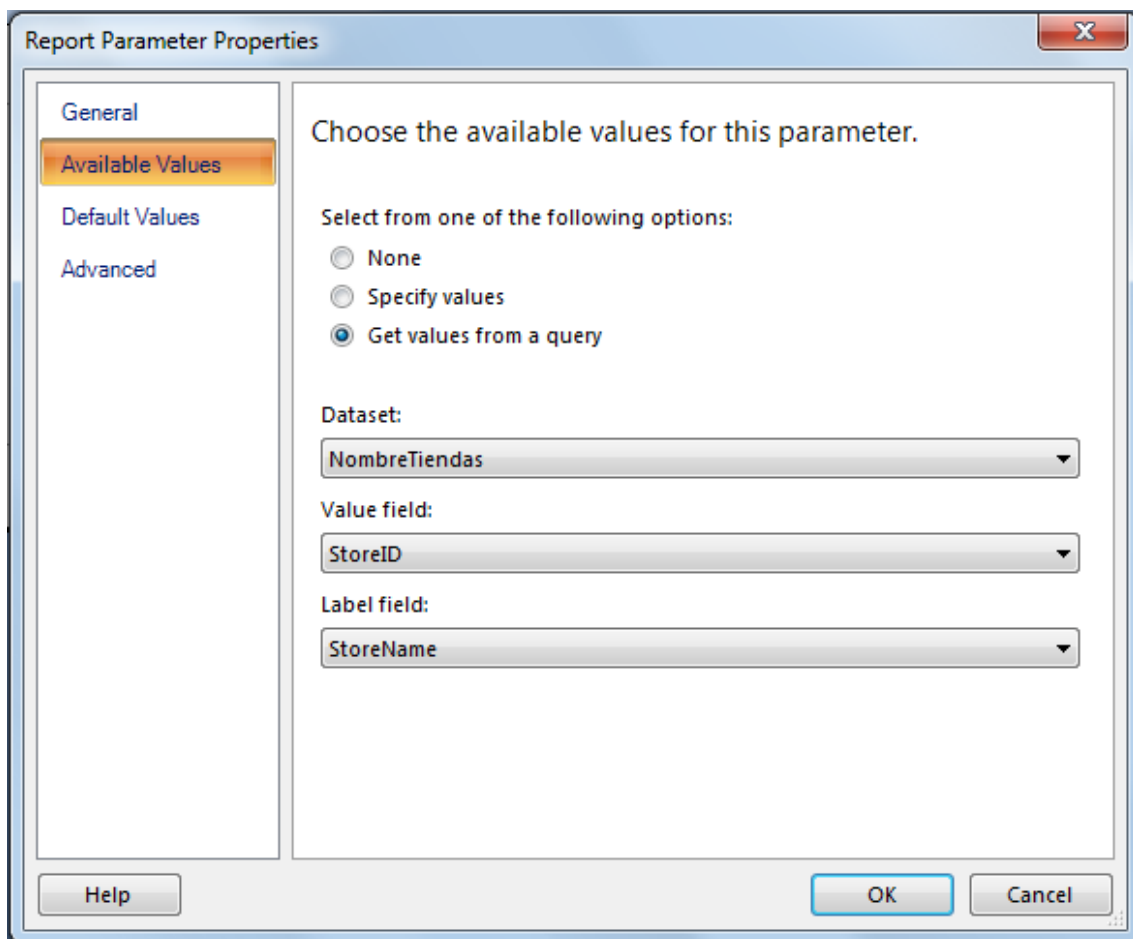


Figura 3-8: Valores disponibles en nuestro parámetro

Ahora ya podemos dar por finalizado la creación de parámetro.

Vamos a ejecutar ahora el informe y comprobar que no hay ningún error, para ello pulsamos el botón 'Run' de la barra de herramientas y veremos una ventana como la de la figura 2-9. Podemos ver como se muestra el nombre de las tiendas aunque de momento no es de mucha utilidad ya que no hemos diseñado la parte de presentación del informe.

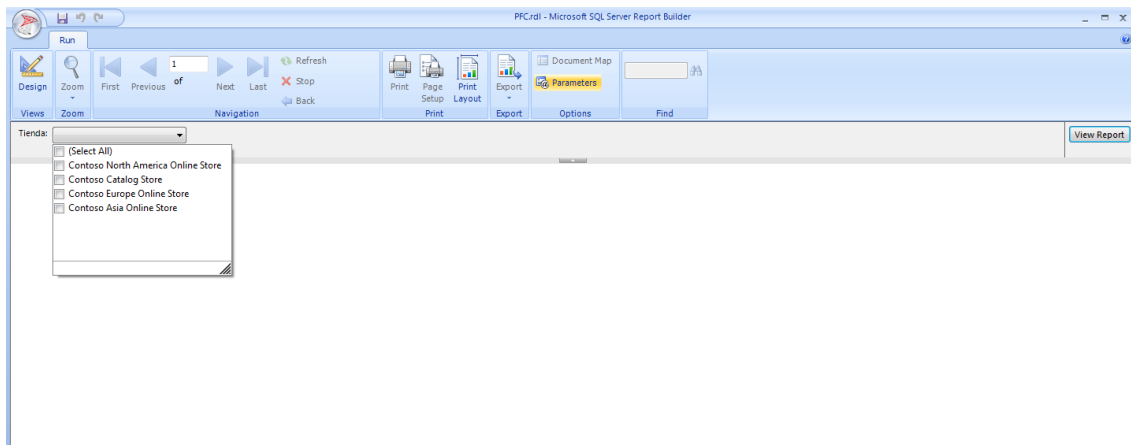


Figura 3-9: Ejecución del informe con el parámetro ya creado

Para la parte de presentación haremos una tabla que represente las ventas de las tiendas de los diferentes productos electrónicos y una gráfica para que lo muestre de manera visual. Empezaremos haciendo un vistazo rápido a la barra de herramientas (ver figura 2-10), como podemos ver Microsoft Report Builder nos ofrece diferentes elementos para el diseño de los informes desde los más típicos, tablas y gráficas a algunos más complejos y más visualmente impactantes como mapas y diferentes elementos como KPI's, también nos ofrece las típicas opciones de insertar imágenes y un encabezado y pie de página, para un resultado más profesional.

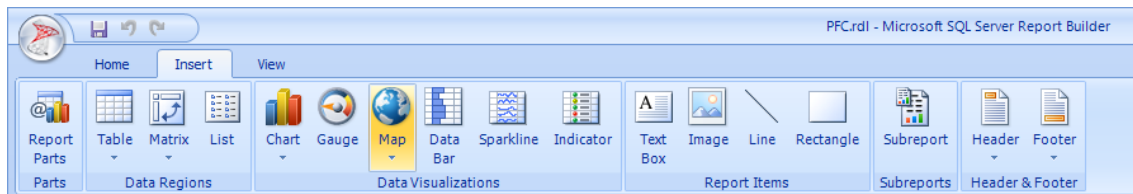


Figura 3-10: Barra de herramientas para insertar los elementos de presentación

Insertamos la tabla pulsando sobre el botón de 'Table' y seleccionando después la opción 'Table Wizard', que iniciará un asistente para la creación de una tabla. En la ventana seleccionamos el dataset 'Tiendas' que es el que contiene toda la información. Pulsamos 'Next' y en la nueva ventana (ver Figura 2-11) seleccionamos los campos del dataset elegido que queremos mostrar como columnas, filas y valores.

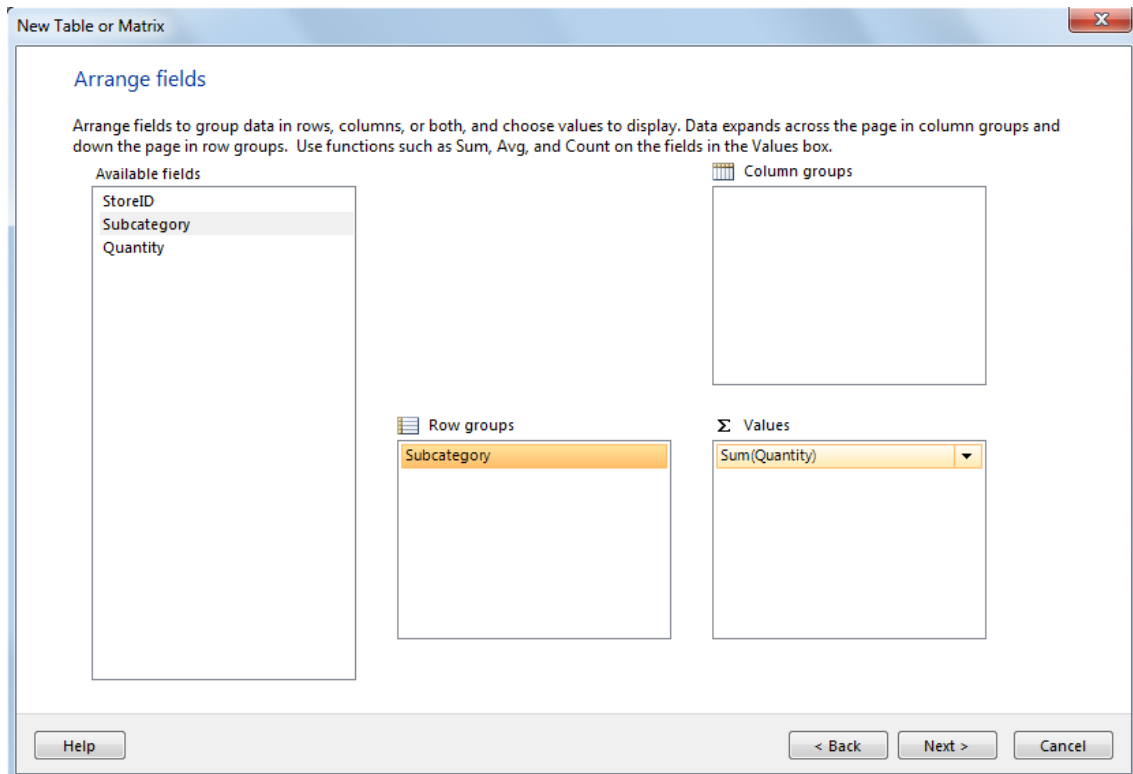


Figura 3-11: Asistente para crear tablas a partir de un dataset

En la siguiente ventana podemos elegir la disposición de los elementos, permitiendo si se pueden colapsar los subtotales, etc. para nuestra tabla. Después seleccionamos un estilo (gama de colores para la tabla) y finalizamos el asistente teniendo la tabla ya en el informe (figura 2-12).

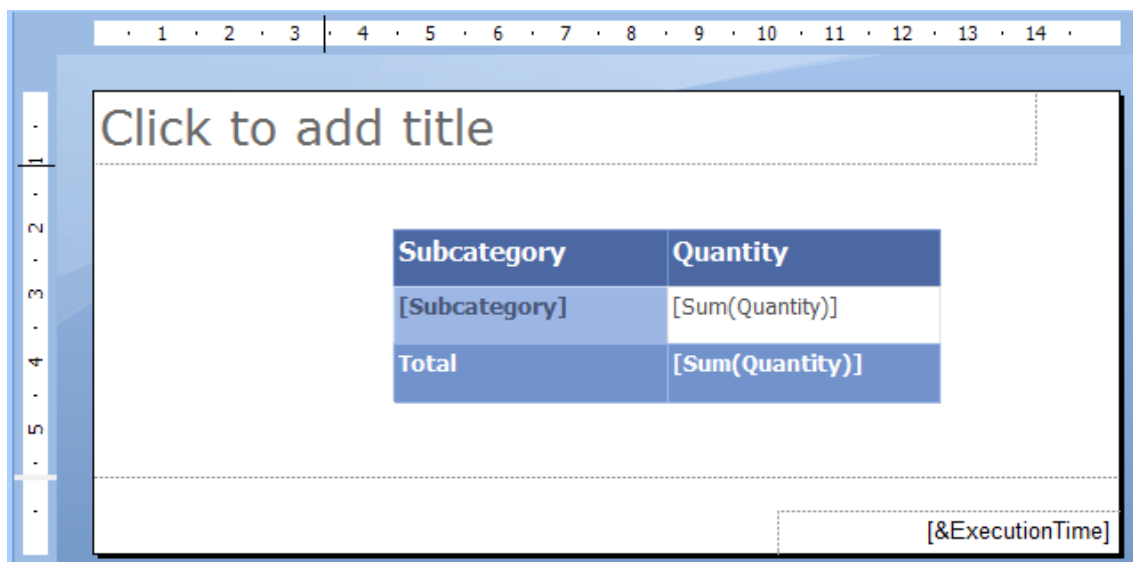


Figura 3-12: Ventana de edición del informe con nuestra tabla ya creada

Si ejecutamos ahora el informe podemos ver que después de la selección de parámetros nos muestra ya datos, pero cambiando los parámetros siempre salen los mismos datos. Eso es porque no hemos “enlazado” lo que el usuario elige con el dataset que utilizamos para la tabla, para eso tendremos que modificar la propiedad ‘Filters’ del dataset ‘Tiendas’ (figura 2-13). Para

ello vamos al Report Data y con el botón derecho seleccionamos la opción 'Properties' del data set Tiendas. Como podemos ver seleccionamos el campo del dataset por el cual queremos filtrar, seleccionamos el operador, en nuestro caso usaremos 'In' ya que es un parámetro que puede tener múltiples valores, después escribimos el nombre del parámetro, @Tienda, entre corchetes. Ahora ya podemos ejecutar el informe otra vez viendo como ahora sí dependiendo de los parámetros elegidos se muestran unos datos o otros.

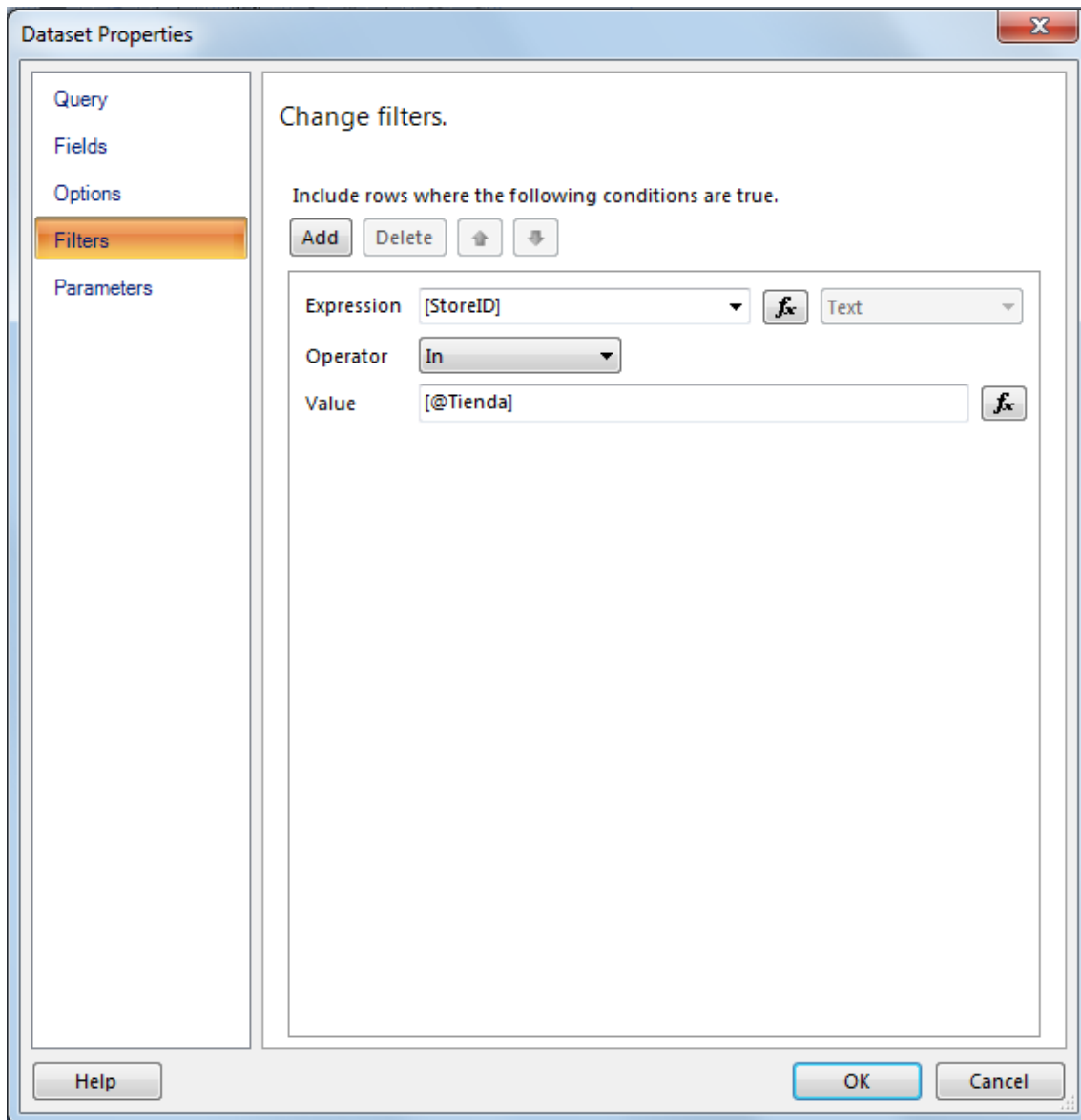


Figura 3-13: Ventana de propiedad del dataset

Ahora toca crear la gráfica para mostrar las cantidades vendidas por las distintas categorías de los productos. Iremos a la barra de herramienta y seleccionaremos la opción 'Chart' usando la opción 'Chart Wizard' para iniciar el asistente para la creación de gráficas (figura 2-14). Es muy parecido al asistente para crear las tablas, primero seleccionamos el dataset, 'Tiendas' y después seleccionamos el tipo de gráfica que queramos, en nuestro caso elegiremos columnas ya que es la mejor manera de representar los valores que queremos. Después seleccionamos los campos del dataset que queremos mostrar en la gráfica (serán los mismos que elegimos

para la tabla) y para finalizar seleccionamos el estilo que le queremos dar. Por fin ya tenemos creado nuestro informe, ahora sólo hace falta unos retoques para que esté completo.

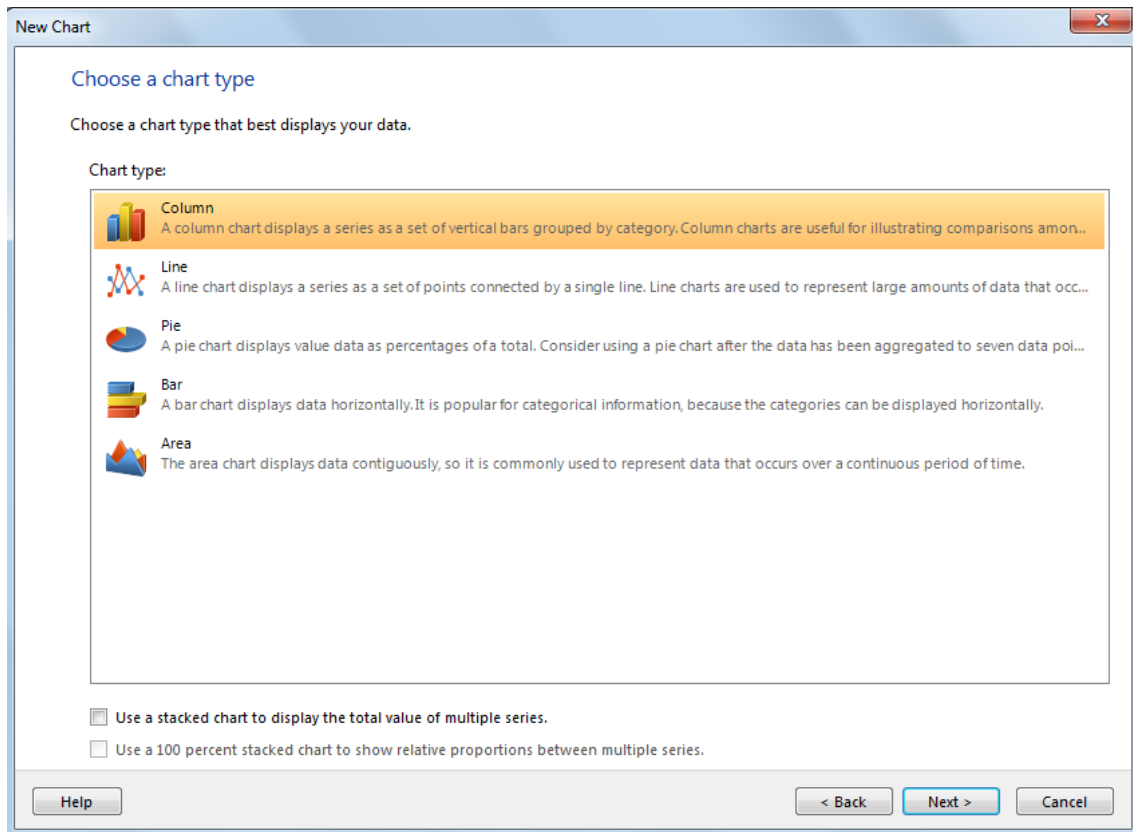


Figura 3-14: Asistente para la creación de gráficas con las distintas opciones

Podemos empezar por poner un título al informe pudiendo mostrar los nombres de las tiendas de las que se están mostrando datos, para ello seleccionamos el cuadro de texto y ponemos el título que queramos. Para mostrar el nombre de las tiendas hay que hacer un poco más de trabajo, primero arrastramos el parámetro Tienda desde la ventana de 'Report Data' al texto que estamos editando, hacemos doble click y podemos editar el 'Placeholder' que se ha creado automáticamente (figura 2-15) donde cambiaremos el valor por la siguiente expresión:

`=Join(Parameters!Tienda.Label, ",")`

Sólo queda ejecutar el informe y ver el resultado final (figura 2-16).

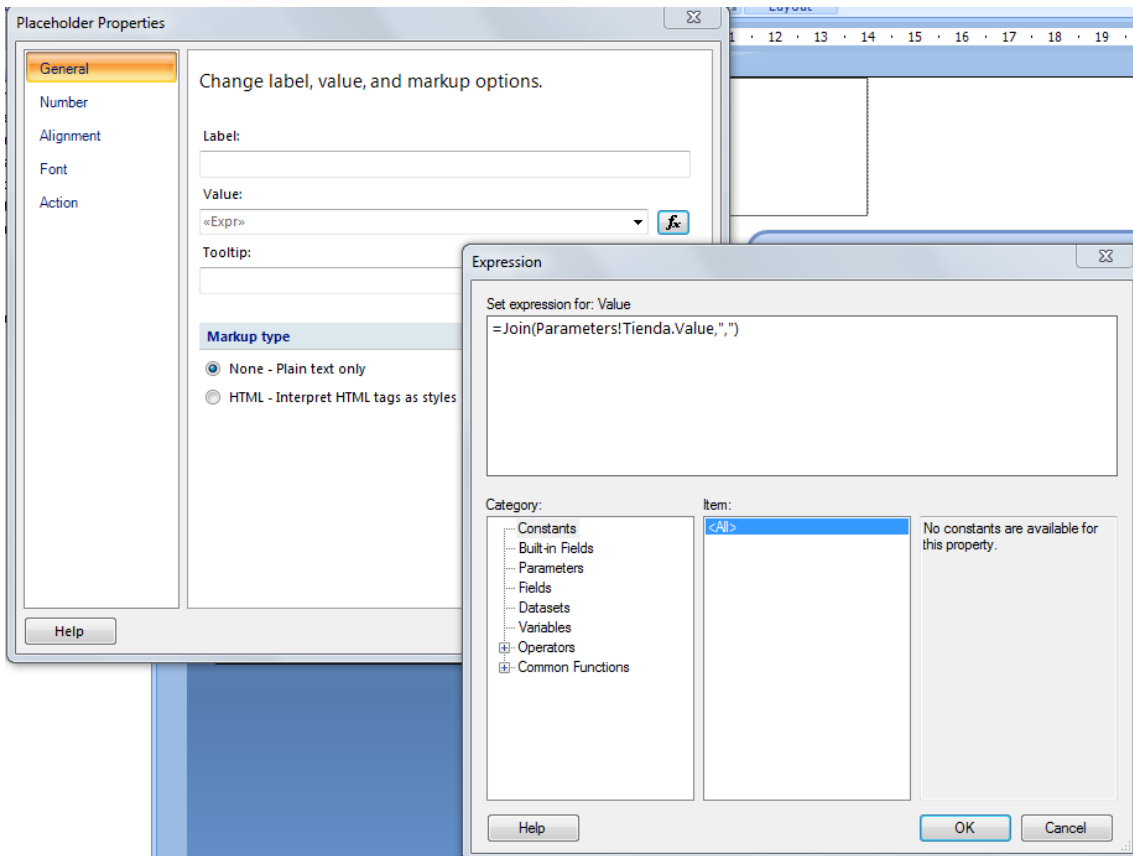


Figura 3-15: Editando el título para que muestre el nombre de las tiendas

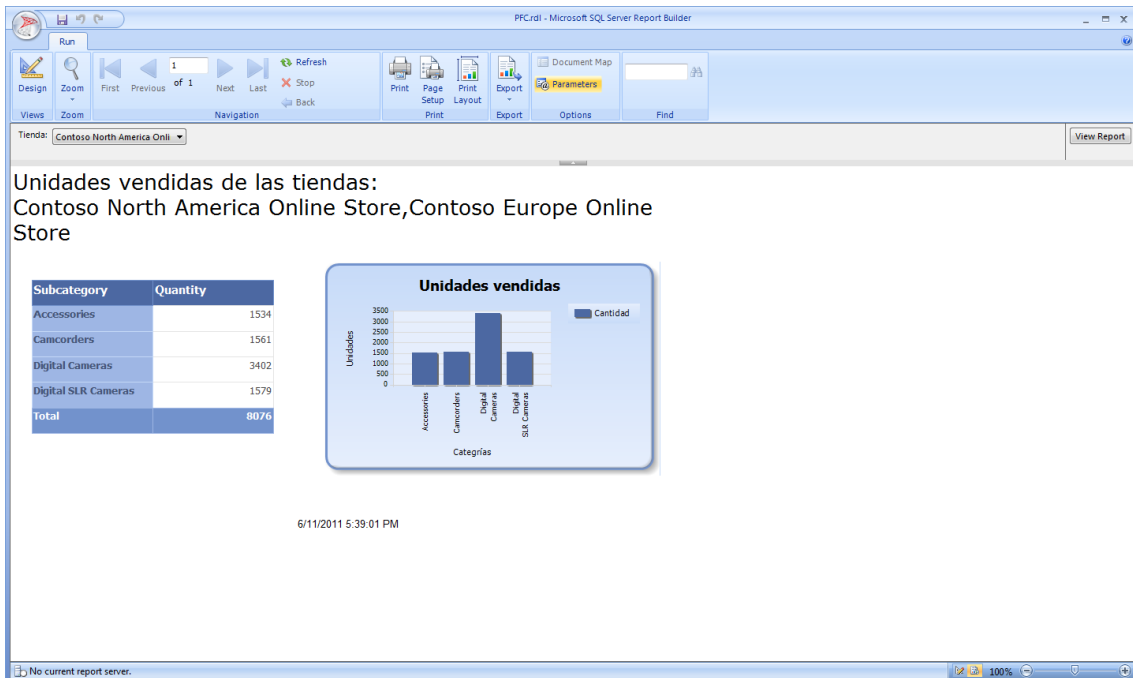


Figura 3-16: Ejecución del informe en su versión final

4. Estado del arte

4.1. Introducción

Actualmente, todo y los avances que se han hecho en la formalización y procesos en la rama ingeniería del software, aún no está tan extendida la idea de los cuadros de mandos como en otras ramas (economía, planificación empresarial, etc.) y por lo tanto la cantidad de ejemplos a estudiar es bastante escasa, aunque hay herramientas que son realmente interesantes y que se acercan a la integración total permitiendo una visión global del proceso realizado a lo largo de la vida del producto.

En este capítulo haremos un repaso a las soluciones existentes en el mercado y que puedan ser interesantes en el campo que estamos estudiando, estudio del desarrollo de software sobre diferentes versiones de un mismo producto.

Hay que decir que no hay ninguna herramienta que por sí sola sea considerada sólo como cuadro de mandos, lo más habitual es que un producto de gestión de proyectos se le añada la posibilidad de visualizar un cuadro de mandos (gráficas, tablas, etc.) respecto a los datos que maneja el programa. Nos centraremos en las herramientas que dan soporte a metodologías de desarrollo ágil, principalmente a la metodología SCRUM.

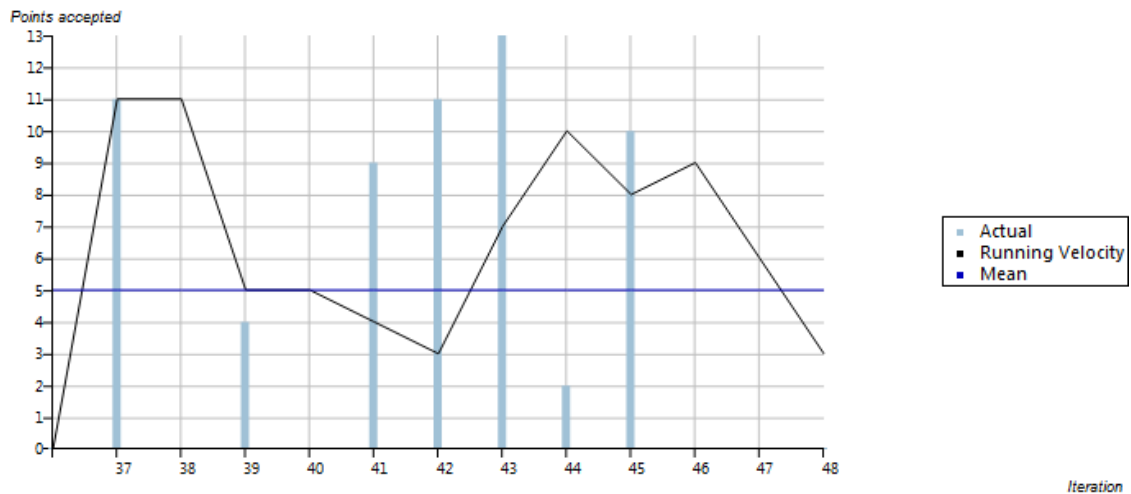
Por lo tanto comentaremos las funcionalidades del producto y cómo implementan un cuadro de mandos, poniendo en especial foco aquellas que nos ofrecen información del proyecto entre versiones.

Las siguientes herramientas han sido extraídos de la referencia [3].

4.2. Pivotal Tracker

Pivotal Tracker (<http://www.pivotaltracker.com>) es una aplicación web para la gestión de proyectos para el desarrollo de software. No se basa en ninguna metodología en concreto y ofrece un workflow ágil, directo y sencillo, basado en 'stories', pequeñas tareas con posibilidad de calificarlas con diferentes puntos que añaden valor al producto final.

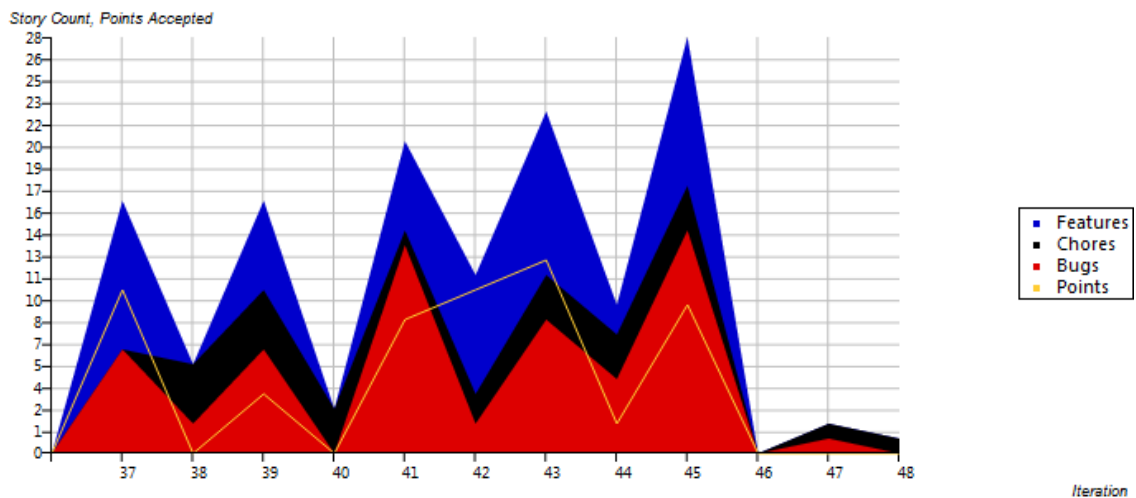
Tiene su propia métrica para calcular la velocidad de una iteración o versión, utilizando las 'stories' ofrece un panorama para estimar el número de iteraciones hasta finalizar el producto. Un ejemplo de esta métrica se puede ver en la figura 3-1. La gráfica muestra las diferentes iteraciones en el eje X, en el eje Y se representa la suma de los puntos de las 'stories' aceptadas en esa versión, por tanto al final podemos ver en cada versión si hemos ido avanzado más rápidamente o lentamente que en previas versiones. También nos muestra la media de los puntos de las iteraciones que hemos elegido y como en diferentes iteraciones hemos estado por encima de la media o cuáles nos han hecho bajar la media.



Over the project's visible history, the mean (average) Velocity is 5. The standard deviation is 5.1. The volatility is 101.8%.

Figura 4-1: Gráfica que representa la velocidad del proyecto

Otra interesante gráfica que nos ofrece Pivotal Tracker es el número de características, tareas, errores y puntos por iteración, un ejemplo se puede ver en la figura 3-2. En esta gráfica también se muestra la media de cada valor, y el usuario puede elegir los elementos que se quieren mostrar en la gráfica.



In the average iteration, 4.4 features, 2.1 chores, and 5.2 bugs are accepted.

Include: Features Chores Bugs
Stack?

Figura 4-2: Número de tareas por iteración

Pivotal Tracker es gratuito para proyectos públicos y ofrece distintos rangos de precios para proyectos privados y que necesiten un número de usuarios mayor.

4.3. Scrumpy

Scrumpy (<http://www.scrumpytool.com/>) es una herramienta de escritorio basada en la metodología SCRUM.

Como podemos ver en la figura 3-3, Scrumpy nos ofrece una gráfica por Sprints (según la documentación del proyecto un sprint es una longitud predeterminada de tiempo en el cual el

equipo va a trabajar un determinado número de tareas), muy parecido al concepto de versión que nosotros utilizaremos en la concepción del cuadro de mandos.

En la gráfica nos muestra lo que el programador ha hecho en cada sprint clasificándolo en Done (hecho) y Committed (entregado o que se ha incluido en la versión). Esta gráfica puede ser de interés para revisar como se ha ido integrando lo que el programador ha hecho con lo entregado en cada sprint/versión. Aunque la herramienta es muy básica, puede ayudar a pequeños desarrollos con poca gente a una mejora en la producción del software.

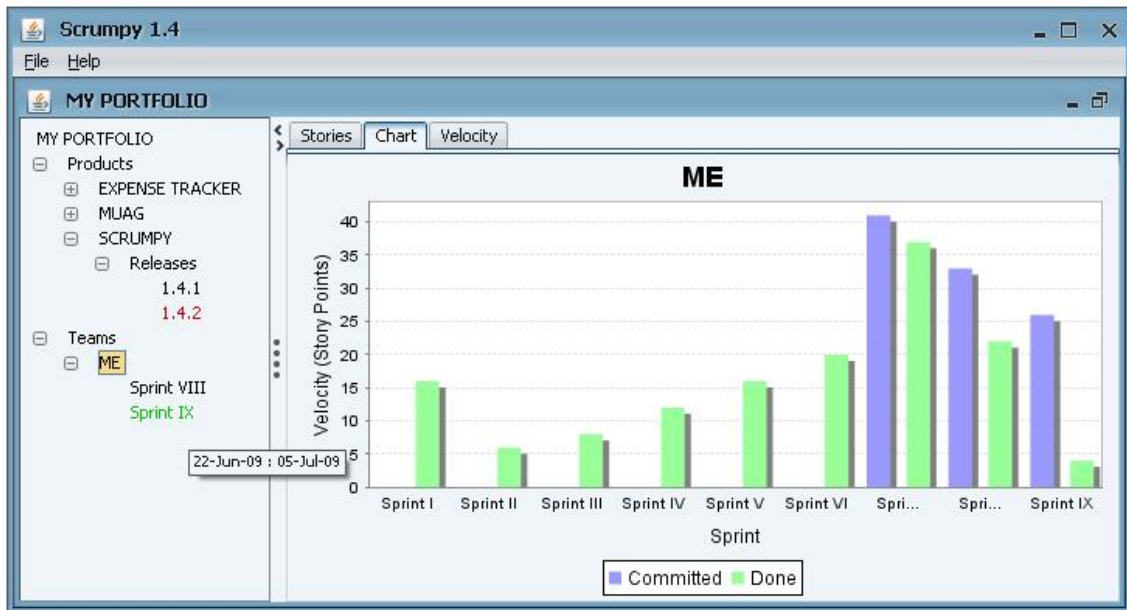


Figura 4-3: Trabajo por Sprints para un programador

ScrumPy es de uso gratuito y nos ofrece un conjunto básico para implementar la metodología SCRUM en cualquier proyecto de desarrollo de software.

4.4. Icescrum

Icescrum (<http://www.icescrum.org>) es una herramienta web basado en la metodología de trabajo SCRUM que ofrece un cuadro de mandos con múltiples gráficas, esta herramienta también se basa en las 'stories' para cuantificar el trabajo.

En la figura 3-4 podemos ver los diferentes tipos de 'stories' (sugeridas, aceptadas, planeadas, estimadas, en progreso y hechas) por sprint (o versión) de una manera acumulativa pudiendo ver el progreso de todas ellas en cada sprint. En esta gráfica sólo se contabiliza el número de 'stories' no contabilizando sus puntos. Como podemos ver en la gráfica los datos aquí mostrados nos pueden ayudar a tener una visión global del trabajo realizado.

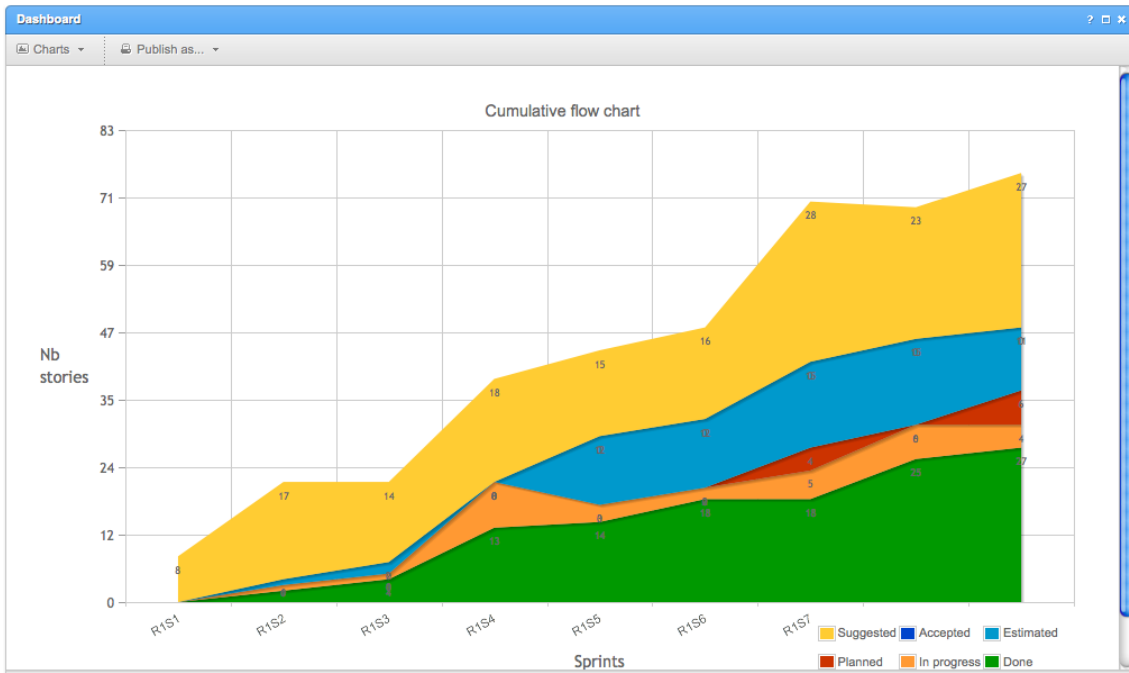


Figura 4-4: Gráfica acumulativa para las diferentes 'stories' según su tipo

Otra gráfica que nos ofrece Icescrum es la visualización de los puntos hechos y los puntos totales de cada versión (figura 3-5). Como podemos deducir de la figura en cada sprint se van añadiendo 'stories' que incrementan los puntos y la funcionalidad del producto, y también se completan 'stories' añadidas en pasadas versiones. Para un producto acabado la gráfica terminaría con el mismo punto para los puntos hechos y todos.

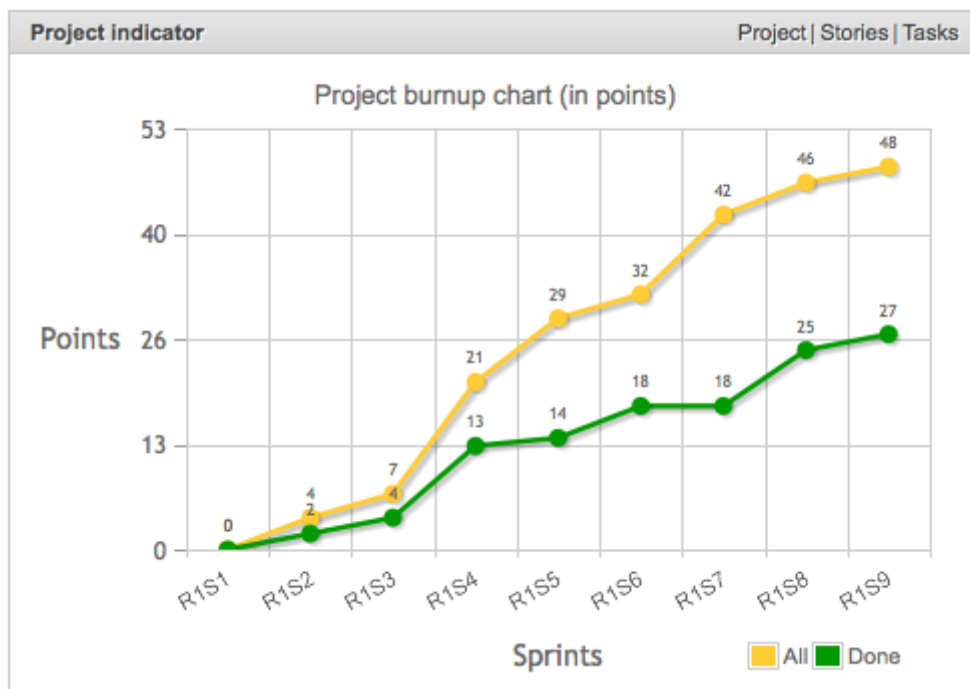


Figura 4-5: Visión general del proyecto por sprints

4.5. Scrumdesk

Scrumdesk (<http://www.scrumdesk.com>) es una aplicación de escritorio de pago para el desarrollo de productos software usando la metodología ágil SCRUM.

Scrumdesk nos ofrece diferentes informes para presentarnos información sobre la evolución de un proyecto entre versiones.

En la figura 3-6 podemos ver como Scrumdesk ha definido una métrica velocidad que nos ayuda a ver como de rápido estamos completando las tareas marcadas. También nos muestra la media de nuestras 3 peores velocidades y la media de las últimas 8 velocidades.



Figura 4-6: Velocidad de los sprints

En la figura 3-7 tenemos un Burn Down Chart pero en este caso es entre versiones, en vez del que se usa comúnmente que es por días. Nos muestra el progreso del desarrollo del software entre versiones (sprints). También nos muestra una estimación de cuando se acabará el proyecto basándose en las últimas 8 velocidades del proyecto y las 3 peores velocidades.

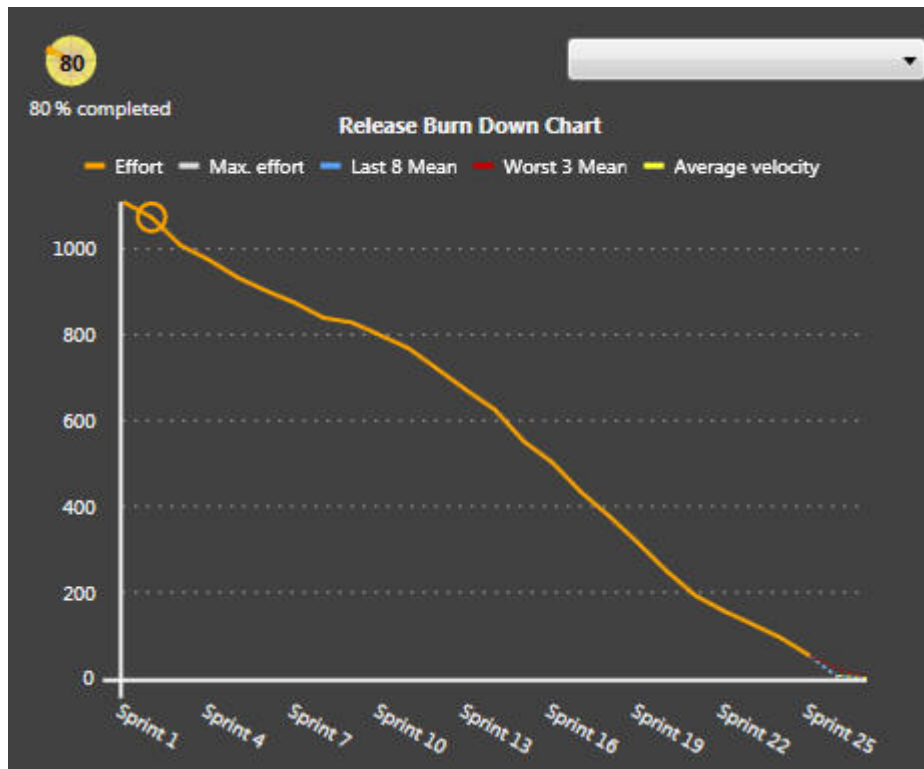


Figura 4-7: Burn Down Chart entre versiones

4.6. ScrumWorks

ScrumWorks (<http://www.open.collab.net/products/scrumworks/>) es una herramienta para la ayuda de desarrollo de productos con metodologías ágiles. Para el acceso a las gráficas hace falta la herramienta de escritorio basada en Java, pudiendo publicar las gráficas que nos interesan en el portal web de ScrumWorks para que el equipo pueda acceder más fácilmente.

ScrumWorks es una de las herramientas que dispone de más gráficas para comparar diferentes parámetros entre versiones. En la figura 3-8 podemos ver el número de puntos acumulados para cada iteración (eje Y), pudiendo comparar en que versiones (eje X) se ha puesto más esfuerzo respecto a otras.

ScrumWorks también nos ofrece una métrica de velocidad y dándonos una estimación de cuando el producto puede estar finalizando respecto pasadas iteraciones (figura 3-9). Como podemos ver nos muestra en columnas el esfuerzo restante para acabar el producto que permite sacar dos líneas de tendencia que marcan como se ha ido añadiendo o eliminando trabajo (línea azul) o cuantos puntos tendrá la siguiente iteración, también nos muestra la que será la nueva base de donde partirá la siguiente iteración (línea negra)-

Por último tenemos la métrica de velocidad (figura 3-10) que contiene la misma información que la primera gráfica vista pero diciéndonos cual es el número de puntos por iteración que hacemos. En el caso de ejemplo es 39 puntos por iteración, que nos podría permitir gestionar mejor nuestro recursos para futuras iteraciones.

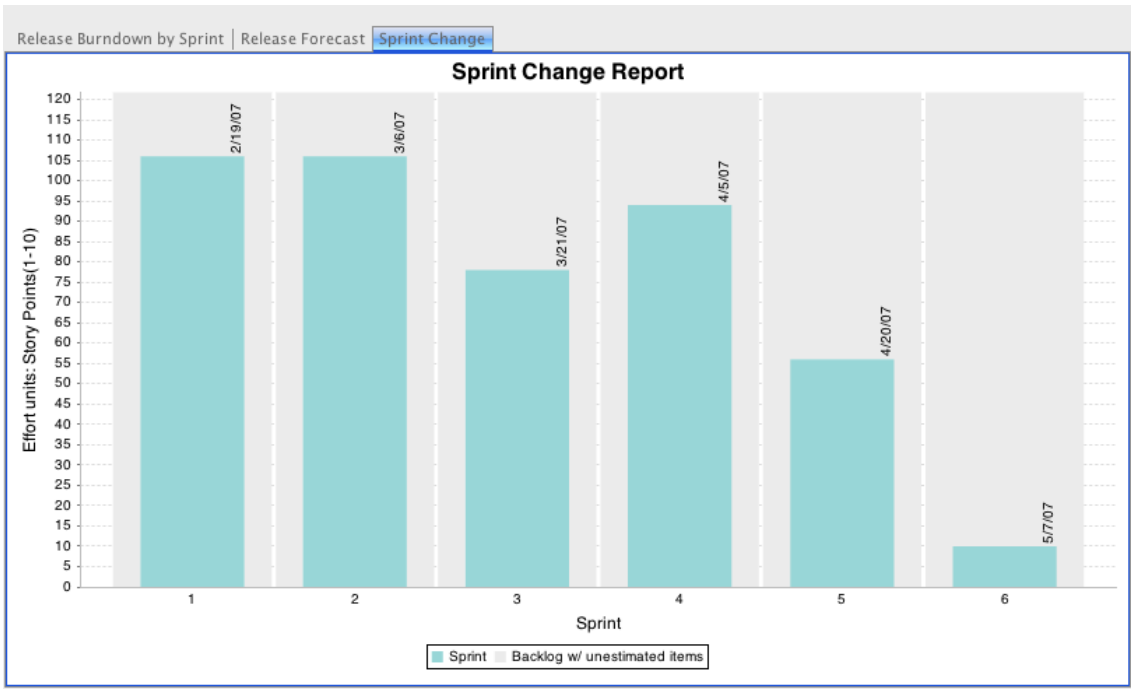


Figura 4-8: Puntos de las 'stories' a lo largo de las iteraciones

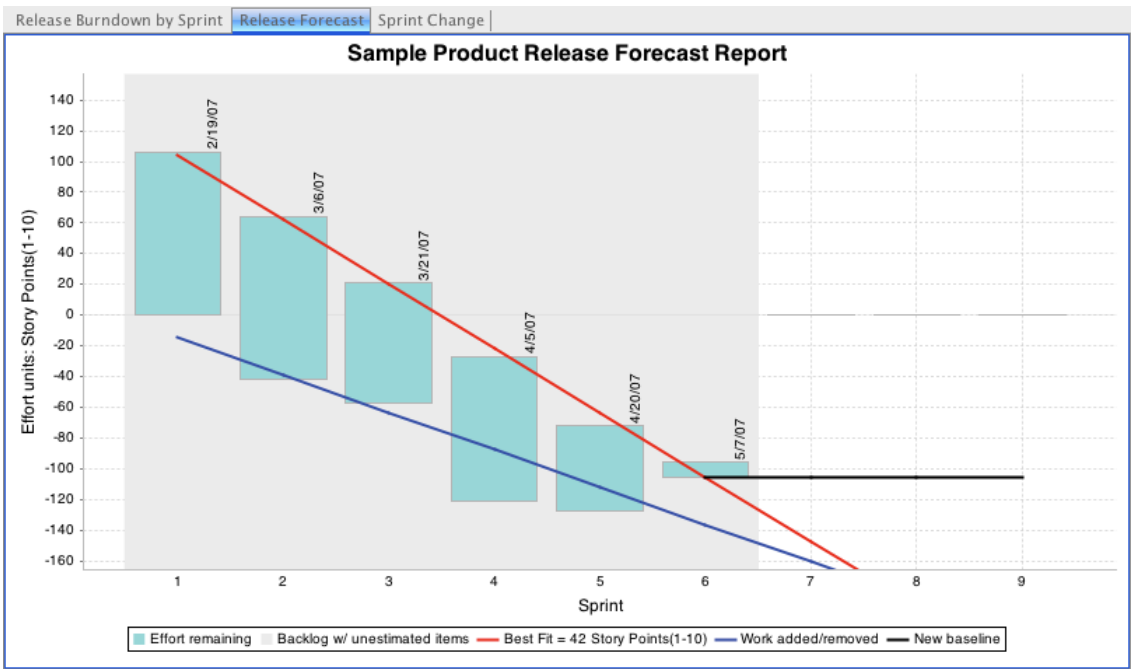


Figura 4-9: Estimación de la finalización del producto

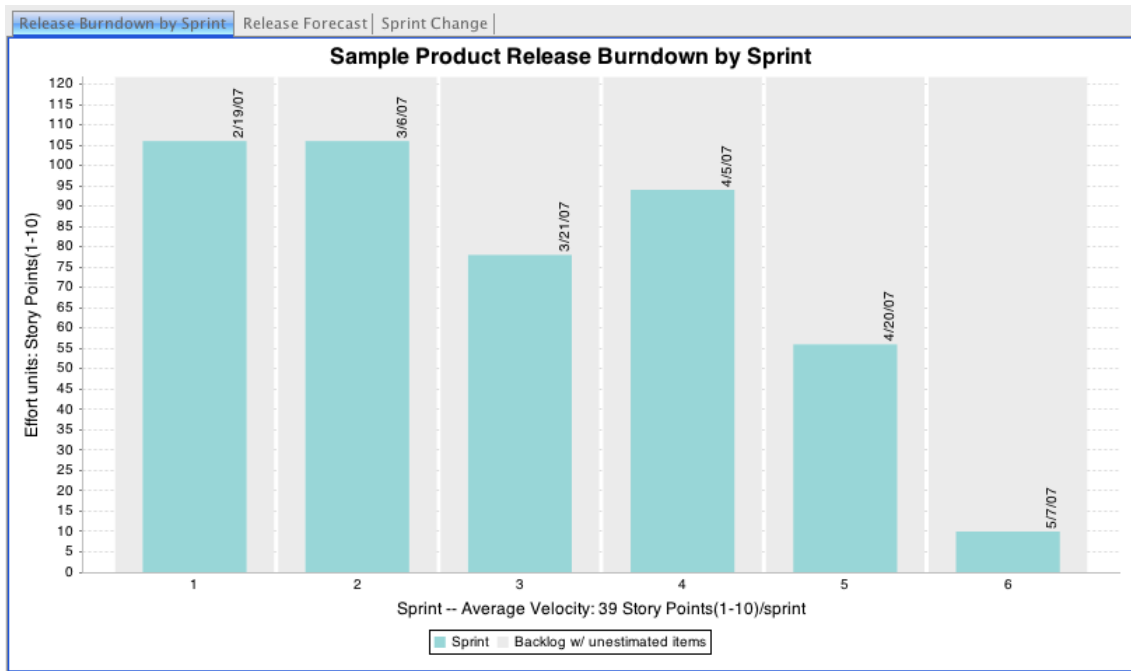


Figura 4-10: Velocidad del proyecto

4.7. TeamPulse

TeamPulse (<http://www.telerik.com/agile-project-management-tools.aspx>), herramienta creada por Telerik y que utiliza la tecnología Microsoft Silverlight, nos ofrece un conjunto de herramienta para la gestión de nuestros proyectos basados en metodologías ágiles.

TeamPulse nos ofrece diferentes gráficas para la comparación de diferentes iteraciones de un mismo producto. En la figura 3-11 podemos ver una gráfica muy completa que nos ofrece información sobre las iteraciones pasadas. En el eje X están las diferentes iteraciones y en el eje Y el número de puntos asociados a las 'stories'. Los puntos de las 'stories' son de dos tipos referente a características del producto y bugs. Además la gráfica nos ofrece dos líneas de tendencia que nos dan una cierta idea como pueden ser las siguientes iteraciones respecto a puntos por iteración y bugs por iteración. También nos ofrece en un recuadro en la parte superior derecha la cantidad media de puntos y bugs por iteración.

En la figura 3-12 podemos ver los puntos de las 'stories' planeadas por iteración respecto a los puntos de las 'stories' entregadas, en la gráfica ejemplo se ve que se ha entregado todo lo que se había planeado, pero es posible que en ocasiones alguna iteración tenga más planeadas que entregadas.

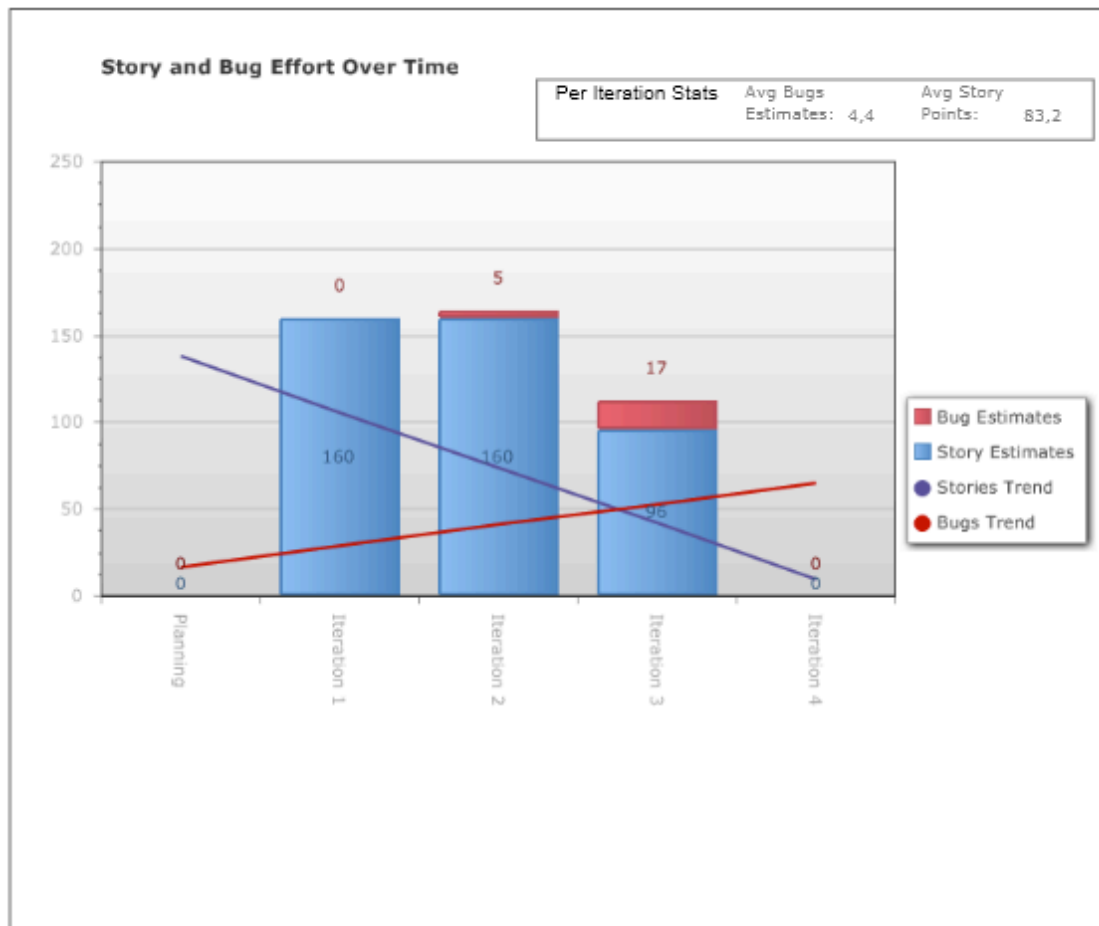


Figura 4-11: 'Stories' y bugs en cada iteración y tendencias

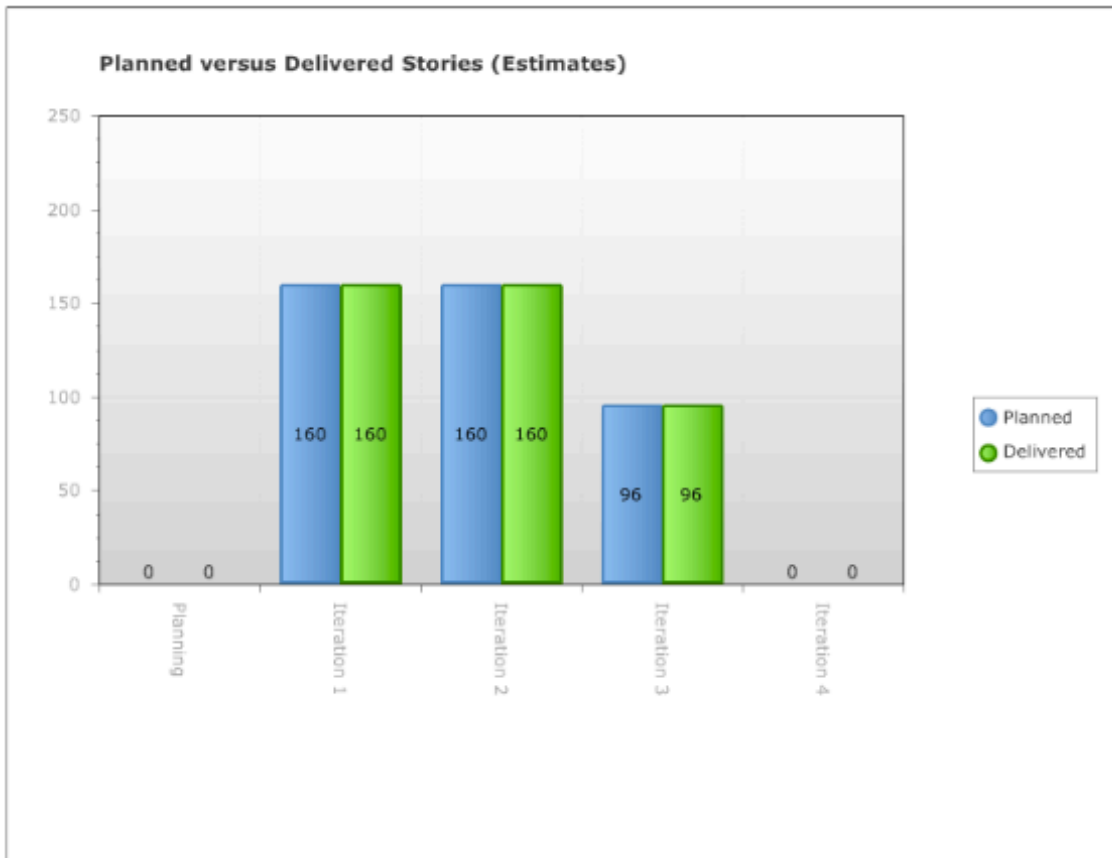


Figura 4-12: 'Stories' planeadas para una versión y las que al final se han entregado

5. Tune-Up

En este capítulo se presenta la metodología que apoya el proceso de desarrollo, y que guarda todos los datos que posteriormente explotaremos para mostrar la información relevante en el cuadro de mandos.

El material presentado en este capítulo ha sido extraído de las siguientes fuentes [4] y [5].

5.1. ¿Qué es TUNE-UP?

TUNE-UP es una metodología que incorpora aspectos ágiles y tradicionales con un sentido marcadamente pragmático. TUNE-UP se caracteriza fundamentalmente por combinar los siguientes elementos:

- **Modelo iterativo e incremental** para el desarrollo y mantenimiento del software. El trabajo se divide en unidades de trabajo (WUs) que son asignadas a versiones del producto. Se realizan ciclos cortos de desarrollo, entre 3 y 6 semanas, dependiendo del producto.
- **Workflows flexibles** para la coordinación del trabajo asociado a cada unidad de trabajo. Los productos, según sus características, tienen disponibles un conjunto de workflows los cuales se asignan a cada una de las unidades de trabajo. Cada unidad de trabajo sigue el flujo de actividades del workflow para completarla. Bajo ciertas condiciones se permite saltar hacia adelante o hacia atrás en el workflow, así como cambios de agentes asignados e incluso cambio de workflow. Por ejemplo, las típicas situaciones de re-trabajo en desarrollo de software ocasionadas por detección de defectos se abordan con saltos atrás no explícitos en el workflow.
- **Proceso de desarrollo dirigido por las pruebas de aceptación (Test-Driven)**. La definición de una unidad de trabajo es básicamente la especificación de sus pruebas de aceptación acordadas con el cliente. A partir de allí, todo el proceso gira en torno a ellas, se estima el esfuerzo de implementar, diseñar y aplicar dichas pruebas, se diseñan e implementan y luego se aplican sobre el producto para garantizar el éxito de la implementación.
- **Planificación y seguimiento continuo centrados en la gestión del tiempo**. En todo momento debe estar actualizado el estado de las versiones, de las unidades de trabajo, y del trabajo asignado a los agentes. El jefe del proyecto puede actuar oportunamente con dicha información, tomando decisiones tales como: redistribuir carga de trabajo entre agentes, cambiar los plazos de la versión, mover unidades de trabajo entre versiones, etc.
- **Control de tiempos**. Los agentes registran el tiempo que dedican a la realización de las actividades, el cual se compara con los tiempos estimados en cada una de ellas, detectando oportunamente desviaciones significativas. Esto permite a los agentes gestionar más efectivamente su tiempo, mejorar sus estimaciones y ofrecer al jefe del proyecto información actualizada del estado de la versión.

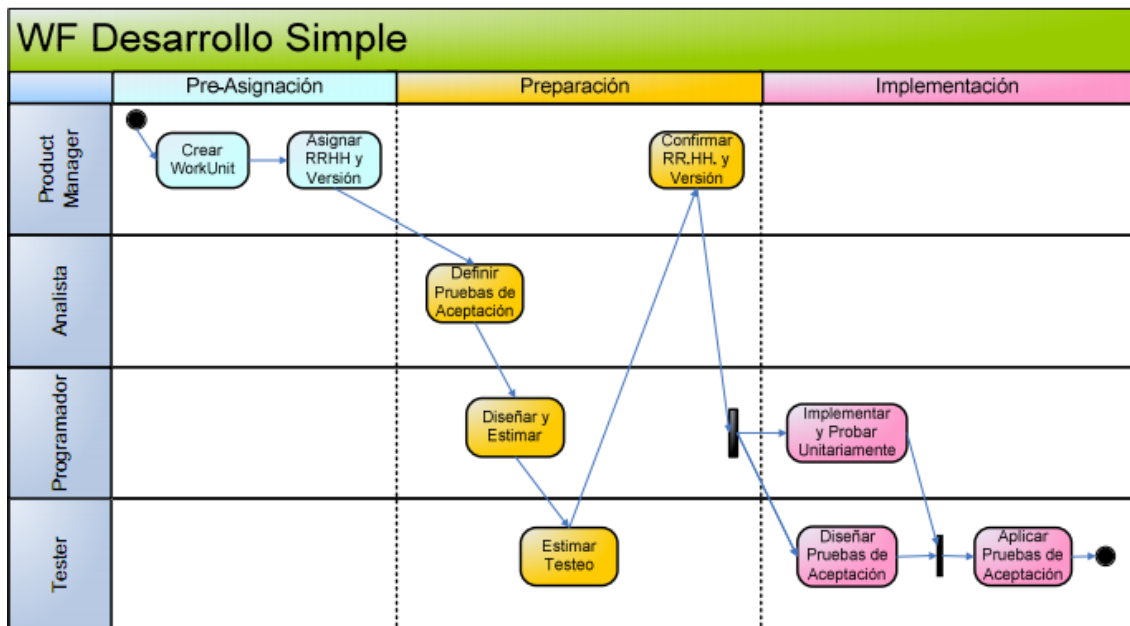


Figura 5-1: WF de desarrollo simple para unidades de trabajo

La Figura 4.1 ilustra un workflow simple para el desarrollo de una unidad de trabajo (WU). Un workflow, en general, incluye actividades asociadas a tres fases del proceso:

- **Pre-Asignación:** actividades realizadas hasta la asignación de los RRHH y versión. La unidad de trabajo ha sido identificada pero aún no tiene una prioridad como para asignarle los RRHH y versión.
- **Preparación:** se pueden realizar cuando la unidad de trabajo ha alcanzado cierta prioridad y se le han asignado los RRHH y una versión. Se comienza a trabajar en la preparación de la unidad de trabajo, y debería concluirse antes del inicio de la versión objetivo en la cual se implementará. Incluye el análisis, las revisiones y estimaciones para su implementación.
- **Implementación:** se realizan durante la versión objetivo. Incluye la implementación, aplicación de pruebas e implantación. Las actividades de cada workflow pueden variar significativamente dependiendo de factores tales como la cantidad y especialización de agentes participantes, validaciones o negociaciones predeterminadas con el cliente, etc. Cada unidad de trabajo en una iteración del proyecto podría tener su propio workflow. Sin embargo, en la práctica basta con disponer de un reducido conjunto de workflows que permitan cubrir los tipos de unidades de trabajo que se presentan en el proyecto.

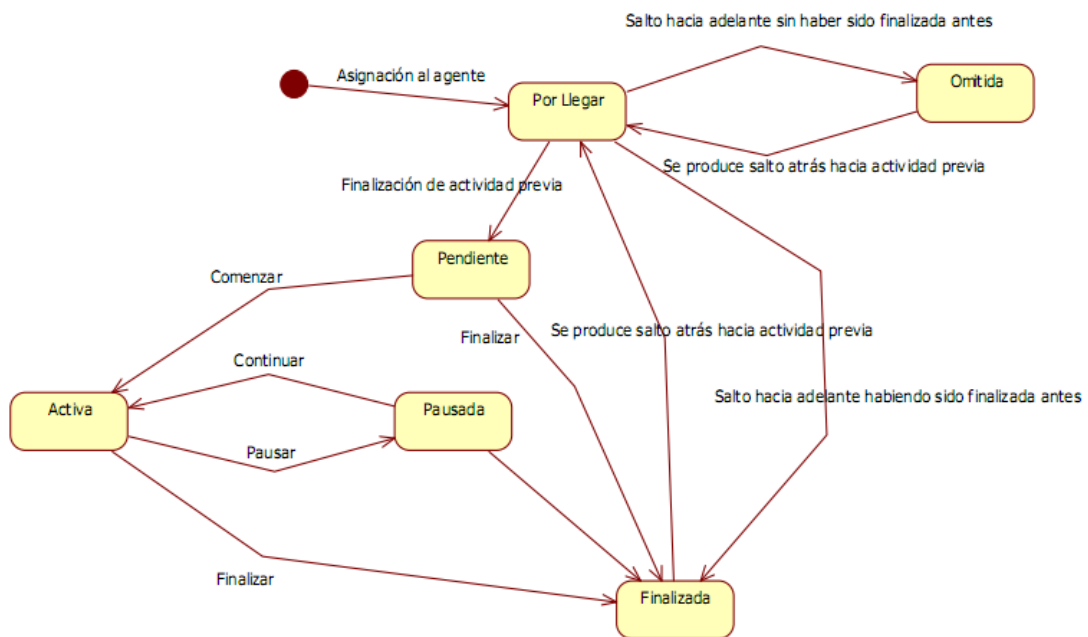


Figura 5-2: Posibles estados de una unidad de trabajo en una actividad

La Figura 4.2 muestra los posibles estados (Por Llegar, Pendiente, Activa, Pausada, Finalizada y Omitida) en los que se puede encontrar una actividad asociada a una unidad de trabajo.

5.2. TUNE-UP Process Tool

TUNE-UP Process Tool es una herramienta de apoyo para la aplicación efectiva de la metodología TUNE-UP. La herramienta está formada por tres módulos principales: Planificador Personal, Gestor de Unidades de Trabajo, Planificador de Versiones y Cuadro de Mandos. A continuación se describe cada uno de estos módulos.

5.2.1. Planificador Personal

El Planificador Personal (PP) presenta el trabajo que tiene un agente. Cuando un agente inicia su jornada laboral, accede al PP (Figura 4.3.) para ver el trabajo que tiene asignado actualmente, y seleccionar qué va a realizar de acuerdo a sus prioridades. El PP ofrece una variedad de facilidades de filtrado y ordenamiento de información, además de datos de tiempos, para que el agente pueda determinar su elección de acuerdo a sus prioridades. La tabla de la izquierda de la figura resume las contabilizaciones de las unidades de trabajo según la actividad y estado en el que se encuentran, y la tabla de la derecha, muestra información de dichas unidades de trabajo incluyendo: producto, versión, descripción de la unidad de trabajo, tiempos calculados, estado de la actividad actual dentro del workflow, etc.

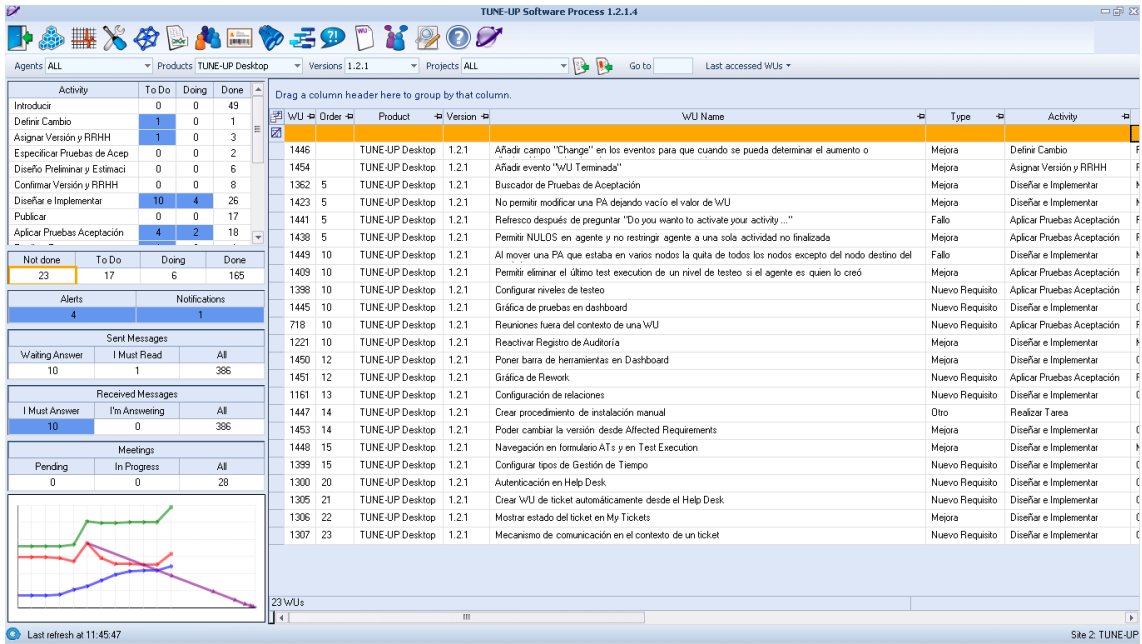


Figura 5-3: Planificador personal de TUNE-UP

5.2.2. Gestor de Unidades de Trabajo

Cuando el agente decide la unidad de trabajo que va a realizar, accede con ella al Gestor de Unidades de Trabajo (GUT) (Figura 4.4.) como apoyo esencial para realizar su tarea. En la parte superior se observan los datos generales de la unidad de trabajo, y en la parte inferior, un conjunto de pestañas con la funcionalidad que ofrece el GUT. Entre la funcionalidad disponible tenemos la posibilidad de activar/pausar/finalizar la actividad, mandar peticiones, añadir/consultar documentación asociada a la unidad de trabajo, ver los tiempos registrados en dicha unidad, etc.

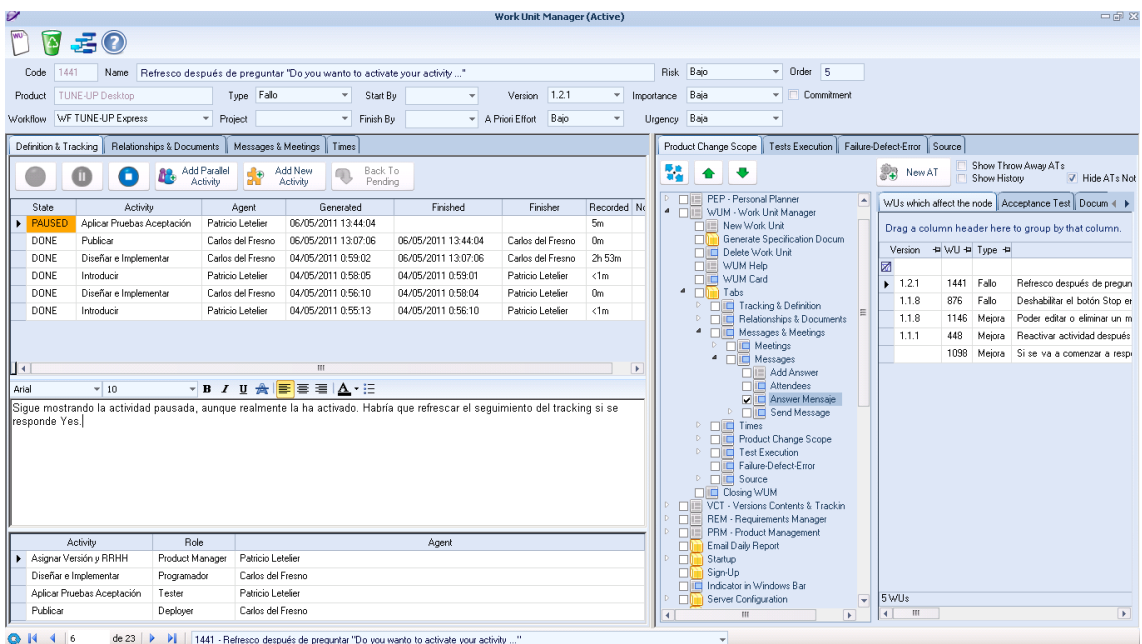


Figura 5-4: Gestor de unidades de trabajo

5.2.3. Planificador de Versiones

El Planificador de Versiones (PV) (Figura 4.5.) permite gestionar los productos, sus versiones, los workflows disponibles para cada producto, los agentes por defecto asignados a las actividades de los workflows y realizar el seguimiento continuo del estado actual de las versiones.

Programación	Testeo	WU	Order	Version	WU Name	Current Activity	Warnings	Type	Project	Importance	Urgency	A Priori Effort
		418		0.0.3	Tienda - Modificación y baja de artículo	Terminar / Jorge		Nuevo Requisito		Baja	Baja	Medio
		419		0.0.3	Tienda - Cesta	Terminar / Jorge		Nuevo Requisito		Alta	Baja	Medio
		424		0.0.3	Alta apuesta Campeón	Aplicar Pruebas de Aceptación / Sergio		Nuevo Requisito		Media	Alta	Alto
		426		0.0.3	Alta apuesta Especial	Terminar / Sergio		Nuevo Requisito		Media	Alta	Alto
		427		0.0.3	Modificación de apuesta Campeón	Terminar / Sergio		Nuevo Requisito		Media	Media	Alto
		429		0.0.3	Modificación de apuesta Especial	Aplicar Pruebas de Aceptación / María		Nuevo Requisito		Media	Media	Alto
		430		0.0.3	Realizar apuesta Campeón	Aplicar Pruebas de Aceptación / Pablo		Nuevo Requisito		Media	Alta	Alto
		435		0.0.3	Simulación de apuesta Campeón	Diseñar e Implementar / Miguel		Nuevo Requisito		Media	Alta	Medio
		436		0.0.3	Simulación de apuesta Especial	Terminar / Sergio		Nuevo Requisito		Media	Alta	Medio
		437		0.0.3	Simulación de apuesta Head to Head	Terminar / Sergio		Nuevo Requisito		Media	Alta	Medio
		605		0.0.3	Marcar apuestas destacadas	Terminar / José		Mejora		Muy Alta	Media	Medio
		608		0.0.3	Mis eventos	Introducir / Jorge		Nuevo Requisito		Alta	Media	Alto
		619		0.0.3	Perfil de usuario - Saldo en cuenta	Aplicar Pruebas de Aceptación / Pablo		Mejora		Alta	Alta	Bajo
		620		0.0.3	Historial de apuestas administrador	Terminar / Jorge		Mejora		Media	Media	Medio
		624		0.0.3	Enviar avisos	Terminar / Jorge		Mejora		Alta	Baja	Medio
		629		0.0.3	Modificación y baja de noticia	Aplicar Pruebas de Aceptación / Sergio		Mejora		Media	Media	Bajo
		630		0.0.3	Modificación de apuesta Head to Head	Terminar / Jorge		Mejora		Media	Media	Medio
		663		0.0.3	Mis eventos 2	Terminar / Jorge		Nuevo Requisito		Alta	Media	Alto
		664		0.0.3	PA Regresión A	Terminar / Jorge		Otros				
		665		0.0.3	PA Regresión B	Terminar / Jorge		Otros				
		666		0.0.3	PA Regresión C	Terminar / Jorge		Otros				

Figura 5-5: Planificador de versiones

5.2.4. Cuadro de Mandos en TUNE-UP

En TUNE-UP podemos encontrar un Cuadro de Mandos (ver Figura 4-6), en el panel superior izquierdo está la Gráfica Burn Down (en la cual nos centraremos en el siguiente párrafo). En el panel inferior izquierdo se muestra día a día la cantidad de esfuerzo asociado a WUs terminadas y a WUs no terminadas. En el panel superior derecho se muestra el esfuerzo invertido en cada WU, el cual se clasifica en tiempo ordinario, tiempo dedicado a responder mensajes y tiempo asociado a participación en reuniones. Además, en una pestaña del panel superior derecho puede también consultarse el esfuerzo asociado a retrabajo en cada WU de la iteración. Finalmente en el panel inferior derecho puede consultarse la cantidad de Pruebas de Aceptación y el estado de ellas en cada WU de la iteración.



Figura 5-6: Cuadro de Mandos de una iteración

TUNE-UP ofrece una Gráfica Burn Down para cada iteración de un producto, junto con una tabla llamada Daily Events con información complementaria para la correcta interpretación de la gráfica (esta tabla se explica en detalle más adelante).

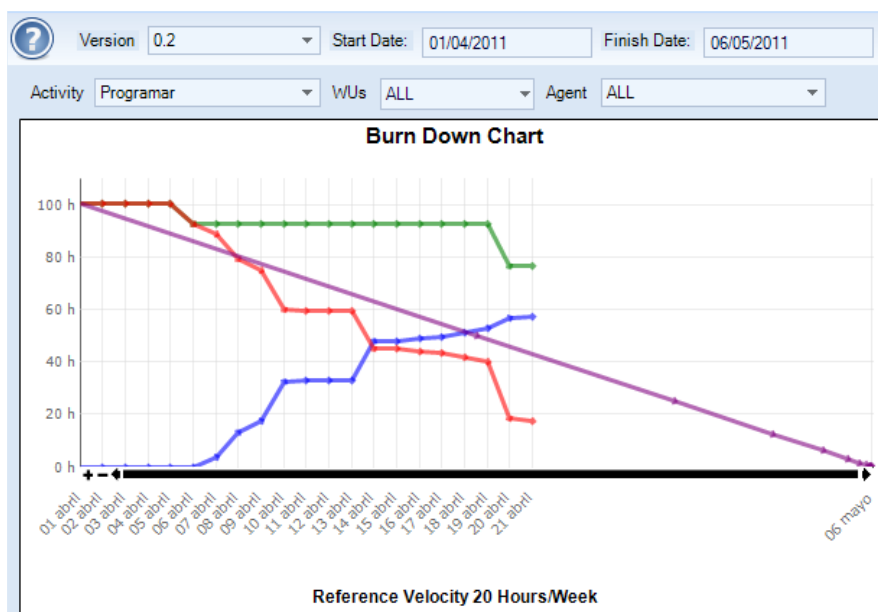


Figura 5-7: Gráfica Burn Down de una iteración

La Gráfica Burn Down de TUNE-UP (ver Figura 4-6) incluye dos gráficas en una; una gráfica básica Burn Down (asociada a la línea serpenteante descendente) y una gráfica Burn Up (asociada a la línea ascendente representando el esfuerzo invertido). Además, se incluye una línea que representa el esfuerzo estimado y una línea diagonal que representa el esfuerzo restante de referencia. Gracias a disponer en una misma gráfica de los esfuerzos estimados, invertidos y restantes, se facilita la interpretación del estado de la iteración y cómo se ha ido desarrollando. Situaciones tales como una bajada o subida pronunciada del esfuerzo restante podrían visualmente explicarse por una correspondiente bajada o subida en la línea de

esfuerzo estimado, o bien en una subida o bajada en la línea de esfuerzo invertido. Sin embargo, la confirmación de estas interpretaciones, como veremos a continuación, exige contar con la información detallada de los eventos que pueden haber ocurrido entre dos puntos consecutivos de la gráfica. El esfuerzo restante de referencia corresponde a la línea que se traza desde el punto de mayor esfuerzo restante hacia el punto de esfuerzo restante 0 en el día de fin de la iteración. Además, asociada a esta línea se muestra en la parte inferior de la gráfica la velocidad requerida para conseguir la tendencia ilustrada por ese esfuerzo restante de referencia. Toda la información de la Gráfica Burn Down puede ser filtrada por Actividad, WU y/o Miembro del equipo.

Date : 20 abril (2 items)		
Event Type : Ajuste Estimación - Decremento (3 items)		
Activity	WU	Name
Programar	31	Unirse a peña
Programar	41	Creacion galeria fotografica ruta
Programar	80	Ofrecer el máximo disponible día ocupado.
Event Type : Introducción de Estimación Faltante (2 items)		
Activity	WU	Name
Programar	85	Mostrar usuario que inserta en la foto en la galeria
Programar	86	Mostrar el numero de comentario totales en cada foto de la galeria
Date : 19 abril (1 item)		
Event Type : Actividad Sin Estimación (2 items)		
Activity	WU	Name
Programar	85	Mostrar usuario que inserta en la foto en la galeria
Programar	86	Mostrar el numero de comentario totales en cada foto de la galeria
+ Date : 18 abril (1 item)		
+ Date : 17 abril (1 item)		

Figura 5-8: Tabla Daily Events

La Tabla Daily Events que muestra la Figura 4-7, contiene los eventos diarios que permiten interpretar con precisión la Gráfica Burn Down. Haciendo clic en un punto de la gráfica de la Figura 4-6, se despliega en la tabla de la Figura 4-7 la lista de eventos ocurridos entre el día previo y el día seleccionado. Para cada evento se indica el miembro del equipo, la actividad e información de la WU donde se produce. En TUNE-UP se supervisan todos los eventos que pueden influir en la correcta interpretación del esfuerzo restante, dichos eventos se describen a continuación:

Eventos que invalidan la lectura del esfuerzo restante

- Actividad con estimación sobrepasada. El esfuerzo invertido por el miembro del equipo en la actividad sobrepasa el estimado (lo cual llevaría a un esfuerzo restante negativo). El miembro del equipo asignado debería re-estimar
- Actividad sin estimación. La actividad no está estimada o su valor es 0. El miembro del equipo asignado debería estimar

Eventos que provocan una variación en el esfuerzo restante observado

- Cambios del esfuerzo invertido. El esfuerzo invertido se ha modificado. Por ejemplo, se había registrado 10 horas de trabajo cuando realmente debían de ser 5 horas.
- Ajuste en Estimación - Incremento/Decremento.
- Introducción de estimación faltante. Indica que se ha estimado una actividad que el día anterior no tenía estimación.

- Actividad asignada/desasignada a/de un miembro del equipo. Esto es sólo relevante cuando se trata de la Gráfica Burn Down filtrada con un miembro del equipo.
- WU nueva. WU creada y añadida a la iteración
- WU eliminada.
- WU desestimada. Su esfuerzo restante se considera igual a 0
- WU añadida. WU que se ha añadido a la iteración (ya existía sin iteración asignada o en otra iteración).
- WU quitada. WU que estaba el día anterior en la iteración pero se ha cambiado a otra.
- WU terminada. Su esfuerzo restante se considera igual a 0

Para ilustrar el uso de estos eventos en la interpretación de la gráfica, a continuación comentamos un ejemplo. La Figura 4-6 muestra la Gráfica Burn Down correspondiente a la actividad Programación de la versión 0.2 de un determinado producto. En la gráfica, el día 20 de Abril se observa un descenso pronunciado del esfuerzo restante. Este descenso, a priori lo podemos asociar al descenso del esfuerzo estimado (observable en la línea superior de la gráfica). Pero cuestiones tales como: ¿por qué ha descendido el esfuerzo estimado?, ¿se ha movido, eliminado o desestimado trabajo?, ¿algún miembro del equipo ha ajustado alguna estimación?, no se pueden responder a simple vista. Para responder tales cuestiones es esencial la información de la tabla *Daily Events*. En la Figura 4-7 vemos los eventos asociados a la actividad Programación que ocurrieron el día 20 de Abril; hubo un decremento en la estimación de la actividad Programación en 3 WUs y se han introducido 2 nuevas estimaciones que faltaban (en la Figura 4-7 vemos que el día 19 de Abril faltaba por estimar la actividad Programación en dichas WUs). De esta forma sabemos que el descenso del esfuerzo restante se debe a un decremento en la estimación de 3 WUs, aunque además se hayan incluido 2 nuevas estimaciones antes no consideradas.

Finalmente, el disponer de información detallada de lo realizado durante cada iteración, aporta información útil para reuniones de revisión de la iteración o reuniones de retrospectiva, pudiendo llegar a evaluar acciones de mejora en el proceso mediante la comparación de datos de diferentes iteraciones y su tendencia.

6. Creación del cuadro de mandos

6.1. Introducción

Después de hacer el repaso de la herramienta que usaremos para crear nuestro proyecto, Microsoft Report Builder, las diferentes opciones de cuadros de mandos para el desarrollo del software que existen actualmente y TUNE-UP, ahora es momento de explicar como vamos a enfocar el desarrollo de nuestro cuadro de mandos alrededor de la información que almacena TUNE-UP en su base de datos.

TUNE-UP como herramienta, gestiona todos los datos anteriormente visto sobre un servidor Microsoft SQL Server, por tanto para la creación de las gráficas hemos ido directamente a la fuente, creando nuestras propias consultas para elegir la información que creemos relevante. Como veremos más adelante hemos incluido dichas consultas y sobre que tablas de TUNE-UP trabajamos.

Hemos creado todas las gráficas en un mismo informe para así tener de un vistazo toda la información de la cual disponemos.

Nuestro cuadro de mandos se basará en 3 gráficos que creemos relevantes para el desarrollo del software basado en la metodología TUNE-UP, que pasaremos a describir en los siguientes apartados, pero primero veamos lo que tienen en común.

El cuadro de mandos tiene dos parámetros que el usuario puede seleccionar: uno para seleccionar el producto del cual queremos mostrar información y después para seleccionar las versiones de las cuales queremos información (figura 5-1). Hemos creado un dataset por cada gráfica, más dos datasets para los parámetros.

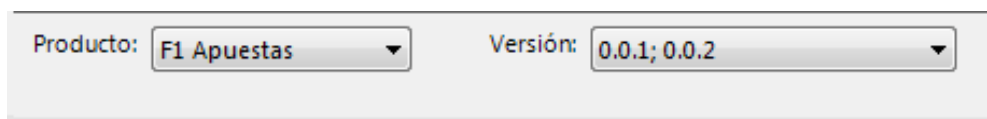


Figura 6-1: Parámetros del cuadro de mandos

El primer gráfico es el porcentaje de unidades de trabajo (WUs) por versión según su tipo, pudiendo ver en que versiones se han hecho WUs de nuevos requisitos o versiones que sólo han servido para reparar los fallos.

En segundo lugar tenemos el número de eventos por versión, así podremos ver los diferentes eventos producidos, si se han desestimado WUs, si se ha sobrepasado el tiempo estimado en alguna WU, etc.

El último será la velocidad del proyecto por versión, métrica que hemos definido como el tiempo registrado de todas las WUs de una versión partido las semanas que ha durado esa versión. Es un gráfico que nos ayuda a asimilar como de rápido hemos sacada una versión pudiendo hacer una estimación para futuras versiones.

Lo que ahora veremos es un repaso más profundo a las gráficas que componen nuestro cuadro de mandos y que hemos descrito anteriormente, añadiendo información técnica (tablas usadas y consultas realizadas). Para todos los gráficos hemos usado una base de datos con dos

proyectos (F1 Apuestas y Web Recetas) con diferentes versiones, estos constituyen nuestros datos de prueba.

6.2. Porcentaje de WUs por versión

6.2.1. Explicación

En este primer gráfico (figura 5-2), nos centramos en las WUs que hemos trabajado en cada versión de nuestro producto. Las WUs son unidades de trabajo asociadas a un proyecto concreto, ej. 'Crear formulario de entrada', y que se pueden agrupar por diferentes categorías como Nuevo Requisito, Fallo, Mejora, etc.

La finalidad de esta gráfica es tener una visión representativa de que manera las versiones se han enfocado, por ejemplo en una gráfica se puede ver que en la versión 0.3 hay un porcentaje considerable de WUs que son de la categoría Error, por lo tanto se puede deducir que ha sido una versión enfocada a resolver los errores de versiones anteriores y que ha sido de mantenimiento.

La gráfica es de barra de columnas, donde cada columna representa una versión del producto elegido en el parámetro. Las WUs se acumulan en porcentajes según los tipos que hemos comentado anteriormente.

En la figura 5-2 podemos ver como la primera versión 0.0.1 se trata de WUs relacionadas con nuevos requisitos que queremos para nuestra aplicación, viendo en futuras versiones como el porcentaje de WUs Nuevo Requisito va decayendo, dando más importancia a WUs de Mejora o clasificadas como Otros.

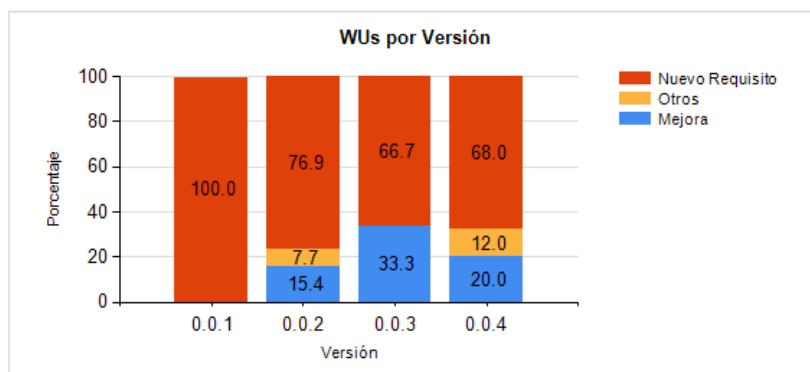


Figura 6-2: Número de WUs por versión

Creemos que esta gráfica puede ser un buen punto de inicio para determinar como las versiones han sido trabajadas y realmente si ha sido productivo (quizás respecto a nuevos requisitos) la versión que queremos estudiar.

6.2.2. Parte técnica

Para la creación de esta gráfica hemos usado las siguientes tablas que se encuentran en la base de datos de TUNE-UP: Producto, Version, UT y TipoUT.

En Microsoft Report Builder hemos creado un dataset llamado WUVersion que incluye todos los campos que necesitamos para presentarlo en forma de gráfica. La consulta es la siguiente:

SELECT

Producto.Nombre AS [Producto Nombre], Version.Nombre AS [Version Nombre], UT.IdUT, Version.IdVersion, Version.IdProducto, Producto.Activo, TipoUT.IdTipoUT AS [TipoUT IdTipoUT], TipoUT.Nombre AS [TipoUT Nombre], UT.IdTipoUT AS [UT IdTipoUT]

FROM

UT INNER JOIN TipoUT ON UT.IdTipoUT = TipoUT.IdTipoUT

INNER JOIN Producto ON UT.IdProducto = Producto.IdProducto

INNER JOIN Version ON Producto.IdProducto = Version.IdProducto AND UT.IdVersion = Version.IdVersion

GROUP BY

Producto.Nombre, Version.Nombre, UT.IdUT, Version.IdVersion, Version.IdProducto, Producto.Activo, TipoUT.IdTipoUT, TipoUT.Nombre, UT.IdTipoUT

Como podemos ver es una consulta bastante directa donde nos traemos la información indispensable para la creación de la gráfica anterior.

6.3. Número de eventos por versión

6.3.1. Explicación

En esta gráfica (figura 5-3) podemos ver los eventos que han sucedido en una versión. Los eventos pueden ser del tipo: WU Nueva, WU Eliminada, etc. que nos da una idea de las cosas que han sucedido en una versión.

La finalidad de esta gráfica es apoyarse en la visualización de estos datos para explicar ciertas tendencias que se puedan ver en otra gráficas, podemos ponernos en el caso que haya una versión del producto que veamos en su BurnDown chart que ha adelantado mucho trabajo y que ha bajado drásticamente, gracias a esta gráfica podemos ver que se ha debido a que se han eliminado WUs por ejemplo.

La gráfica está compuesta por barras de columnas por cada tipo de evento, pudiendo comparar los eventos que se dan más a lo largo de las versiones, en el caso de nuestro ejemplo podemos ver como el evento WU Añadida a la Versión supera en cuantía los otros eventos, pudiendo establecer patrones porque ocurre eso y comprender ciertos aspectos del desarrollo del producto que estamos trabajando actualmente.

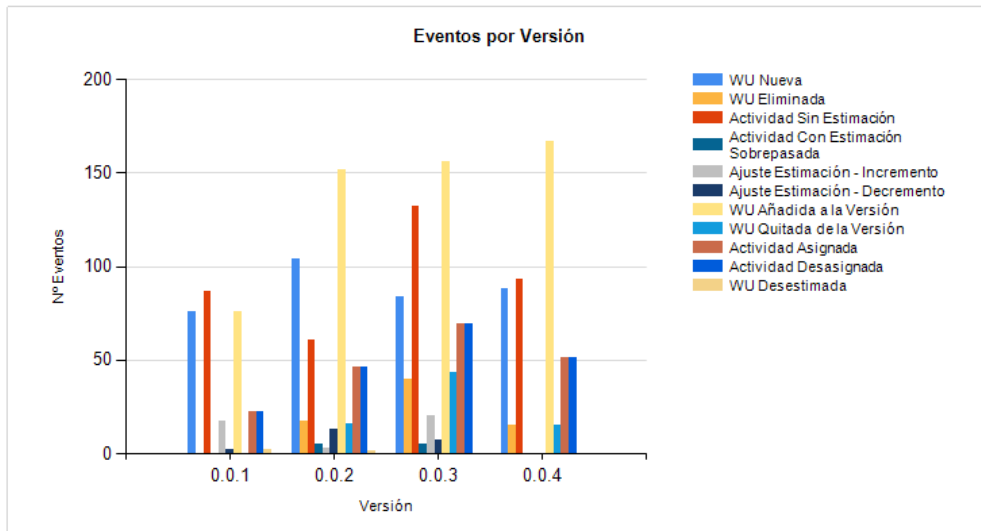


Figura 6-3: Número de eventos por versión

6.3.2. Parte técnica

En esta gráfica hemos trabajado con las siguientes tablas de TUNE-UP: BurnDownChartEventosDiarios, BurnDownChartTipoEvento y Versión.

En Microsoft Report Builder hemos creado un dataset llamado EventoVersion que incluye los campos que usaremos para al confección de la gráfica. La consulta ha sido la siguiente:

SELECT

Version.Nombre AS [Version Nombre], BurnDownChartTipoEvento.Nombre AS [BurnDownChartTipoEvento Nombre], BurnDownChartEventosDiarios.IdProducto, BurnDownChartEventosDiarios.IdVersion AS [BurnDownChartEventosDiarios IdVersion], BurnDownChartEventosDiarios.Fecha, BurnDownChartTipoEvento.IdTipoEvento, Version.IdVersion AS [Version IdVersion], BurnDownChartEventosDiarios.IdUT

FROM

BurnDownChartTipoEvento

INNER JOIN *BurnDownChartEventosDiarios* **ON** *BurnDownChartTipoEvento.IdTipoEvento=BurnDownChartEventosDiarios.IdTipoEvento*

INNER JOIN *Version* **ON** *BurnDownChartEventosDiarios.IdVersion = Version.IdVersion*

WHERE

BurnDownChartEventosDiarios.IdVersion IS NOT NULL

GROUP BY

*Version.Nombre, BurnDownChartTipoEvento.Nombre,
BurnDownChartEventosDiarios.IdProducto,
BurnDownChartEventosDiarios.IdVersion,
BurnDownChartEventosDiarios.Fecha, BurnDownChartTipoEvento.IdTipoEvent,
Version.IdVersion, BurnDownChartEventosDiarios.IdUT*

6.4. Velocidad de la versión

6.4.1. Explicación

Esta última gráfica (figura 5-4) que compone nuestro cuadro de mandos está basada en una métrica definida por nosotros como:

$$\frac{\text{Tiempo empleado en la versión (horas)}}{\text{Semanas}}$$

dando un resultado de horas/semanas. El tiempo empleado en la versión se calcula sumando todas las duraciones de las actividades que se han realizado en esa versión.

Creemos que es importante este tipo de métricas ya que a parte de describirnos el estado pasado y actual del proyecto nos permite hacer estimaciones de las velocidades de siguientes versiones.

Como podemos ver en la gráfica de ejemplo vemos como la velocidad de las versiones han ido bajando, ya sea porque la primera versión es la que más trabajo representa. Como todas las gráficas nos hemos de apoyar con más datos o gráficas para realmente saber lo que está pasando en cada momento, aunque esta gráfica nos da un buen punto de partida para describir ciertas tendencias a la hora de sacar versiones.

Para esta gráfica nos hemos decidido por uno de líneas ya que puede describir mejor el cambio de velocidad entre las diferentes versiones. En cada punto vemos exactamente el valor de la velocidad de dicha versión.

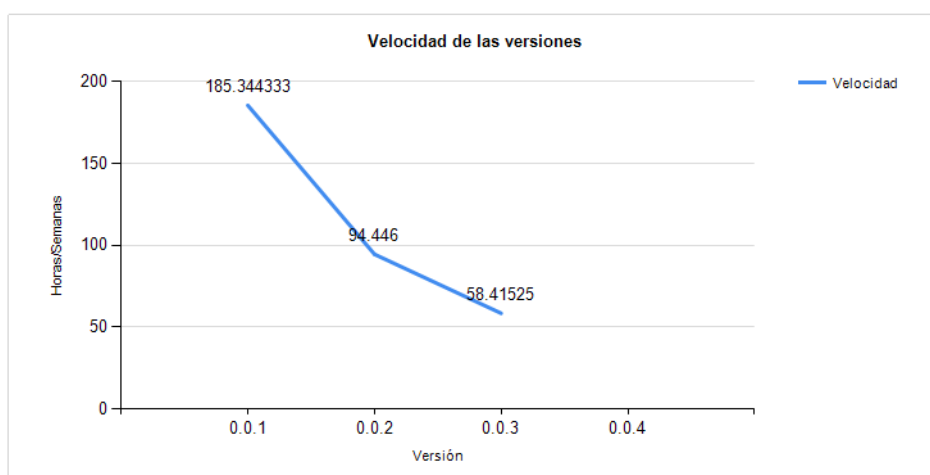


Figura 6-4: Velocidad de la versión

6.4.2. Parte técnica

Para esta gráfica hemos usado las siguientes tablas de TUNE-UP: BurnDownChartSnapshot y Versión.

En Microsoft Report Builder hemos creado un dataset llamado VelocidadVersion con la siguiente consulta:

```
SELECT

BurnDownChartSnapshot.IdProducto AS [BurnDownChartSnapshot IdProducto],
Version.IdVersion AS [Version IdVersion], Version.Nombre,
Version.FechaEstimadaInicio, Version.FechaEstimadaFin,
Version.FechaRealInicio, Version.FechaRealFin, Version.IdProducto AS [Version
IdProducto], BurnDownChartSnapshot.IdVersion AS [BurnDownChartSnapshot
IdVersion], SUM(DISTINCT BurnDownChartSnapshot.[Real])/DATEDIFF ( wk ,
isnull(Version.FechaRealInicio,Version.FechaEstimadaInicio) ,
isnull(Version.FechaRealFin,Version.FechaEstimadaFin)) AS [Sum Distinct_Real]

FROM

BurnDownChartSnapshot

INNER JOIN Version ON BurnDownChartSnapshot.IdVersion = Version.IdVersion

WHERE

BurnDownChartSnapshot.[Real] IS NOT NULL

GROUP BY

BurnDownChartSnapshot.IdProducto, Version.IdVersion, Version.Nombre,
Version.FechaEstimadaInicio, Version.FechaEstimadaFin,
Version.FechaRealInicio, Version.FechaRealFin, Version.IdProducto,
BurnDownChartSnapshot.IdVersion
```

Un dato que hemos tenido en cuenta es, si en la tabla versión la columna FechaRealInicio es null, cogemos la FechaEstimadaInicio, esto también lo aplicamos para la fecha fin del proyecto.

7. Conclusiones

Actualmente los cuadros de mandos integrales están siendo un gran apoyo para la industria del software, permitiendo manejar grandes volúmenes de información con respecto al desarrollo para ofrecernos gráficas y métricas que nos ayuden a comprender y mejorar el proceso de desarrollo. Como hemos podido ver hay una gran cantidad de herramientas que nos ofrecen gráficas y métricas respecto al proceso de desarrollo de software pero también hemos podido ver que cada producto utiliza sus propias gráficas y no hay ninguna gráfica estándar que sea común a todos los productos, en nuestro caso también hemos definido nuestras propias gráficas y métricas y hemos hecho un proyecto a medida. Creemos que los cuadros de mandos son buenos indicadores del estado de un proyecto y que irán adquiriendo más importancia en el campo de la ingeniería del software.

En cuanto a los objetivos de este proyecto, podemos afirmar lo siguiente:

- Hemos hecho un repaso al concepto de Cuadro de Mandos, repasando su historia y evaluando las ventajas y inconvenientes de su implantación.
- Se ha hecho un repaso de la herramienta utilizada para la creación de nuestro propio cuadro de mandos. Creemos que es importante ofrecer un trasfondo técnico de la herramienta en la que nos hemos apoyado completamente para la creación del cuadro de mandos ya que nos ofrece una visión de cómo hemos podido enfocar su creación.
- Hemos hecho una búsqueda sobre productos que implementan los cuadros de mandos más interesantes actualmente y que acotaban el mostrar información entre versiones como el nuestro.
- Se ha presentado la metodología TUNE-UP, poniendo hincapié en la utilidad de su método y como ya han aplicado los conceptos de cuadro de mandos en el programa.
- La creación de nuestro cuadro de mandos se rige a las características que nos habíamos propuesto al principio del proyecto, básico, útil y muestre información que ayude al proceso de desarrollo de software. Creemos que este cuadro de mandos es un buen punto de partida y que es fácilmente ampliable con toda la información que hay en la base de datos de TUNE-UP.

Respecto a la experiencia de la creación de un cuadro de mandos podemos determinar lo siguiente:

- Microsoft Report Builder 3.0 es una herramienta básica para hacer informes y que todavía le queda bastante para ser considerada una solución respecto a otros software como Crystal Reports. Su gran ventaja es la integración de los informes creados en entornos Sharepoint y la facilidad de uso.
- Aunque el desarrollo del cuadro de mandos ha sido básico, ha sido esencial la selección de los campos de las tablas que queríamos para mantener las consultas simples y sencillas, y que nos dieran todo lo que necesitábamos para crear las gráficas. El cuadro de mandos se ejecuta rápidamente y creemos que eso es una necesidad.
- La falta de un estándar que guíe que parámetros son realmente importantes para la creación de un cuadro de mandos resulta en que es un producto enfocado especialmente para TUNE-UP no pudiendo determinar si en otras metodologías o productos podría funcionar un cuadro de mandos como el que hemos hecho.

Personalmente el proyecto me ha dado una perspectiva sobre lo importante que es tener información del producto software que se está creando de una manera visual y directa, y como de acertada es el enfoque de los cuadros de mandos. Para entornos grandes es más que necesario la pronta implantación de estos sistemas, ya que permiten aplicar medidas si se ve que no se está siguiendo las directrices marcadas al principio del desarrollo. TUNE-UP ofrece ahora un cuadro de mandos integrado en la herramienta que creemos que es relevante y necesario para nuevos proyectos.

También creo que un estándar para este tipo de cuadro de mandos sería una más que buena idea pudiendo entender de un vistazo la forma global del producto, aunque entiendo que cada tipo de proyecto es diferente y puede necesitar otro tipo de información. Un estándar que sea un mínimo para que los productos lo implementen y después puedan añadir más información que creen necesaria dependiendo de la metodología escogida.

Referencias

- [1] Kaplan, Robert S.; Norton, David P. *The Balanced Scorecard*. Harvard Business School Press (Boston), 1992
- [2] Microsoft MSDN. *Report Builder 3.0*. Obtenido en <http://msdn.microsoft.com/en-us/library/dd207008.aspx>
- [3] Fábricas de software. *Herramientas para el desarrollo ágil*. Obtenido en <http://www.fabricasdesoftware.es/herramientas>
- [4] Marante Estellés, María Isabel. *Planificación y seguimiento en proyectos de desarrollo y mantenimiento de software dirigido por la gestión de tiempos*. Diciembre 2009.
- [5] María Isabel Marante, Maria Company y Patricio Letelier. *Actas de las Jornadas de Ingeniería de Software JISBD 2011, A Coruña*. Septiembre de 2011.