UNIVERSITAT POLITÈCNICA DE VALÈNCIA

DSIC
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Desarrollo de un módulo para la gestión gráfica de workflows para procesos software

Autor: Javier Casal Ruiz

Director: Dr. Patricio Letelier Torres

Diciembre del 2009

Tesis presentada para cumplir con los requisitos finales para la obtención del título de Máster en Ingeniería del Software, Métodos Formales y Sistemas de Información, de la Universidad Politécnica de Valencia, 2009.

Agradecimientos

Mi más sentido agradecimiento a Patricio Letelier por su dirección en el presente trabajo, la cual ha sido fundamental, y a mis compañeros de la empresa colaboradora, especialmente a Mabel Marante y Carlos Del Fresno por la inestimable ayuda prestada.

A la gente que me ha acompañado en el camino, a mi familia y en especial a Marina. Gracias por la confianza depositada.

Índice

Capítulo 1 Introducción y motivación

1.1 – Introducción 8
1.2 – Motivación 9
1.3 - Estructura 10

Capítulo 2 Workflows

2.1 – Introducción 13
2.1 – Workflow 14
2.3 – BPMN 16
2.3.1 – Introducción 16
2.3.2 – Fundamentos 17
Objetos de flujo 18
Objetos conectores 19
Swimlanes 21
Artefactos 23
2.3.3 – Uso general de BPMN 25
Procesos B2B colaborativos 25
Procesos de negocio internos 26
2.3.4 – Diferentes niveles de precisión 26
2.3.5 – Valor de modelar en BPMN 28
2.3.6 – Mapear un diagrama BPMN a BPEL4WS 29
2.3.7 – Futuro de BPMN 30
2.4 – Herramientas 31
2.4.1 - Lotus Notes 31
2.4.2 - IBM WebSphere Business Monitor 31
2.4.3 - Microsoft BizTalk 32
2.4.4 - K2.NET 33
2.4.5 - Adobe LiveCycle Workflow 34
2.4.6 - Auraportal BPM 34
2.4.7 - DynaFlow EZ-Process 35
2.4.8 - Fujitsu – Interstage BPM 36
2.4.9 - Oracle Fusion Middleware 38
2.4.10 - Ultimus BPM Suite 38
2.4.11 – Comentarios finales 39

Capítulo 3 Workflows para el proceso de desarrollo de software

3.1 – Introducción 41
3.2 – Introducción a TuneUp 42
3.3 – TuneUp Process Tool 43
3.3.1 – Planificador personal 43
3.3.2 –Gestor de Unidades de Trabajo 44
3.3.3 – Planificador de Versiones 46
3.3 – Gestión de workflows 47

Capítulo 4 Librerías para desarrollo de editores gráficos

4.1 – Comparativa de componentes de diagramas 50
4.2 – Componentes a comparar 50
4.2.1 – AddFlow for .NET 51
4.2.2 – Syncfusion Essential Diagram 53
4.2.3 – Nevron Diagram 57
4.2.4 – GoDiagram 60
4.2.5 – Netron 62
4.2.6 – Open Diagram 63
4.2.7 – yFiles 64
4.3 – Conclusiones 66

Capítulo 5 Módulo para especificación de Workflows

5.1 - Especificación de requisitos 69
5.1.1 – Propósito 69
5.1.2 – Descripción del sistema actual 69
5.2 – Arquitectura propuesta 72
5.2.1 – Casos de uso 72
Visor 73
Editor 75
5.3 – Diseño 79
5.3.1 – Introducción 79
5.3.2 – Diagrama de clases 79
5.3.3 – Arquitectura de la aplicación 81
Cargar workflow sin diagrama 82
Publicar workflows 87
Estadísticas de la implementación 89
5.4 – Diseño de las pruebas de aceptación 90
5.4.1 –Visor 90
5.4.2 –Editor 97

Capítulo 6 Conclusiones

6.1 – Conclusiones y trabajo futuro 123

Anexo – Referencias 126

Capítulo 1: Introducción

1.1 Introducción

Como respuesta a los problemas surgidos por el desarrollo de software nacen distintas metodologías y estándares que ayudan a abordar un proyecto de manera que el proceso sea predecible y repetible. Dentro de las metodologías asociadas al proceso de desarrollo de software, TUNE-UP (Marante y Letelier, 2009) [25] surge como una metodología enfocada al trabajo en equipos pequeños que integra distintos aspectos de las metodologías ágiles y de las metodologías tradicionales. Las características fundamentales de TUNE-UP son:

- El uso de un modelo iterativo e incremental para el desarrollo y mantenimiento de software.
- El proceso de desarrollo está dirigido por pruebas de aceptación.
- La coordinación del trabajo entre agentes depende de workflows flexibles.
- Planificación y el seguimiento continuo centrados en la gestión del tiempo.
- Control de tiempos.

Las dos primeras características clasifican a TUNE-UP como metodología ágil, sin embargo, las otras características están más próximas de lo que sería una metodología tradicional.

TUNE-UP lleva 4 años implantada en la PYME en la que se realiza este trabajo de tesis, en el contexto de un convenio universidad-empresa. La PYME es una empresa de desarrollo de software que tiene un ERP perteneciente al sector socio-sanitario y cuenta con más de 40 empleados y más de 700 clientes.

Como herramienta de apoyo para la aplicación efectiva de la metodología TUNE-UP surge TUNE-UP Process Tool. Esta ofrece a los agentes un

planificador personal, un gestor de unidades de trabajo y un planificador de versiones. La aplicación se rige por un motor de workflows que se encarga de coordinar las distintas actividades, incidencias y programadores. Estos workflows además permiten una gran flexibilidad, posibilitando el salto entre actividades dentro del mismo flujo de actividades. La definición de los mismos es el punto que nos lleva a la motivación de esta tesis.

1.2 Motivación

La herramienta TUNE-UP Process Tool está guiada por el motor de workflows, el cual se basa en los workflows definidos en el sistema. Actualmente están definidos más de 20 workflows, unos específicos para un determinado producto, otros compartidos entre diferentes productos, unos con pocas actividades y otros con hasta 30 actividades. Cada producto utiliza alrededor de cinco workflows. El motivo de este trabajo es abordar un tema pendiente por parte de la herramienta, la gestión de los mismos, ya que no existe un editor y tampoco un visor que se facilite la gestión de workflows.

En el caso del visor, el funcionamiento habitual es abrir desde la propia aplicación un archivo HTML elaborado en Microsoft Visio, que añade funcionalidades de zoom a una imagen, generada manualmente, que representa el diseño del workflow en cuestión. Un ejemplo de esto sería la Figura 1.

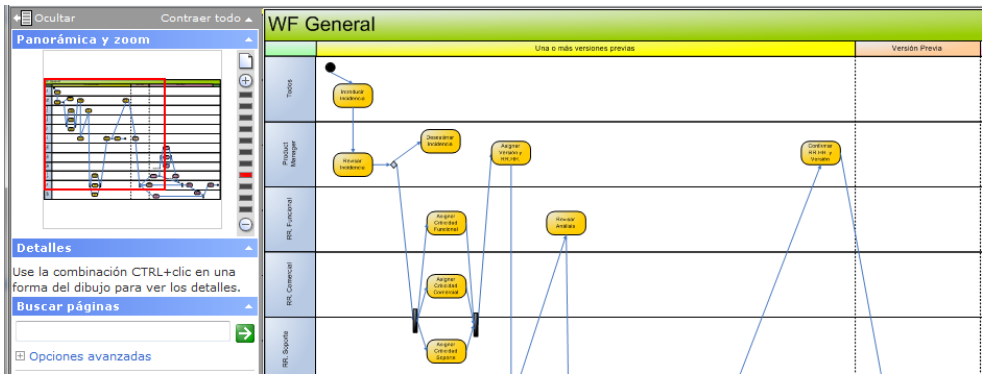


Figura 1 Workflow generado con Microsoft Visio

Aunque la presentación ofrecida por Visio es satisfactoria existen deficiencias. Una modificación del workflow supone modificar el archivo de Visio y volver a generar el HTML. Es un proceso manual engorroso que es evitable con un visor que interprete los workflows definidos en la base de datos, además de que existe el riesgo de que los workflows elaborados manualmente en Visio sean inconsistentes respecto a lo almacenado en la base de datos y explotado por el motor de workflows.

Más crítico aún es la propia edición de los workflows, ya que actualmente es totalmente manual. Los workflows están representados en la base de datos a partir de tablas que definen las distintas actividades y sus conexiones entre ellas. Esta representación no es trivial y propensa a errores durante su modificación manual, ya que no hay ninguna comprobación fuera de las propias claves ajenas de las tablas. El objetivo de este trabajo de tesis es diseñar y construir una extensión de TUNE-UP Process Tool que permita una edición gráfica y sencilla de los workflows y que asegure la corrección de los mismos respecto a las reglas de formación del motor de workflows.

1.3 Estructura

En el presente documento se muestra el proceso de desarrollo del módulo de gestión gráfica de workflows, a continuación se presenta la estructura de la tesis de máster.

En el capítulo 1 se incluye una introducción de la motivación del trabajo, explicando las necesidades a cubrir con el módulo desarrollado.

En el capítulo 2 se realiza una introducción sobre los workflows, pasando a explicar brevemente la notación estándar BPMN, para acabar comentando varias de las herramientas de tipo BPM existentes en el mercado.

El capítulo 3 se centra en la aplicación de los workflows en el proceso de desarrollo de software, para ello se introduce la metodología TUNE-UP y la herramienta que la soporta, a la cual se le añadiría el módulo de gestión gráfica de workflows: TUNE-UP Process Tool.

En el capítulo 4 se expone el estudio previo de búsqueda de un componente de edición de diagramas que facilite la edición de diagramas de workflow, sobre el cual se desarrollará el módulo. Se fijan los criterios de búsqueda para luego enumerar los distintos componentes con sus pros y sus contras basándose en un pequeño ejemplo. Finalmente se presentan las conclusiones y la elección tomada.

En el capítulo 5 se detalla la especificación del módulo a desarrollar partiendo de la descripción del sistema actual. El módulo queda dividido entre el visor y el editor, teniendo funcionalidades distintas como así se muestra en la arquitectura propuesta y en los casos de uso. El capítulo concluye con las pruebas de aceptación del módulo, las cuales también estarán divididas para las dos partes del módulo.

El capítulo 6 concluye el trabajo mediante las conclusiones obtenidas en el proceso y el trabajo futuro surgido a raíz de esta tesis de máster.

Capítulo 2: Workflows

2.1 Introducción

Un proceso de negocio es un conjunto de tareas lógicamente relacionadas, ejecutadas para obtener un resultado de negocio (Allen, 2001) [3].

Los procesos de negocio pueden ser controlados y administrados por un sistema basado en software. Los procesos de negocio automatizados de esta manera se denominan workflow. Esta automatización resulta en una importante potenciación de las virtudes de dicho proceso. Se obtienen mejoras en cuanto a rendimiento, eficiencia y productividad de la organización.

El caso particular de la industria del desarrollo de software, no es diferente al del resto de las industrias. Dentro de ella, se encuentran los procesos de negocios tendientes a la construcción o generación de un producto (software) de calidad en un tiempo determinado. El proceso de negocio más importante dentro de la industria de desarrollo de software es conocido como "metodologías de desarrollo", encargadas de guiar la producción. Este trabajo aporta a la optimización del proceso de producción de software mediante la automatización de las metodologías de desarrollo. Para esto se trabajó sobre la hipótesis de que el proceso de desarrollo de software es un tipo de proceso de negocio particular, y los procesos de negocio pueden ser automatizados en todo o en parte a través de un motor de workflow, el objetivo es transformar el proceso de desarrollo de software en un proceso de un workflow para poder lograr su automatización en todo o en parte. El paradigma workflow ofrece interoperabilidad con otros sistemas, ejecución en ambientes distribuidos, facilidades para el monitoreo y manejo de recursos humanos.

2.2 Workflow

Un workflow se define como la automatización total o parcial de un proceso de negocio, durante la cual documentos, información o tareas son intercambiadas entre los participantes conforme a un conjunto de reglas procedimentales preestablecidas.

Un workflow comprende un número de pasos lógicos, conocidos como actividades. Una actividad puede involucrar la interacción manual o automática con el usuario.

Un motor workflow es un sistema de software que controla la ejecución de las actividades definidas en el workflow. La WfMC ha definido un Modelo de Referencia Workflow (Hollingsworth, 1995) [15]. Este modelo define 5 interfaces para la interoperabilidad de diferentes productos con un motor workflow.

La interfaz 1 especifica el formato de intercambio común para soportar la transferencia de definiciones de procesos entre productos diferentes, utilizando un lenguaje de definición de procesos como el XML Process Definition Language (XPDL) definido por la WfMC o el Business Process Execution Language for Web Services (BPEL4WS) [6] adoptado por OASIS. XPDL permite escribir especificaciones de procesos workflow de manera estandarizada. Esto significa que cualquier definición de proceso que cumpla con todos los requisitos establecidos en la interfaz 1 podrá ser tomada como entrada por cualquier motor workflow que respete el estándar establecido por la WfMC, por ejemplo OFBiz Workflow Engine [29] u Open Business Engine [31].

BPEL4WS es un lenguaje para la especificación de procesos de negocio, el cual permite especificar procesos de negocio basados en servicios Web, esto es, que sólo pueden importar y exportar funcionalidad mediante servicios Web. La especificación inicial (BPEL4WS 1.0) fue desarrollada por IBM, Microsoft y BEA. WebSphere Process Server de IBM y BPEL Process Manager de Oracle son ejemplos de motores de workflow que implementan BPEL4WS. Es importante a la hora de modelar un proceso de negocio poder utilizar una herramienta independiente de la implementación, así, de esta manera, poder utilizar la especificación del proceso de negocio para diferentes plataformas. Una herramienta de estas características que está siendo muy utilizada por grandes empresas es BPMN.

La OMG junto con la Bussines Process Modeling Initiative (BPMI) [7] han desarrollado una notación para el modelado de procesos de negocio. Esta notación se denomina Bussines Process Modeling Notation (BPMN) [8]. BPMN define una notación para la definición de procesos de negocio, lo que es una plataforma independiente con respecto a definiciones específicas (por ejemplo XPDL o BPEL4WS) de procesos de negocio. Esta notación define una representación abstracta para la especificación de procesos ejecutables de negocio que se ejecutan dentro de una empresa (con o sin intervención humana); y puede colaborar con otro proceso de negocio independiente ejecutado en otra unidad de negocio o empresa. Partiendo de un modelo especificado en BPMN se puede obtener, mediante un mapping, la definición de un proceso de negocio en un lenguaje especifico como ser XPDL o BPEL4WS. En está definido el mapping de BPMN a BPEL4WS. Los elementos de la notación se pueden clasificar en elementos de flujo, de conexión, swinlanes y artefactos. Estos elementos que forman parte de la notación están especificados en el metamodelo BPMN. Este metamodelo está definido en el nivel M2 de la OMG y está basado en MOF [30].

2.3 BPMN

2.3.1 Introducción

El contenido de este apartado ha sido extraído del artículo de introducción a BPMN por Stephen A. White [40].

BPMN es una notación estándar desarrollada por la BPMI. La especificación de la versión 1.0 salió al público en mayo del 2004. El objetivo principal de los esfuerzos de BPMN es dar una notación rápidamente comprensible por toda persona implicada en un proyecto, independientemente de su preparación. Desde el analista de negocio que hace el borrador inicial de los procesos, pasando por los desarrolladores técnicos responsables de implementar la tecnología que llevarán a cabo dichos procesos, llegando finalmente al cliente del negocio que gestionará y monitorizará esos procesos. Además, BPMN está apoyado en un modelo interno que genera el ejecutable BPEL4WS. Así, BPMN crea un puente estandarizado para el hueco entre el diseño de los procesos de negocio y la implementación de procesos.

BPMN define el Business Process Diagram (BPD), que se basa en una técnica de grafos de flujo para crear modelos gráficos de operaciones de procesos de negocio. Un modelo de procesos de negocio, es una red de objetos gráficos, que son actividades (trabajo) y controles de flujo que definen su orden de rendimiento.

2.3.2 Fundamentos

Un BPD está formado por un conjunto de elementos gráficos. Estos elementos habilitan el fácil desarrollo de diagramas simples que serán familiares para la mayoría de analistas de negocio (diagrama de flujo). Los elementos fueron elegidos para ser distinguibles los unos de los otros y para usar formas familiares para la mayoría de modeladores. Por ejemplo, las actividades se representan con rectángulos y las decisiones con diamantes. Debe notarse que uno de los objetivos del desarrollo de BPMN es crear un mecanismo simple para crear modelos de procesos de negocio, y al mismo tiempo que sea posible gestionar la complejidad inherente en dichos procesos.

El método elegido para manejar estos dos conflictivos requisitos fue organizar los aspectos gráficos de la notación en categorías específicas. Esto da un pequeño grupo categorías que alguien que lea un BPD pueda reconocer fácilmente los tipos básicos de elementos y pueda entender el diagrama. Dentro de las categorías básicas de elementos, se puede añadir información y variaciones adicionales para dar soporte a los requerimientos complejos sin cambiar dramáticamente el *look-and-feel* básico del diagrama.

Las cuatro categorías básicas de elementos son:

- Objetos de flujo
- Objetos conectores
- Artefactos
- *Swimlanes*

Objetos de flujo

Un BPD es un pequeño conjunto (tres) de elementos básicos, que son los Objetos de Flujo, de modo que los modeladores no tienen que aprender y reconocer un gran número de formas diferentes. Los tres objetos de flujo son:

- **Evento:** un evento se representa con un círculo como se puede ver en la Figura 2. Es algo que ocurre durante el curso del proceso de negocio. Estos eventos afectan al flujo del proceso y suelen tener una causa (disparador) o un impacto (resultado). Los eventos representados con un círculo con centro abierto permiten a los marcadores internos diferenciar diferentes disparadores y resultados. Hay tres tipos de eventos, basados en cuando afectan al flujo: Inicio, Intermedio y Fin.

Figura 2 Tipos de evento

- **Actividad:** una actividad se representa con un rectángulo redondeado, como se puede ver en la Figura 3, y es un término genérico para un tipo de trabajo desempeñado. Una actividad puede ser atómica o compuesta. Los tipos que hay son: *Tarea* y *Sub-Proceso.* El Sub-Proceso se distingue por una pequeña marca de suma en la parte central inferior de la figura.

Figura 3 Representación de una actividad

- **Gateway (compuerta):** una *gateway* se representa por la típica figura de diamante, como se puede observar en la Figura 4, y se usa para controlar la divergencia o convergencia de la secuencia de flujo. Así, esto determina las tradicionales decisiones, así como la creación de nuevos caminos, la fusión de estos o la unión. Los marcadores internos indicarán el tipo de control de comportamiento.

Figura 4 Representación de una compuerta

Objetos conectores

Los objetos de flujo se conectan entre ellos en un diagrama para crear el esqueleto básico de la estructura de un proceso de negocio. Hay tres objetos conectores que hacen esta función como se puede ver en la Figura 5:

Figura 5 Representación de los distintos tipos de conectores

- **Sequence Flow:** el flujo de secuencia se representa por una línea sólida con una cabeza de flecha sólida y se utiliza para mostrar el orden (la secuencia) en el que las diferentes actividades se ejecutarán en el proceso.
- **Message Flow:** el flujo de mensaje se representa por un línea discontinua con una punta de flecha hueca y se usa para mostrar el flujo de mensajes entre dos participantes del proceso separados (entidades de negocio o roles de negocio). En BPMN, dos *pools* separadas en el diagrama representan los dos participantes.

- **Association:** una asociación se representa por una línea de puntos con una punta de flecha de líneas y se usa para asociar datos, texto, y otros artefactos con los objetos de flujo. Las asociaciones se usan para mostrar entradas y salidas de las actividades.

Para los modeladores que requieren o desean más precisión para crear modelos de proceso por motivos de documentación y comunicación, los elementos básicos más los conectores dan la posibilidad de crear fácilmente diagramas comprensibles. Un ejemplo sería la Figura 6, en la que se puede observar la facilidad de comprensión que ofrece BPMN.

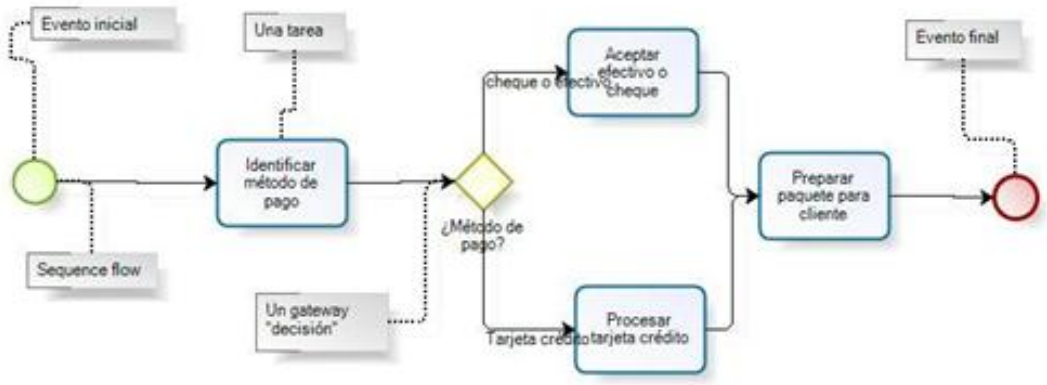


Figura 6 Diagrama de ejemplo

Para los diseñadores que necesiten un nivel más alto de precisión, para análisis detallado o que sean manejados por un Business Process Management System (BPMS), existen detalles adicionales que se pueden añadir a los elementos básicos.

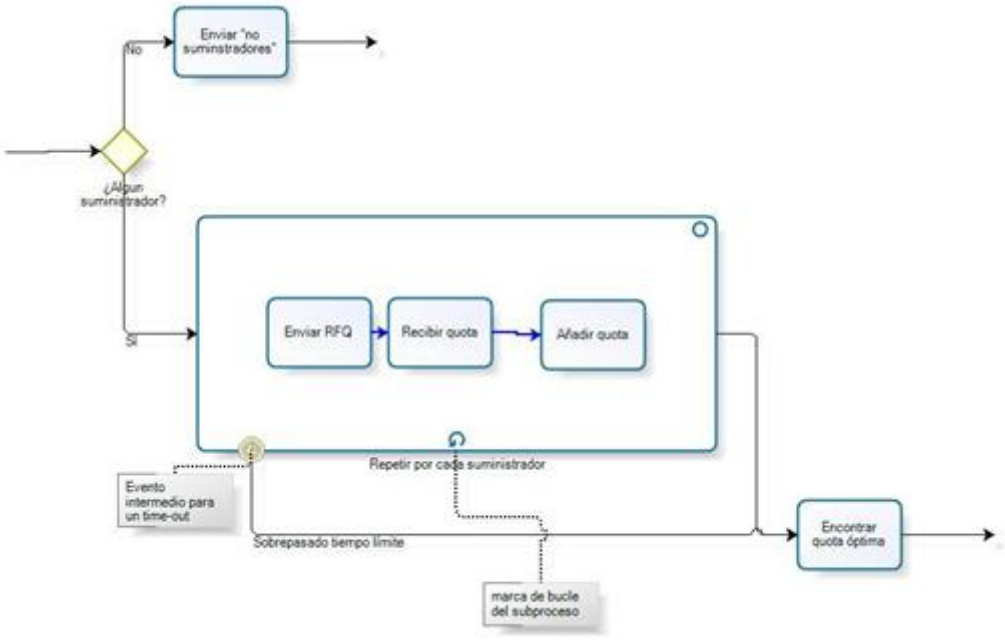


Figura 7 Diagrama de ejemplo con mayor precisión

Swimlanes (canales)

Muchas metodologías de modelado de procesos usan el concepto de *swimlanes* como un mecanismo para organizar actividades en categorías separadas visualmente para ilustrar diferentes capacidades funcionales o responsabilidades. BPMN soporta los swimlanes con dos constructores principales, que quedan representados en la Figura 8. Los dos tipos de objetos swimlanes son:

- **Pool:** una *pool* representa un Participante de un Proceso. Además actúa como un contenedor gráfico para dividir un conjunto de actividades desde otros pools, normalmente en el contexto de B2B.
- **Lane:** una *lane* es una sub-partición dentro de un pool y extiende la longitud del pool, verticalmente u horizontalmente. Las lanes se usan para organizar y categorizar actividades.

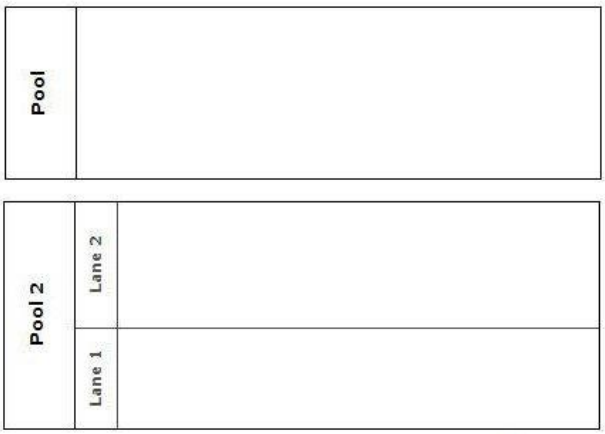


Figura 8 Representación de swimlanes

Las pools se usan cuando un diagrama implica dos entidades de negocio o participantes separados y están físicamente separados en el diagrama. Las actividades dentro de pools separadas se consideran procesos autocontenidos. Así, el flujo de secuencia no debe cruzar el límite de un pool. El flujo de mensajes se define como el mecanismo para mostrar las comunicaciones entre dos participantes, y, de este modo debe conectar dos pools (o los objetos dentro de las pools). Eso mismo se puede observar en el siguiente ejemplo de la Figura 9.

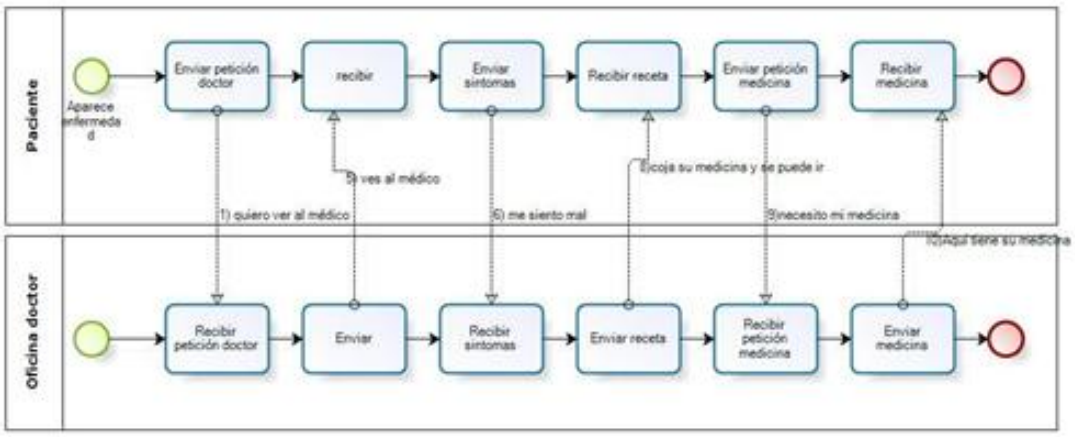


Figura 9 Diagrama de ejemplo con swimlanes

Las pistas (lanes) están más estrechamente relacionadas con las metodologías tradicionales de las swimlanes. Las pistas se suelen usar para separar las

actividades asociadas con la función o rol de una compañía específica. El flujo de secuencia puede cruzar los límites de las pistas dentro de un pool, pero el flujo de mensajes no puede ser usado entre objetos de flujo en pistas de mismo pool.

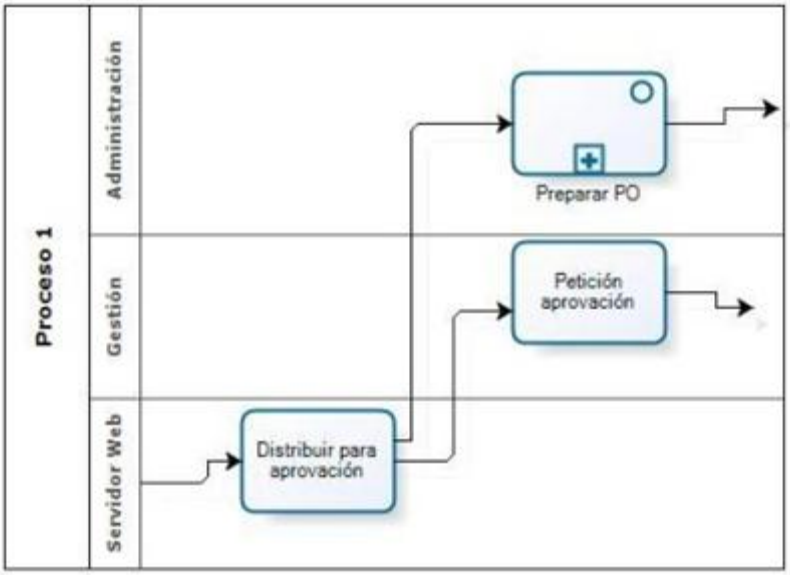


Figura 10 Diagrama de ejemplo con cruce entre pistas

Artefactos

BPMN fue diseñado para permitir a los modeladores y las herramientas de modelado un poco de flexibilidad a la hora de extender la notación básica y a la hora de habilitar un contexto apropiado adicional según una situación específica, como para un mercado vertical (por ejemplo, seguros o banca). Se puede añadir cualquier número de artefactos a un diagrama como sea apropiado para un contexto de proceso de negocio específico. La versión actual de la especificación de BPMN sólo tiene tres tipos de artefactos BPD predefinidos, como se puede ver en la Figura 11, los cuales son:

- **Data Object:** los objetos de datos son un mecanismo para mostrar como los datos son requeridos o producidos por las actividades. Están conectados a las actividades a través de asociaciones.

- **Group:** un grupo es representado por un rectángulo redondeado con línea discontinua. El agrupamiento se puede usar documentación o análisis, pero no afecta al flujo de secuencia.
- **Annotation:** las anotaciones son mecanismos para que un modelador pueda dar información textual adicional.

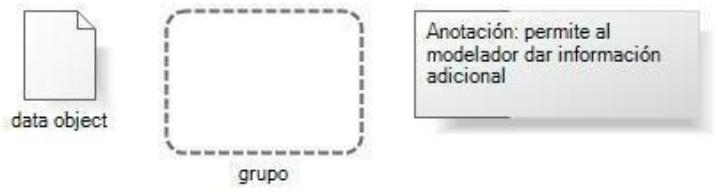


Figura 11 Representación los distintos artefactos

Los modeladores pueden crear sus propios tipos de artefactos, que añaden más detalle sobre cómo se ejecuta el proceso. Sin embargo, la estructura básica del proceso, determinada por las actividades, gateways, y flujos de secuencia, no se cambia por añadir artefactos al diagrama, como se puede ver en la Figura 12.

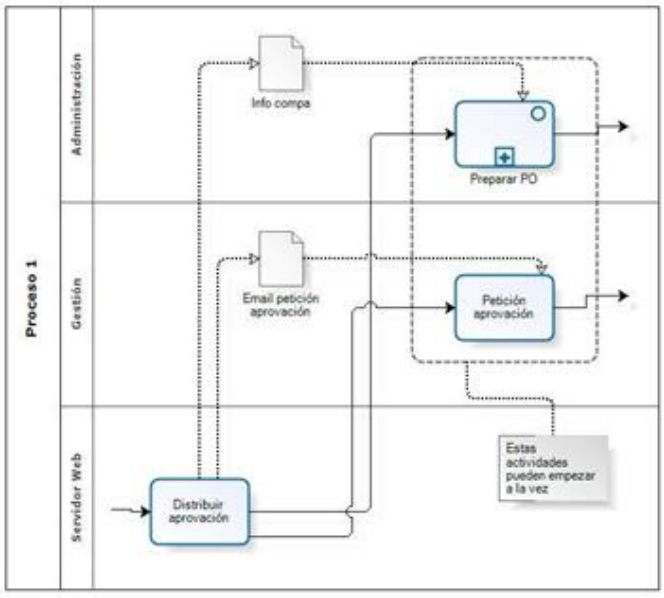


Figura 12 Diagrama de ejemplo con artefactos

2.3.3 Uso general de BPMN

El modelado de procesos de negocio se usa para comunicar una amplia variedad de información a diferentes audiencias. BPMN está diseñado para cubrir muchos tipos de modelados y para permitir la creación de segmentos de proceso así como procesos de negocio end-to-end, con diferentes niveles de fidelidad. Dentro de la variedad de objetivos de modelado de procesos, hay dos tipos de modelos básicos que se pueden crear con un BPD:

- Procesos B2B colaborativos (públicos)
- Procesos de negocio internos (privados)

Procesos B2B colaborativos

Un proceso B2B colaborativo ilustra las interacciones entre dos o más entidades de negocio. Los diagramas para estos tipos de procesos están diseñados generalmente desde un punto de vista global, mostrando las interacciones entre los participantes. Las interacciones están ilustradas como una secuencia de actividades y los patrones de intercambio de mensajes entre participantes. Las actividades para los participantes pueden ser considerados como los *"touch-points"* entre participantes, así pues, el proceso define las interacciones que son visibles al público para cada participante. En el caso de un proceso en un único Pool (por ejemplo, para un participante), el proceso público es también llamado proceso *abstracto*. Los procesos reales (internos) alcanzan un mayor detalle que en el caso de los procesos B2B colaborativos. La Figura 13, es una repetición de la Figura 9 para mostrar un ejemplo de proceso colaborativo B2B.

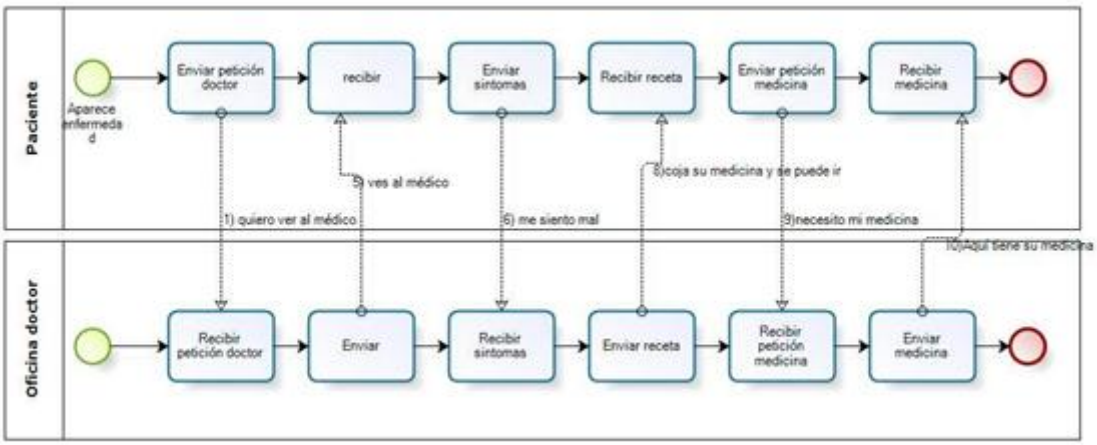


Figura 13 Diagrama de ejemplo de proceso colaborativo B2B

Procesos de negocio internos

Un proceso de negocio interno se enfocará generalmente en el punto de vista de una única organización de negocio. Aunque los procesos internos suelen mostrar interacciones con participantes externos, definen las actividades que generalmente no están visibles para el público, denominadas privadas.

Si se usan swimlanes entonces un proceso interno estará contenido dentro de un solo Pool. El flujo de secuencia del proceso está por lo tanto contenido dentro de un Pool y no puede cruzar los límites del Pool. El flujo de mensajes puede cruzar los límites del Pool para mostrar las interacciones que existen entre procesos de negocios internos separados. Así, un solo diagrama de procesos de negocio puede mostrar múltiples procesos de negocio privados.

2.3.4 Diferentes niveles de precisión

El modelado de procesos de negocio suele empezar capturando actividades de alto nivel para luego ir bajando de nivel de detalle dentro de diferentes diagramas. Pueden haber múltiples niveles de diagramas,

dependiendo de la metodología usada para desarrollar los modelos. De todas formas BPMN es independiente de cualquier metodología.

La Figura 14 muestra un ejemplo de un proceso de alto nivel, capturados para un caso de estudio de BPMN. Se trata de una serie de sub procesos con tres puntos de decisión

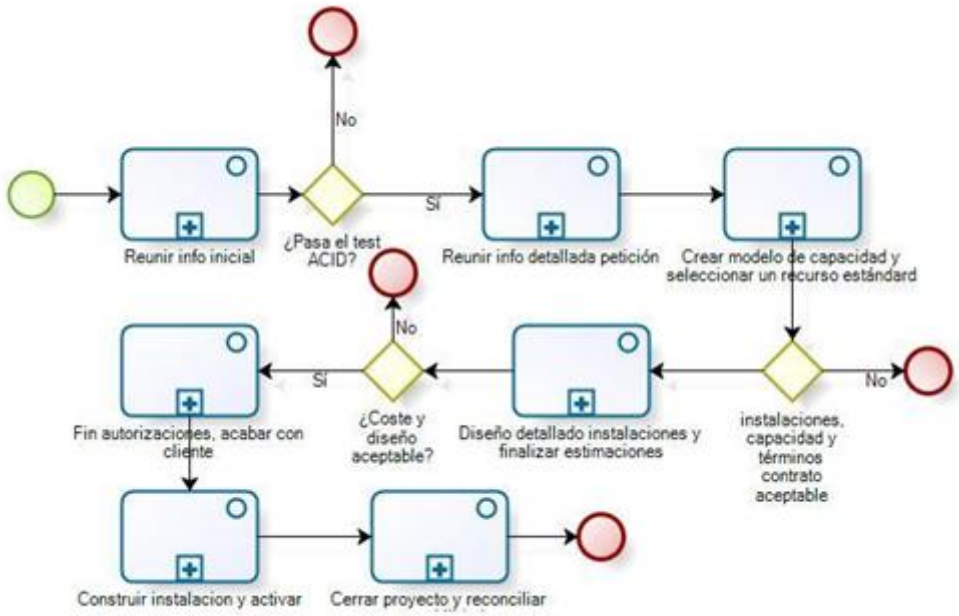


Figura 14 Diagrama de ejemplo de proceso de alto nivel

En la Figura 15 se baja de nivel para mostrar en detalle el primer sub proceso: dos pools, una para los clientes y otra para la compañía suministradora Este diagrama muestra un proceso de negocio interno para la compañía y un proceso abstracto para el cliente.

Las actividades de la compañía están particionadas con pistas o lanes para mostrar los roles/departamentos responsables de su rendimiento.

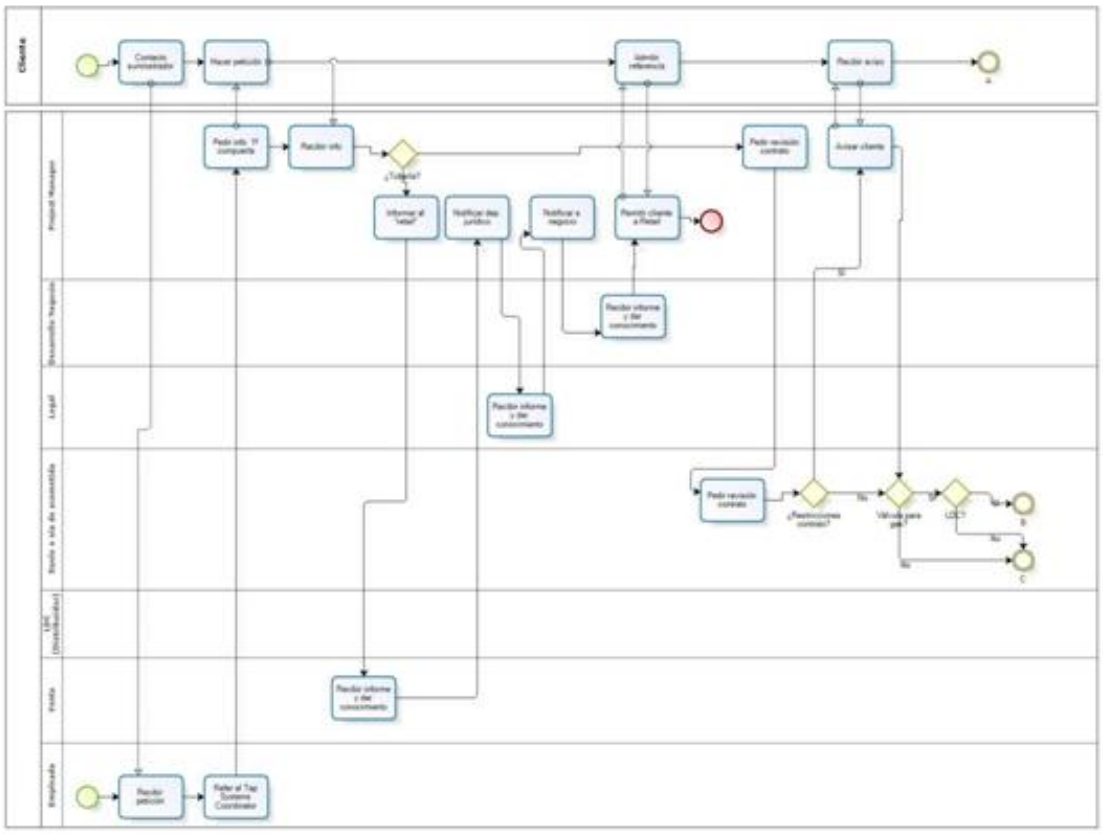


Figura 15 Diagrama de ejemplo de proceso de bajo nivel

2.3.5 Valor de modelar en BPMN

Los miembros del BPMI Notation Working Group representan un gran segmento de la comunidad de modelado de procesos de negocio y han llegado a un consenso y presentan BPMN como la notación de modelado de procesos de negocio estándar. El desarrollo de BPMN es un paso importante para reducir la fragmentación que existe con la gran cantidad de herramientas de modelado de procesos y notaciones.

El BPMI Notation Working Group tiene una gran experiencia con muchas de las notaciones existentes y trabajan para consolidar las mejores ideas de todas estas notaciones para crear una sola notación estándar. Ejemplos de otras notaciones o metodologías que fueron revisadas son: diagramas de actividades de UML, UML EDOC Business Processes [38], IDEF [18], ebXML BPSS [10], RosettaNet [34], Event Driven Process Chains (EPCs) [16].

Una única notación bien definida reduce la confusión entre los usuarios IT y de negocios.

Otro factor del desarrollo de BPMN es que, históricamente, los modelos de procesos de negocio desarrollados por la gente de negocios han estado técnicamente separados de las representaciones de procesos requeridas por los sistemas diseñados para implementar y ejecutar dichos procesos. Así, era necesario traducir manualmente los modelos de procesos de negocio originales a los modelos de ejecución. Esas traducciones están sujetas a errores y dificultan a los dueños del proceso entender la evolución y el rendimiento de los procesos desarrollados.

2.3.6 Mapear un diagrama BPMN a BPEL4WS

Para ayudar a aliviar el vacío técnico de modelado, un objetivo clave para el desarrollo de BPMN era crear un puente entre la notación de modelado de procesos de negocios y los lenguajes de ejecución respecto a las Tecnologías de la Información que implementan los procesos que hay dentro de un sistema. Los objetos gráficos de BPMN, más un buen número de atributos de estos objetos, se han mapeado al Business Process Execution Language para Web Services (BPEL4WS v1.1), el estándar de facto para la ejecución de procesos. En la Figura 16 tenemos un segmento de un proceso de negocio que marca el mapeo con BPEL4WS.

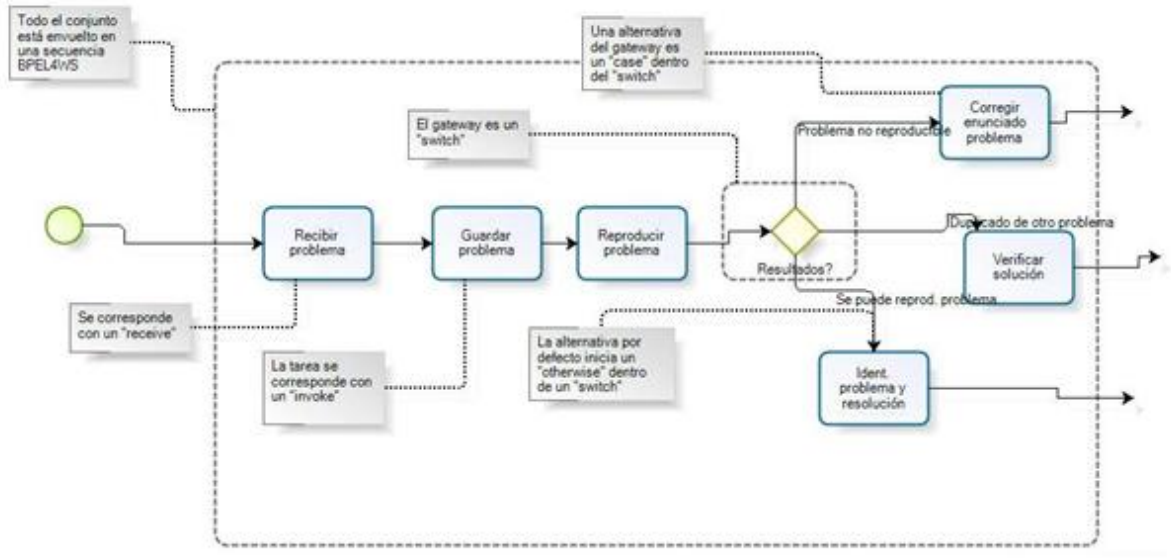


Figura 16 Diagrama con mapeo a BPEL4WS

2.3.7 El futuro de BPMN

Aunque la especificación de BPMN se encuentra en su versión 1.2, muchas compañías la soportan e implementan dicha especificación, y existe una propuesta para la futura versión 2.0. El futuro inmediato dará un punto de experiencia entre usuarios y vendedores que permitirá, mediante feedback, afinar detalles de la especificación, en concreto con BPEL4WS. En las siguientes versiones de mantenimiento es de esperar un esfuerzo en estandarización de los artefactos para que soporten modelado de negocios generales y dominios de negocios verticales (seguros, manufacturación, finanzas). Además, se está intentando encajar BPMN en un mayor contexto de modelado de negocios de alto nivel (incluyendo reglas de negocio y estrategias de negocio).

2.4 Herramientas

En esta sección se van a mostrar algunas de la gran cantidad de Suites BPM que se pueden encontrar en el mercado. Dado su número y la ingente información disponible se indica únicamente una breve descripción de la misma.

2.4.1 Lotus Notes

Dentro de las herramientas orientadas a procesos colaborativos guiados por workflow una de las más utilizadas es Lotus Notes, desarrollada por Lotus Software, una filial de IBM [23]. Los puntos fuertes de esta herramienta es la sencillez de desarrollo de aplicaciones, la posibilidad de utilizarla tanto en escritorio como en web y la integración con el entorno. Es de tipo cliente/servidor, siendo Notes el cliente y Domino el servidor. Con diferencia es la herramienta más madura del sector, siendo ampliamente utilizada.

2.4.2 IBM WebSphere Business Monitor

WebSphere Business Monitor es un producto integrado en el catálogo de gestión de procesos empresariales de IBM [17]. Incluye múltiples funciones avanzadas de supervisión de procesos empresariales con sofisticadas funciones de análisis, con un gestor de acción adaptable que invoca una acción seleccionada en tiempo real o configura un conjunto de acciones en función de reglas predefinidas.

Ofrece supervisión en tiempo real con métricas, pantallas visuales y alertas:

- Mide el rendimiento empresarial para conseguir los objetivos mediante una tarjeta de resultados.

- Alerta a los usuarios clave de las situaciones empresariales decisivas que empresariales decisivas que requieren alguna acción.
- Envía notificaciones de alerta a partir de situaciones anómalas.
- Realiza un seguimiento de los flujos de trabajo empresariales.
- Supervisa las métricas de los procesos empresariales, como la duración transcurrida o el trabajo.

IBM WebSphere Business Monitor permite supervisar los procesos empresariales en tiempo real, alerta y notifica a los usuarios para mejorar continuamente los procesos empresariales.

2.4.3 Microsoft BizTalk

BizTalk es la apuesta por parte de Microsoft en cuanto a BPM a nivel de orquestación de servicios, integrando diferentes tipos de software utilizados en una empresa para automatizar e integrar los procesos de negocio [26]. BizTalk está totalmente orientado a servicios y cuenta con excelentes herramientas gráficas y de transformación, siendo posible definir mensajes, canales y procesos mediante workflows de manera gráfica. Es un integrador de sistemas dispares.

2.4.4 K2.NET

Para la automatización de procesos de negocio que requieren ejecución de tareas por parte de personas, K2 ofrece una plataforma de workflows humanos completamente integrada con el entorno de aplicaciones de Microsoft [21]. Entre sus características están:

- Permite rápidas soluciones conjuntas para optimizar interacciones entre personal, sistemas y procesos.
- Proporciona visibilidad a actividades dirigidas por procesos para identificar formas de optimizar el proceso: detección de cuellos de botella, escalabilidad, gestión de excepciones.

K2.net a su vez se divide en los siguientes componentes:

- Studio: un potente entorno de diseño de flujos de trabajo. Une personal, usos e información en procesos de negocio integrados y automatizados.
- Templates: un editor de etapas de flujos de trabajo. Permite a los usuarios crear estos componentes de flujos de trabajo sin necesidad de programar.
- SmartForm controls for ASP.NET: conjunto de formularios ASP.NET que permiten a los usuarios a construir formularios de flujo de trabajo, que se ejecutan sobre ASP.NET y se acceden mediante un navegador web estándar.
- Server: proporciona alta escalabilidad, eficacia y una plataforma segura para procesos de negocio entre personal y entre el usuario y el sistema.
- Workspace: autoriza a los usuarios a manejar y rastrear tareas del flujo de trabajo. Esto informa sobre las actividades en la empresa, y autoriza a los gestores del conocimiento a usar esta información para eliminar cuellos de botella, conocer el seguimiento de los procesos y asegurar niveles de servicio óptimos.

2.4.5 Adobe LiveCycle Workflow

El software de Adobe LiveCycle Workflow ayuda a las organizaciones a agilizar, integrar y asegurar los procesos empresariales basados en el hombre dentro y fuera del cortafuegos, independientemente de que los usuarios estén conectados o no a la red [2]. Gracias a la única arquitectura basada en componentes de Adobe LiveCycle Workflow, las empresas y los profesionales de TI pueden ensamblar, de forma visual, flujos de trabajo completos que unifican de manera flexible y rápida a personas, sistemas, documentos, reglas empresariales y servicios Web. Puesto que aprovecha la tecnología estándar del sector, como PDF, J2EE, XML y los servicios Web, Adobe LiveCycle Workflow puede ampliarse e integrarse fácilmente en cualquier infraestructura de TI.

2.4.6 Auraportal BPM

AuraPortal BPM [4] es un sistema de Gestión Empresarial que ofrece en un solo paquete cuatro componentes relacionados entre sí que constituyen las áreas de mayor interés en la gestión empresarial:

- BPM (Gestión por Procesos) con Reglas de Negocio
- Intranet con Plataforma de Workflow
- Gestión Documental con MS Sharepoint
- Portales para Comercio Electrónico

Ofrece un modelador adaptado al estándar BPMN y un motor de procesos permite generar workflows a partir de modelos sin necesidad de programación.

2.4.7 DynaFlow EZ-Process

Importante herramienta de gestión de procesos de negocio que cuenta entre sus clientes a compañías importantes como Comcast Cable (mayor proveedor de cable de EEUU), Siemens, Fujitsu, MD Robotics (Fabricante del Manipulador remoto de lanzaderas espaciales bajo contrato con la NASA), Solar Turbines/Caterpillar, Herman Miller, GKN Sinter Metals, y otros muchos [12]. La suite de DynaFlow proporciona los siguientes componentes:

- EZ-Modeler: para realizar modelados de negocio independientes proporcionando un repositorio corporativo central para la gestión de procesos de negocio.
- EZ-Librarian: para apoyar la gestión de documentos/contenidos.
- EZ-Book: para realizar la gestión de conocimientos extendida mediante la generación dinámica de manuales dirigidos a roles de usuario para el entrenamiento u otros objetivos de la gestión del conocimiento.
- EZ-Publisher: para proporcionar a las organizaciones con portales de información sobre procesos de negocio dirigidos por roles de usuario la posibilidad de desarrollar sus procesos totalmente documentados.
- EZ-Compliance: para soportar la certificación de cumplimiento SOX (u otros) mediante la integración de procesos y controles, mandos de acceso para personal o aplicaciones y un motor de identificación dinámica de conflictos.
- EZ-Workflow: para apoyar la producción y capacidades de Workflow actuales, implicando un motor basado en reglas, un cuadro de mandos KPI y capacidades PDA.
- EZ-BPR-Analyzer: para apoyar a la optimización de procesos de negocio mediante el análisis cuantitativo y la simulación.
- EZ-ISO: para apoyar la certificación de calidad como ISO, QS, TS, OHSAS, etc…

2.4.8 Fujitsu – Interstage BPM

Herramienta creada por Fujitsu de gran reputación, entre otras cosas porquer Fujitsu España Services es Country Chair del organismo WfMC en España [13]. Este organismo es uno de los encargados de definir los estándares de workflow a nivel mundial. Además I-Flow es el software de workflow de Fujitsu ganador del premio "Gold Award for Excellence in Workflow".

La solución que aporta Fujitsu posibilita la organización automática del trabajo, un programa supervisa el proceso, asigna tareas, las avanza y monitoriza su progreso, automatizando procesos de negocio complejos, y disminuyendo los tiempos de desarrollo y mantenimiento de éstos. Con las soluciones de BPM de Fujitsu, podemos realizar tareas de forma paralela y documentar todo el proceso, lo que nos ofrece la oportunidad de hacer una reingeniería de los procesos de negocio. La solución está orientada a:

- Mejorar de los procesos de negocio
- Automatizar tareas
- Facilitar la monitorización y el control de procesos
- Simplificar la dinamización de la lógica de los procesos
- Disminuir los costes de desarrollo

La solución integral de BPM propuesta por FUJITSU se basa en:

- Consultoría de reingeniería de procesos.
- Servicios Profesionales de implantación, desarrollo y personalización.
- Tecnología de workflow: Interstage BPM (i-Flow) tecnología que posibilita el modelado de procesos de negocio para su automatización y monitorización on-line, reduciendo tiempos de desarrollo y aumentando el control.

La herramienta posee mecanismos de integración con sistemas externos a través de Web Services, Java y XML. También, se conecta con bases de datos externas a través de adaptadores.

Adicionalmente, admite programación a través de Java y JavaScript con la posibilidad de crear formularios de cliente de forma automática (HTML). Resumiendo las principales funcionalidades del producto i-Flow son:

- Diseño de procesos drag and drop
- Gestión de timers con distintos calendarios
- Control de notificaciones basadas en email
- Listener/triggers/eventos
- Soporte Web Services
- Posibilidad de personalización del cliente(css)
- Soporte a Java 1.4
- Módulos de reporting, analítico y auditoria
- Modificación de procesos en tiempo de ejecución
- Generación automática de formularios HTML
- Sistema de Gestión Documental
- Balanceo de Carga
- Plataforma de alta disponibilidad
- Firma Digital

2.4.9 Oracle Fusion Middleware

Oracle Fusion Middleware da soporte a un ciclo de vida completo para aplicaciones orientadas a servicios y ofrece interoperabilidad con la infraestructura de IT existente en la organización gracias a su arquitectura modular [33]. Oracle Fusion Architecture es un modelo unificado de tendencias emergentes como son la arquitectura Grid Computing, arquitectura orientada a servicios y arquitectura de información.

2.4.10 Ultimus BPM Suite

Ultimus BPM Suite aborda el ciclo de vida completo de los procesos de negocio [37]. La primera fase es el diseño de los procesos. Mediante la simulación, se desarrollan y documentan procesos optimizados desde el principio. A continuación se desarrollan e integran formularios y agentes. Luego se completa el proceso y se instala a los usuarios. Además de gestionar los proyectos, se pueden recopilar y analizar los datos del proceso para optimizaciones posteriores.

Ultimus es una plataforma en la cual se pueden automatizar muchos procesos de negocio internos. Los elementos destacables de Ultimus son:

- Amplia experiencia en automatización de procesos.
- Solución completa: se ocupa de todas las fases del ciclo de vida de un proceso de negocio con una completa solución de software que pueden utilizar todos los miembros de un equipo BPM con una amplia variedad de habilidades.
- Flexibilidad: permite la integración en sistemas con diferentes reglas y diferentes procesos. Además el software tiene ganchos o hooks de integración y personalización para ser fácilmente adaptados a una gran variedad de configuraciones y entornos.
- Económico: ofrece un modelo de licencia flexible y escalable.

La herramienta adapta los estándares más comunes del sector, como Microsoft .NET, XML Schema, Web Services (WSDL, SOAP), MAPI, SMTP, LDAP y ADSI. Todos los componentes del servidor son individuales, modulares y adaptables. También ofrece interfaces programables (API) completas para desarrollar procesos de negocio a medida.

2.4.11 Comentarios finales

En el mercado de herramientas BPM, como se ha podido observar, existen múltiples posibilidades. Todas ellas son herramientas que intentan dar una solución global para la gestión de procesos de negocio, pero ninguna está orientada al proceso de desarrollo de software. Carecer de este enfoque supone no cubrir ciertas necesidades propias del proceso de desarrollo de software, como tener workflows flexibles que permitan saltos entre las distintas actividades e incluso cambio de workflow.

Por ese motivo para la herramienta TUNE-UP Process Tool se optó por desarrollar un motor propio de workflows que fuera sencillo y a la vez potente que permitiese la flexibilidad necesaria.

Capítulo 3: Workflows para el proceso de desarrollo de software

3.1 Introducción

El número de metodologías y estándares relacionados a los procesos de producción de software han aumentado en los últimos años y a su vez el interés por parte de las empresas desarrolladores de software. Cada vez son más las que adoptan el modelo de referencia CMMI (Capability Maturity Model Integrated, 2009) [11] o los estándares SPICE (ISO, 1998) [19] e ISO 90003 (ISO, 2004) [20] y las que deciden seguir Metodologías Ágiles (p.ej. XP (Beck, 2000) [5], Scrum (Schwaber y Beedle, 2001) [35]) o Tradicionales (p.ej. RUP (Kroll et al., 2003) [22], Métrica 3 (MAP, 2005) [24]) en pos de una mayor efectividad.

A la hora de afrontar un proyecto de desarrollo de software hay que enfrentarse a las dificultades habituales sumando las específicas del mismo como los cambios en los requisitos o la colaboración entre participantes. Esto hace que las herramientas genéricas para planificación y seguimiento sean insuficientes.

En cuanto a las metodologías por un lado están las metodologías ágiles, que resultan excesivamente simplistas, y por otro, las herramientas tradicionales, que no ofrecen pautas para la gestión de un proyecto en un proceso iterativo e incremental. De la necesidad de una solución intermedia surge TUNE-UP como metodología que aborda el trabajo día a día en una PYME de desarrollo de software de manera pragmática y con una vocación de mejora continua del proceso. TUNE-UP se inspira en la esencia de PSP (Personal Software Process) (Watts, 2001) [39], donde se destaca que la base del éxito radica en una disciplina de trabajo y productividad individual centrada en la gestión de los compromisos.

En este capítulo veremos cómo se incorpora la gestión de workflows en la herramienta de apoyo a TUNE-UP, TUNE-UP Process Tool.

3.2 Introducción a TUNE-UP

TUNE-UP se caracteriza fundamentalmente por combinar los siguientes elementos:

- Modelo de proceso iterativo e incremental para el desarrollo y mantenimiento del software.
- Proceso de desarrollo dirigido por las pruebas (Test-Driven).
- Workflows flexibles para la coordinación del trabajo asociado a cada unidad de trabajo.
- Planificación y seguimiento continuo.
- Control de tiempos.

Un ejemplo del tipo de workflow que podemos encontrar en la herramienta sería la Figura 17, el cual representa una unidad de trabajo.

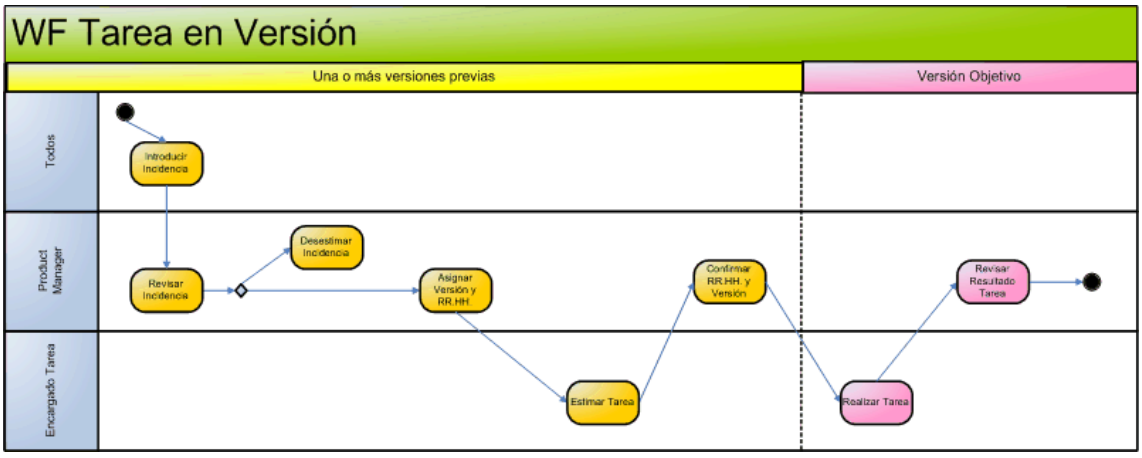


Figura 17 Un workflow de desarrollo simple para unidades de trabajo

3.3 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.

3.3.1 Planificador personal

El Planificador Personal (PP) es la herramienta que utilizará un agente para gestionar sus trabajos asignados durante su jornada laboral. La herramienta le ayudará a seleccionar qué actividad realizar en base a las prioridades y llevar un control sobre las tiempos dedicados y restantes de cada actividad. En la Figura 18 podemos ver el listado de actividades a la izquierda y sus unidades de trabajo correspondientes a la derecha.

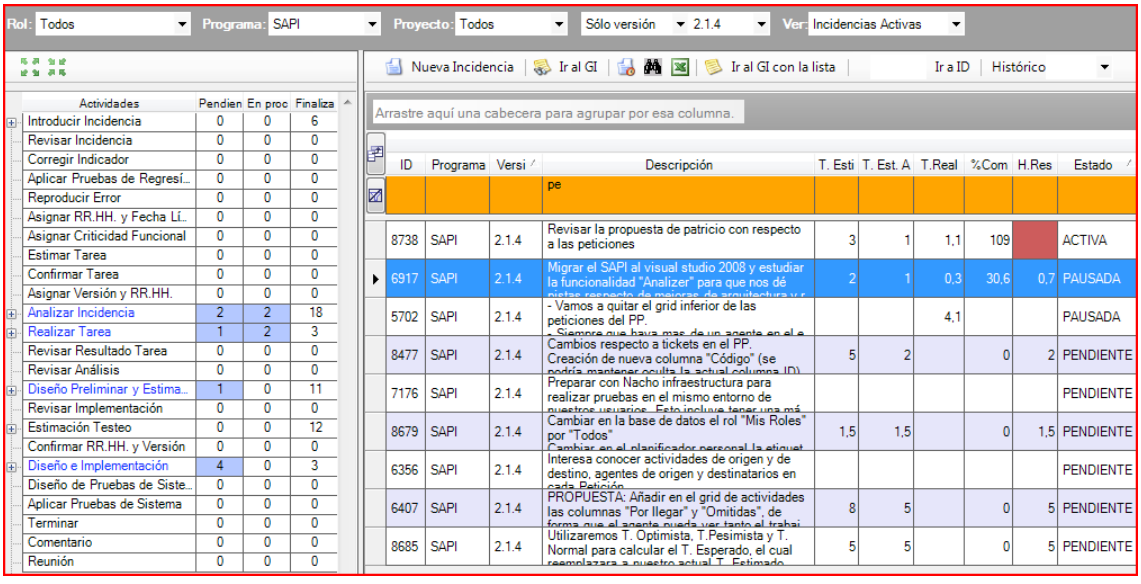


Figura 18 Fragmento de interfaz del Planificador Personal

3.3.2 Gestor de Unidades de Trabajo (GUT)

El agente para empezar a trabajar con una unidad de trabajo debe acceder mediante el GUT, el cual le sirve de apoyo para realizar su tarea en múltiples aspectos:

- Gestión de seguimiento: la herramienta ofrece datos de seguimiento, a partir de los que es posible saber por qué actividades ha pasado la unidad de trabajo, quienes han trabajado en ella, en qué estado está y a su vez permite modificar el estado de la misma como se puede ver en la Figura 19.

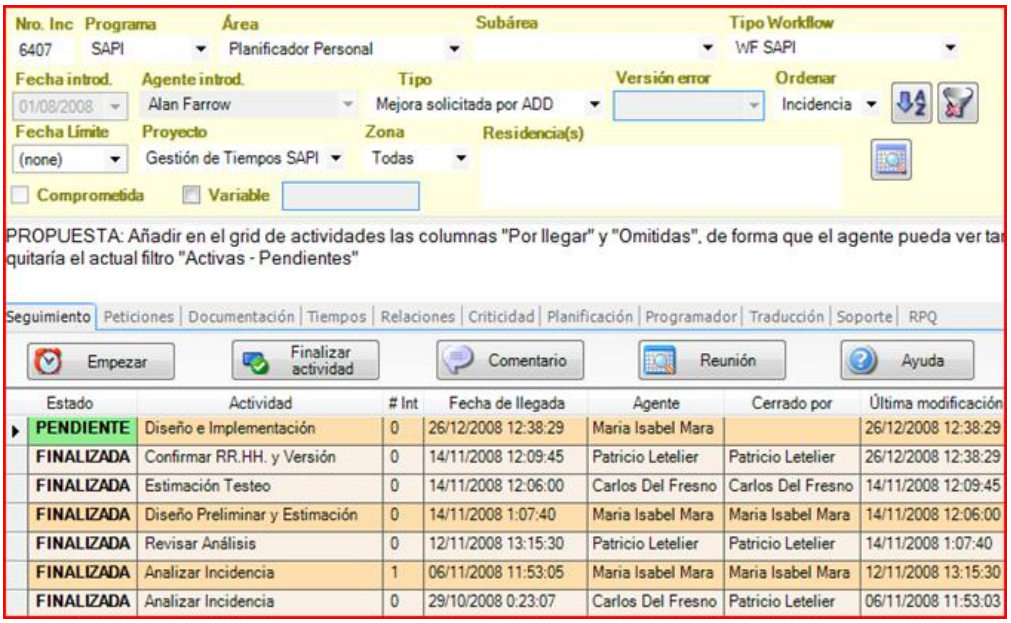


Figura 19 Gestor de Unidades de Trabajo – Pestaña Seguimiento

- Gestión de tiempos: el agente puede llevar un control del tiempo consumido en la unidad de trabajo y de sus estimaciones como se puede ver en la Figura 20.

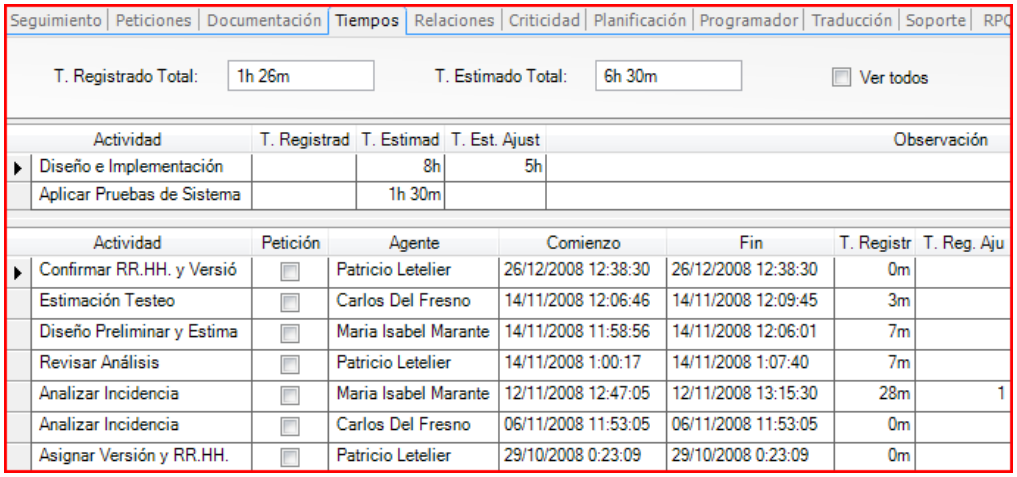


Figura 20 Gestor de Unidades de Trabajo – Pestaña Tiempos

- Gestión documental: dentro de una unidad de trabajo es posible asociar todos los documentos relacionados con la misma, incluyendo plantillas y un control de versiones de los documentos.

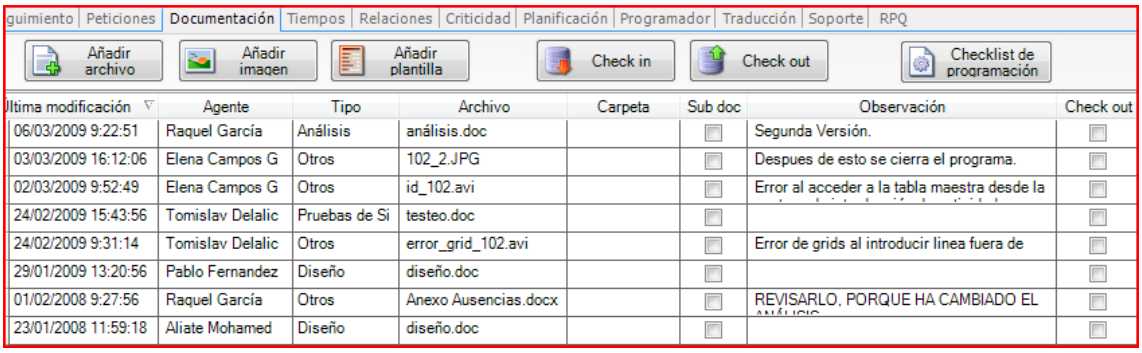


Figura 21 Gestor de Unidades de Trabajo – Pestaña Documentación

- Gestión de peticiones: mecanismo sencillo de comunicación entre los agentes para comentar respecto de la unidad de trabajo.
- Gestión de relaciones: relaciones de dependencia de la unidad de trabajo con respecto de otras
- Gestión de planificación: asignación de agentes a actividades de la unidad de trabajo y asignación a una versión.

3.3.3 Planificador de Versiones (PV)

El PV es una herramienta que ayuda a 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.


Incidencias | Carga de Agentes en Versión

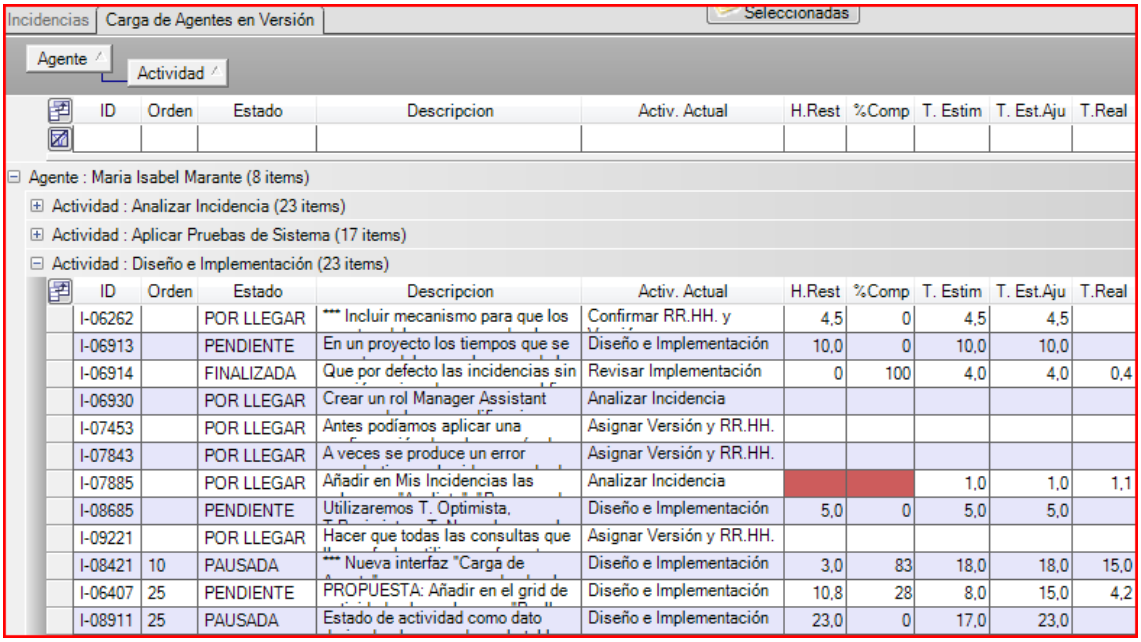
Arrastre aquí una cabecera para agrupar por esa columna.

ID	Version	Orden	Descripción	Proyecto	Activ. Actual	Analizar Incidencia
	2.1.4				Desestimar	
8679	2.1.4		Cambiar en la base de datos el rol "Mis Roles" por "Todos"...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
6261	2.1.4	10	Documentación de Ayuda - Permitir la edición de estos los documentos de...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
8477	2.1.4	15	Cambios respecto a tickets en el PP. Creación de nueva columna "Código" (se podría...	Tickets SAPI	Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
8395	2.1.4	20	Vamos a echarle un vistazo nuevamente para ver hasta qué punto sigue fallando y si encontramos...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
7110	2.1.4	30	Tener una pestaña de documentación para cada program, al estilo de la pestaña de las incidencia...		Analizar Incidencia (1) / Maria Isabel Marante	Maria Isabel Maran...
8194	2.1.4	40	Que aparezca en el PP la actividad "Desestimar" en el grid de actividades.		Analizar Incidencia (0) / Maria Isabel Marante	Maria Isabel Maran...
8635	2.1.4	40	Poner la funcionalidad "Ir al GI con la Lista" en el grid Detalles de Versión		Analizar Incidencia (0) / Maria Isabel Marante	Maria Isabel Maran...
6264	2.1.4	50	Mantenimiento de Tablas Maestras en el SAPI: Agentes, Roles, Actividades y Programas. Se po...		Diseño e Implementación (1) / Maria Isabel Marante	Maria Isabel Maran...
5702	2.1.4	100	PROPUESTA: Incluir las peticiones (con su estado) en el grid de actividades. El agente vería...	Gestión de Tiem...	Diseño e Implementación (1) / Maria Isabel Marante	Maria Isabel Maran...
6407	2.1.4	100	PROPUESTA: Añadir en el grid de actividades las columnas "Por llegar" y "Omitidas", de forma que	Gestión de Tiem...	Diseño e Implementación (0) / Maria Isabel Marante	Maria Isabel Maran...

Figura 22 Planificador de Versiones – Pestaña Incidencias

La ventana de la Figura 23 muestra la lista de unidades de trabajo asignadas a una versión de un determinado producto. El jefe de proyecto puede consultar los datos resumidos de cada unidad de trabajo en la versión, en particular, puede conocer la actividad actual en que se encuentra, el orden de prioridad, el esfuerzo que implica elaborar la unidad de trabajo, el agente asignado a las actividades principales del workflow, etc.

A su vez ofrece información a los agentes sobre el trabajo asignado y al jefe de proyecto la posibilidad de realizar un seguimiento sobre la carga de trabajo de los agentes.



ID	Orden	Estado	Descripcion	Activ. Actual	H.Rest	%Comp	T. Estim	T. Est.Aju	T.Real
Agente : Maria Isabel Marante (8 items)									
Actividad : Analizar Incidencia (23 items)									
Actividad : Aplicar Pruebas de Sistema (17 items)									
Actividad : Diseño e Implementación (23 items)									
ID	Orden	Estado	Descripcion	Activ. Actual	H.Rest	%Comp	T. Estim	T. Est.Aju	T.Real
I-06262		POR LLEGAR	*** Incluir mecanismo para que los	Confirmar RR.HH. y	4,5	0	4,5	4,5	
I-06913		PENDIENTE	En un proyecto los tiempos que se	Diseño e Implementación	10,0	0	10,0	10,0	
I-06914		FINALIZADA	Que por defecto las incidencias sin	Revisar Implementación	0	100	4,0	4,0	0,4
I-06930		POR LLEGAR	Crear un rol Manager Assistant	Analizar Incidencia					
I-07453		POR LLEGAR	Antes podíamos aplicar una	Asignar Versión y RR.HH.					
I-07843		POR LLEGAR	A veces se produce un error	Asignar Versión y RR.HH.					
I-07885		POR LLEGAR	Añadir en Mis Incidencias las	Analizar Incidencia			1,0	1,0	1,1
I-08685		PENDIENTE	Utilizaremos T. Optimista,	Diseño e Implementación	5,0	0	5,0	5,0	
I-09221		POR LLEGAR	Hacer que todas las consultas que	Asignar Versión y RR.HH.					
I-08421	10	PAUSADA	*** Nueva interfaz "Carga de	Diseño e Implementación	3,0	83	18,0	18,0	15,0
I-06407	25	PENDIENTE	PROPUESTA: Añadir en el grid de	Diseño e Implementación	10,8	28	8,0	15,0	4,2
I-08911	25	PAUSADA	Estado de actividad como dato	Diseño e Implementación	23,0	0	17,0	23,0	

Figura 23 Planificador de Versiones – Pestaña Carga de Agentes

Todo esto ayuda al jefe de proyecto a tomar decisiones correctivas como reasignar y dividir unidades de trabajo o modificar plazos de entrega.

3.3 Gestión de Workflows

El motor de workflows que incorpora TUNE-UP Process Tool se encarga del correcto orden de las unidades de trabajo y permite realizar saltos a cualquier actividad del workflow, soportando el paralelismo, secuenciación y bifurcación de las actividades.

La definición de los workflows se realiza mediante un proceso manual, directamente sobre la base de datos y la visualización de los mismos depende de imágenes generadas a partir de ficheros Microsoft Visio.

Los workflows del sistema debían ofrecer flexibilidad para la coordinación del trabajo asociado a cada unidad de trabajo, permitiendo saltos hace adelante o

hacia atrás en el workflow, así como cambios de agentes asignados e incluso cambio de workflow.

Por eso se decidió desarrollar un motor de workflows ad hoc para la herramienta TUNE-UP Process Tool.

Capítulo 4: Librerías para desarrollo de editores gráficos

4.1 Comparativa de componentes de diagramas

En este apartado hablaremos sobre el estudio previo de búsqueda de un componente que facilite la edición de diagramas de workflow. El componente buscado debía ser de tipo Windows Forms para .NET, para poder integrarlo con el resto de la aplicación. Los requisitos que el componente debía cumplir son los siguientes:

- Edición de diagramas.
- Edición de los tipos de nodos.
- Representación de swim lanes.
- Gestión de layout.
- Precio.
- Soporte y comunidad.
- Facilidad de aprendizaje.

4.2 Componentes a comparar

Dentro de la oferta de componentes destinados a la creación de diagramas, se encuentran varias opciones tanto de software comercial como de software de código abierto. Esta fue una de las cuestiones a tener en cuenta a la hora de seleccionar el componente. Los componentes de código libre por una parte permiten realizar modificaciones para ajustarlos a las necesidades del proyecto, pero a su vez no tienen versiones estables, muchos están aún en desarrollo y carecen de una documentación amplia. A su vez los componentes de tipo comercial incluyen numerosos ejemplos, soporte en foros, por mail, etc. y en su contra están el desembolso económico y las restricciones de uso.

4.2.1 AddFlow for .NET

Software comercial de Lassalle Technologies [1] de gestión de diagramas y workflows para .NET 2.0 con un precio de 499$. No incluye gestión de layout, se vende aparte o en un pack con Addflow por 699$. Los controles para .NET se han creado a partir de la versión para ActiveX, no como en otros casos.

Cabe destacar su simplicidad, con una pequeña interfaz y una gran flexibilidad. Además es muy ligero, siendo la librería de un tamaño de 316Kb. También es posible introducirlo en una página web mediante ActiveX.

En cuanto al soporte, la página carece de foro donde consultar dudas y que a la vez sirva de base de conocimiento, todo el soporte es vía mail. Como documentación incluye un único documento de Word y numerosos ejemplos pero la mayoría sólo muestran soluciones simples. Uno de ellos es el que se muestra en la Figura 24, en el que se implementan las swim lanes pero de una manera muy simple, únicamente anclando el eje y de las actividades.

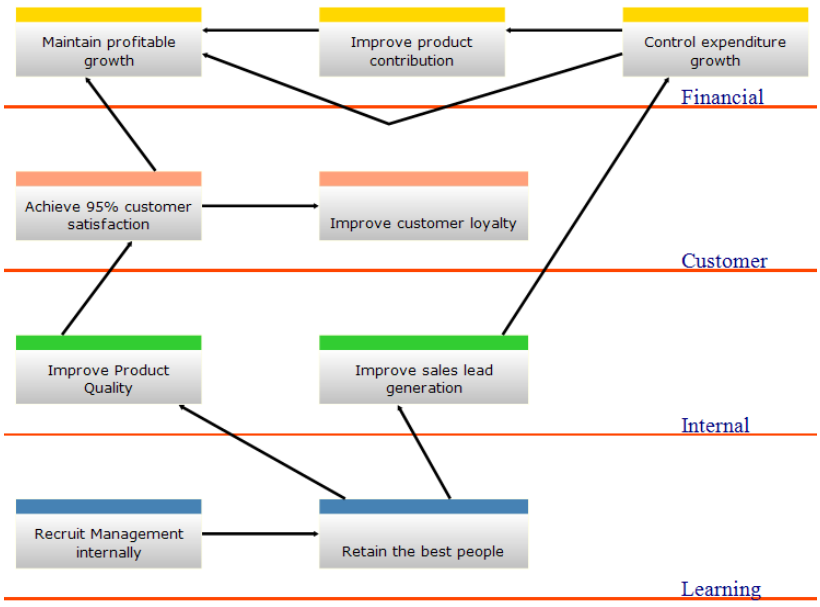


Figura 24 Ejemplo de diagramas de AddFlow

Para crear los nodos del ejemplo utilizamos el siguiente código:

```
Node node0 = AddFlow6.Nodes.Add(432, 16, 128, 48);
node0.Properties["swimline"].Value = "Financial";
AddChildren(node0, "Control expenditure growth");
```

Siendo "AddFlow6" el objeto que contiene todo el dibujo al que le añadimos un nodo con las coordenadas pasadas como parámetro del método "Add". A ese nodo le añadimos una propiedad "swimline" a la que se le asigna el valor "Financial", de esta manera es como definimos qué nodos están en qué *swim lane*.

Las conexiones entre nodos se crean desde el nodo origen, pudiendo definir por coordenadas los distintos puntos que va a tener la flecha que representa la conexión. En el ejemplo podemos ver como al nodo "node0" se le añade un enlace a sus enlaces de salida que lo une al nodo "node2". En este caso al querer que la flecha cambie su dirección añadimos el punto de inicio y el de salida a la colección "Points" del enlace y después le añadimos un tercer punto para que tenga forma de "V".

```
link = node0.OutLinks.Add(node2);
link.Points[0] = new PointF(432, 48);
link.Points[1] = new PointF(144, 48);
link.Points.Add(new PointF(272, 96));
```

Lo que nos daría como resultado la Figura 25:

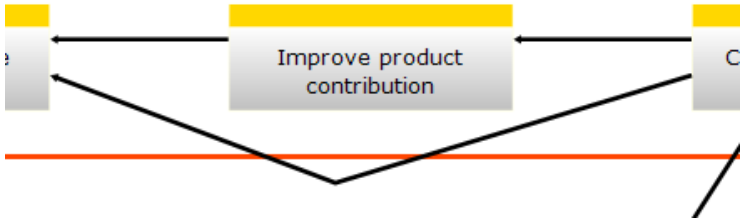


Figura 25 Ejemplo de diagramas de AddFlow

Como se puede observar tiene una gestión sencilla de nodos y conexiones pero es necesaria una abstracción mayor para la gestión completa de un editor de workflows.

En resumen, es una de las opciones a tener en cuenta, pero le restan enteros el alto precio y las carencias en documentación y ejemplos.

4.2.2 Syncfusion Essential Diagram

Software comercial que incluye una gran cantidad de controles de edición de diagramas [36]. En este caso incluye gestión de layout, un editor de diagramas y un editor de paletas de figuras, e incluso es posible importar figuras directamente desde Visio. También incluye la posibilidad de crear scripts para hacer diagramas interactivos por un precio de 495$. Ofrece una gran extensibilidad y personalización. Es uno de los más completos en cuanto a posibilidades tanto de funcionalidad como de presentación.

La documentación es otro de los puntos fuertes de los componentes de Syncfusion, con una gran documentación en la web como se puede ver en la Figura 26, con ejemplos incluidos, una amplia base de conocimiento y con un foro dedicado especialmente para cada tipo de componente.

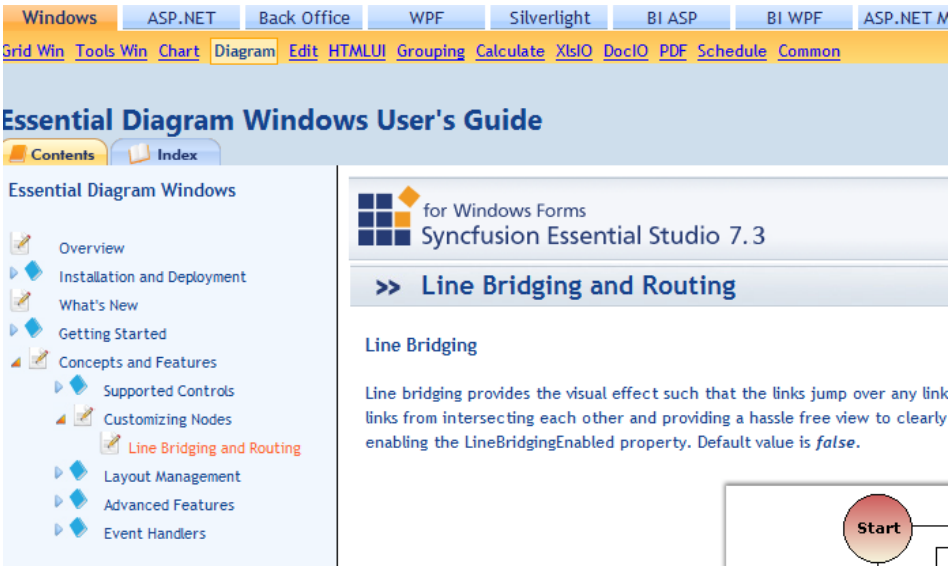


Figura 26 Documentación online de Syncfusion

La compañía desarrolladora, Syncfusion, está avalada por los diferentes premios al reconocimiento que ha recibido durante los últimos años, considerándola una de las mejores en el sector.

Por otro lado el componente es considerablemente más complejo que en el caso de AddFlow y supera en exceso las necesidades del proyecto. Como comparativa con AddFlow podemos ver las diferencias de implementación en el ejemplo de las *swim lanes* en Syncfusion.

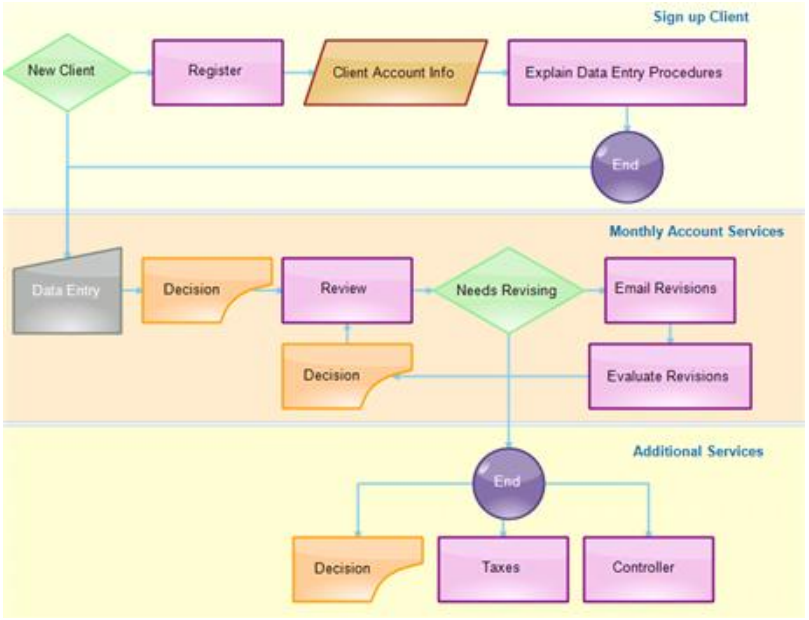


Figura 27 Ejemplo de diagramas de Syncfusion

Lo que resalta a primera vista, como se puede ver en la Figura 27, es la calidad de diseño en los componentes de Syncfusion respecto AddFlow, lo cual se ve traducido en mayor complejidad a nivel de presentación de la necesaria. Para representar el nodo "Data Entry" del diagrama anterior, asignarle un nombre, el tipo de fuente, el color de relleno, el borde y el gradiente es necesario todo este código:

```
//Data Entry node
pts = new PointF[] { new PointF(10, 250), new PointF(110, 230),
    new PointF(110, 310), new PointF(10, 310) };
nodePoly = new Polygon(pts);
nodePoly.FillStyle.Type = FillStyleType.LinearGradient;
nodePoly.FillStyle.Color = Color.White;
nodePoly.FillStyle.ForeColor = Color.FromArgb(181, 186, 181);
nodePoly.FillStyle.GradientCenter = 0.85f;
nodePoly.LineStyle.LineColor = Color.FromArgb(132, 142, 132);
nodeCaption = new Syncfusion.Windows.Forms.Diagram.Label();
nodeCaption.Text = "Data Entry";
nodeCaption.FontStyle.Italic = true;
nodeCaption.FontStyle.Bold = false;
nodeCaption.FontStyle.Family = "Georgia";
nodeCaption.FontStyle.Size = 11f;
nodePoly.Labels.Add(nodeCaption);
Node node6 = nodePoly;
this.diagram1.Model.AppendChild(node6);
```

Las *swim lanes* en este caso están diseñadas con rectángulos, sin tener ningún tipo de implementación especial que les de cierta funcionalidad. En este caso las tres *swim lanes* se definen con el siguiente código, que crea los tres rectángulos y los añade al modelo:

```
//Creation of the Layers
Syncfusion.Windows.Forms.Diagram.Rectangle Layer1 = createRectangle(0, 0);
Syncfusion.Windows.Forms.Diagram.Rectangle Layer2 = createRectangle(0, 198);
Syncfusion.Windows.Forms.Diagram.Rectangle Layer3 = createRectangle(0, 396);

//Appending the Layers on to the Diagram.Model
this.diagram1.Model.AppendChild(Layer1);
this.diagram1.Model.AppendChild(Layer2);
this.diagram1.Model.AppendChild(Layer3);
```

La creación de conexiones entre nodos también es considerablemente más compleja que en el caso de AddFlow, permitiéndonos personalizar desde el tipo de flecha, el color de la misma o el orden en el eje Z. Igual que en los casos anteriores hay que añadir todos los nuevos elementos al modelo.

```
private void ConnectNodes(Node parentNode, Node childNode)
{
    if (parentNode != null && childNode != null
        && parentNode.CentralPort != null && childNode.CentralPort != null)
    {
        LineConnector connector = new LineConnector(PointF.Empty,
            new PointF(0, 1));
        connector.HeadDecorator.DecoratorShape = DecoratorShape.Filled45Arrow;
        connector.LineStyle.LineColor = Color.Black;

        this.diagram1.Model.AppendChild(connector);
        this.diagram1.Model.SetZOrder(connector, 1);
        parentNode.CentralPort.TryConnect(connector.TailEndPoint);
        childNode.CentralPort.TryConnect(connector.HeadEndPoint);
    }
}
```

Syncfusion también tiene a su favor la existencia de un completo editor de diagramas, que se muestra en la Figura 28, similar a Microsoft Visio que ofrece un explorador de documentos, un editor de propiedades similar a Visual Studio, funciones de layout respecto a una cuadrícula, herramientas de zoom y permite acciones de deshacer y rehacer.

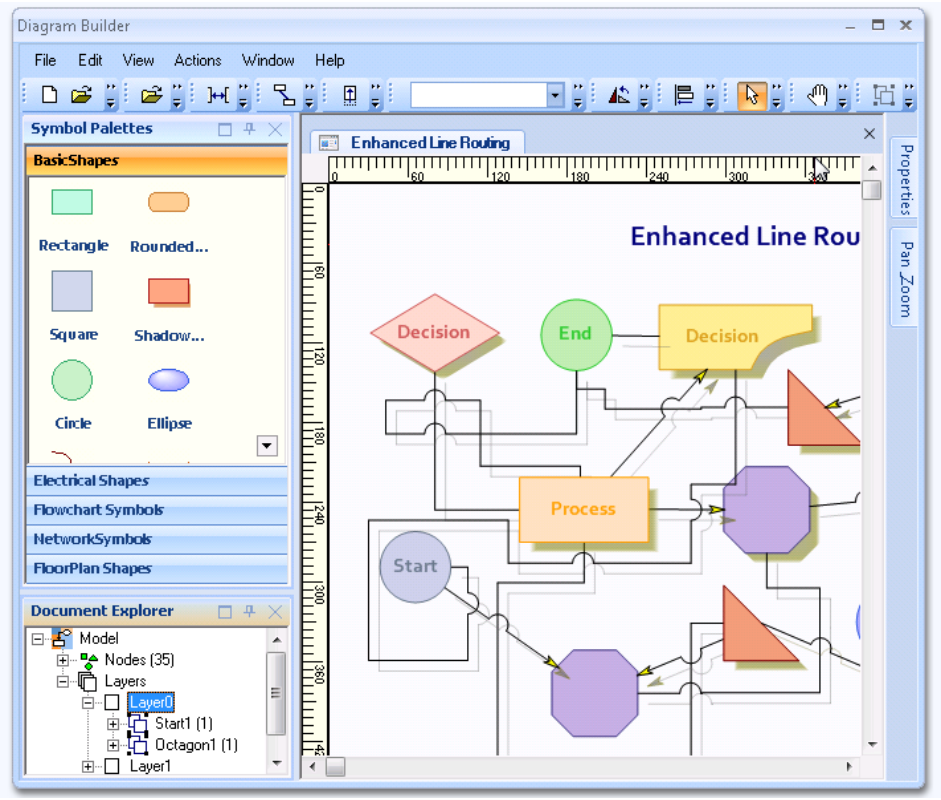


Figura 28 Editor de diagramas incluido en los ejemplos de Syncfusion

Para el problema que nos atañe es un editor excesivamente complejo que permite demasiada personalización a nivel de interfaz. En resumen podemos decir que toda la flexibilidad extra que proporciona se convierte en complejidad añadida sin ser realmente necesaria para el proyecto.

4.2.3 Nevron Diagram

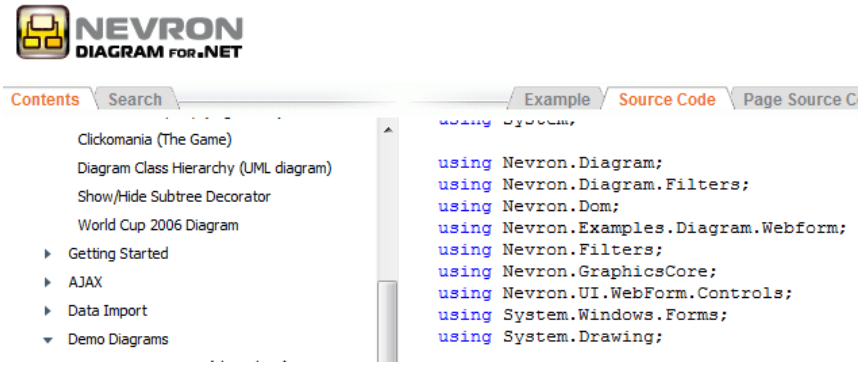


Figura 29 Documentación online de Nevron

Software de tipo comercial que incluye controles para ASP.NET tanto con AJAX como con ActiveX y gestión de layout [28]. Su precio es de 589$. En su web tiene una extensa documentación con ejemplos y código como se puede ver en la Figura 29.

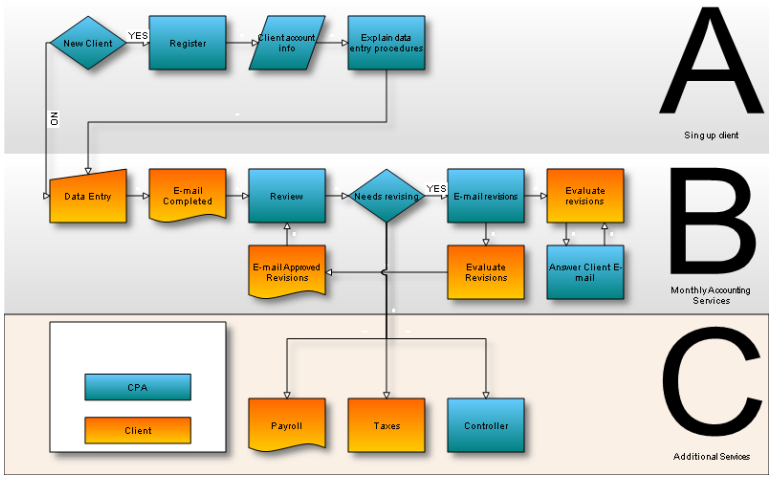


Figura 30 Ejemplo de diagramas de Nevron

Como en el caso de Syncfusion, las *swim lanes* carecen de ningún tipo de implementación, siendo simplemente rectángulos en el fondo de la imagen. En

el código se puede ver como crea un objeto *NRectanglePath*, le asigna un nombre y lo añade a la capa activa del documento para cada una de las *swim lanes*, llamadas *stripes* en este caso.

```
// create the stripes
NRectanglePath rect = new NRectanglePath(0, 0, document.Width, document.Height / 3);
rect.StyleSheetName = "STRIPE";
document.ActiveLayer.AddChild(rect);

rect = new NRectanglePath(0, document.Height / 3, document.Width, document.Height / 3);
rect.StyleSheetName = "STRIPE";
document.ActiveLayer.AddChild(rect);

rect = new NRectanglePath(0, 2 * document.Height / 3, document.Width, document.Height / 3);
rect.StyleSheetName = "STRIPE";
document.ActiveLayer.AddChild(rect);
```

Para la creación de nodos utilizamos los métodos de la clase padre, que es la que define el diagrama y todas sus funciones y atributos. En el siguiente ejemplo se realiza la instanciación de los nodos de la primera *swim lane*, pasando como parámetro el documento al que pertenece, el tipo de figura, la posición, el título y el estilo del nodo.

```
// shapes in row 1
NShape newClient = base.CreateFlowChartingShape(document,
    FlowChartingShapes.Decision, base.GetGridCell(0, 0), "New Client", "CPA");
NShape register = base.CreateFlowChartingShape(document,
    FlowChartingShapes.Process, base.GetGridCell(0, 1), "Register", "CPA");
NShape clientAccountInfo = base.CreateFlowChartingShape(document,
    FlowChartingShapes.Data, base.GetGridCell(0, 2), "Client account info", "CPA");
NShape explainDataEntryProcedures = base.CreateFlowChartingShape(document,
    FlowChartingShapes.Process, base.GetGridCell(0, 3), "Explain data entry procedures", "CPA");
```

Y en el siguiente ejemplo define las conexiones entre los nodos creando conectores. Para ello le pasa por parámetros el documento al que pertenece, el nodo de salida, la posición, el nodo de llegada y el tipo de conector.

```
// connect shapes in levels
base.CreateConnector(document, newClient, "Center", register, "Center", ConnectorType
base.CreateConnector(document, register, "Center", clientAccountInfo, "Center", Conne
base.CreateConnector(document, clientAccountInfo, "Center", explainDataEntryProcedure
```

Los componentes de Nevron vienen con un editor de ejemplo muy potente, como se puede ver en la Figura31, bastante similar al que ofrece Syncfusion. Este incluye una completa gestión de librerías de dibujos para diagramas,

herramientas de zoom, dibujo y texto e implementa las funciones "deshacer/rehacer".

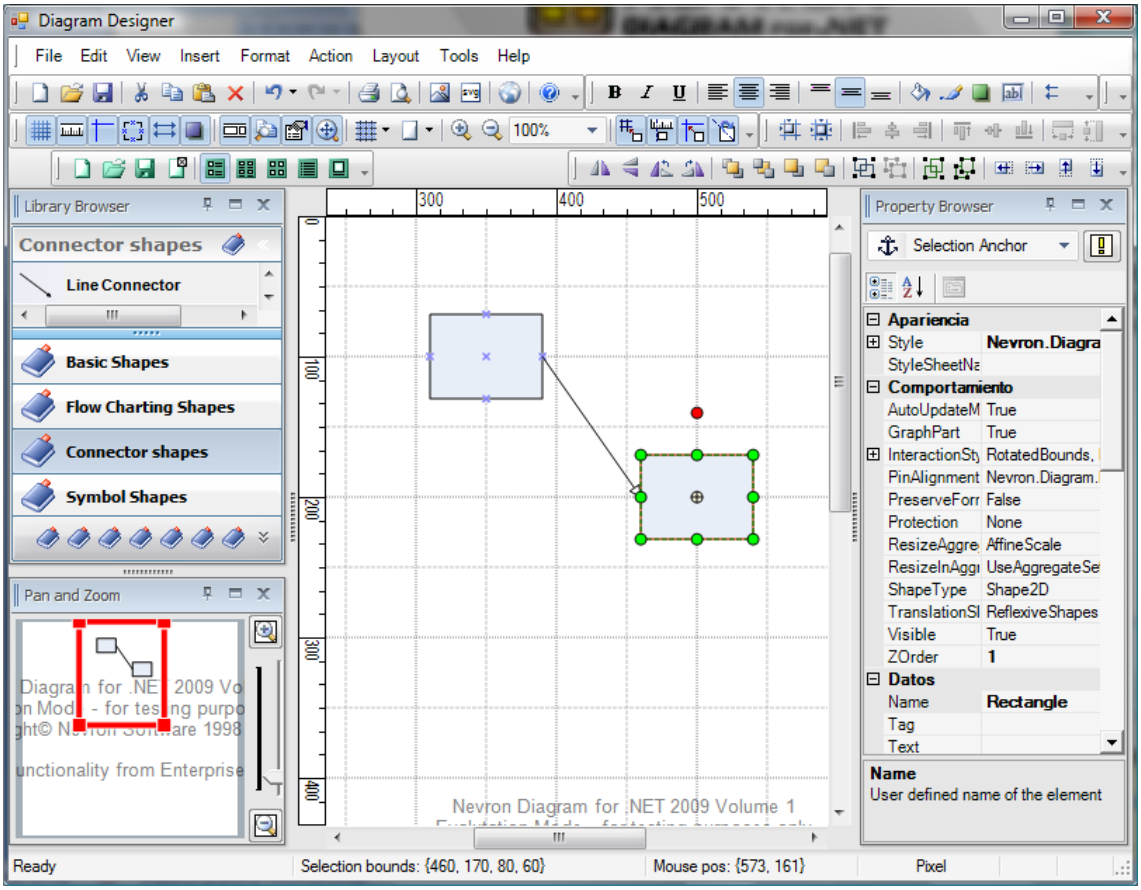


Figura 31 Editor de diagramas de Nevron

Los componentes de Nevron comparten los pros y contras de Syncfusion, una herramienta muy potente y con extensa documentación pero excesiva para el problema que nos atañe. Comparativamente con Syncfusion este componente es algo inferior en cuanto a documentación.

4.2.4 GoDiagram

Software comercial de la empresa Northwoods Software cuya mayor ventaja es la existencia de una versión reducida por 199$ (la versión completa alcanza los 899$), la cual es suficiente para la edición de diagramas simples [14]. La funcionalidad básica permite construir diagramas con nodos y conexiones, dejando fuera la gestión de layout y el soporte para *swim lanes*.

Los componentes GoDiagram tienen una API más simple que Syncfusion o Nevron pero no tanto como AddFlow. Parte de esa simplicidad viene por la mayor abstracción de elementos, la cual se puede ver en el editor de diagramas que incluye entre sus ejemplos, el cual está representado en la Figura 32.

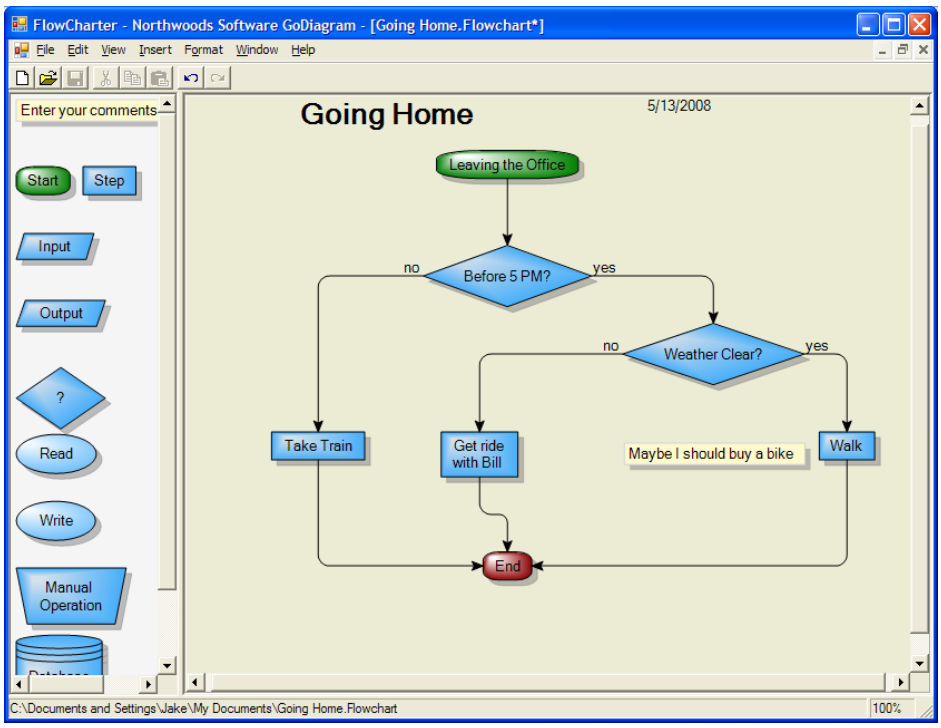


Figura 32 Editor de ejemplo incluido en GoDiagram

Los nodos están definidos en diferentes clases dependiendo de su tipo, lo que implica una simplificación en los parámetros a definir en su creación. Además de que están más enfocados a su tratamiento como nodos en vez de figuras

geométricas. En el siguiente ejemplo podemos ver el constructor de la clase *cActividad*, la cual hereda de la clase *GraphNode* (la que define el comportamiento de los nodos en el diagrama).

```
public cActividad()
    : base(GraphNodeKind.Step)
{
    this.TopPort = null;
    this.BottomPort = null;
    this.LeftPort.IsValidTo = true;
    this.LeftPort.IsValidFrom = false;
    this.RightPort.IsValidTo = false;
    this.RightPort.IsValidFrom = true;

    this.Label.Editable = false;
    this.activa = true;
}
```

Para definir la presentación del nodo únicamente hay que especificar el tipo de nodo a crear, el cual es de tipo “Step”, para luego pasar a configurar el comportamiento del mismo. En este caso se define el comportamiento de los puertos del nodo, que son los cuatro puntos de conexión que tiene un nodo para conectarse con otro, especificando que únicamente el puerto de la derecha sea de salida y el de la izquierda de entrada.

La abstracción del diseño ayuda a centrarse en la implementación de las funciones de la herramienta, pudiendo trabajar sobre el editor y aprovechar su gestión y edición de diagramas. Comparativamente con los otros editores el que ofrece GoDiagram es mucho más simple y más acorde con las necesidades de la aplicación a desarrollar.

La documentación no es tan completa como la que ofrece Syncfusion o Nevron, ya que no hay una API online si no que son documentos PDF además de los foros, pero los ejemplos son suficientes para poder desarrollar sin mayor problema.

4.2.5 Netron

Software de código abierto [27] en un estado avanzado de desarrollo, pero finalmente fue abandonado. Se creó un nuevo proyecto como continuación llamado Netron Reloaded, pero no tiene un ritmo constante de actualizaciones. Esto hace que tenga un soporte nulo como contrapartida de los beneficios de que sea código abierto. Como documentación incluye un IDE de diagramas, mostrado en la Figura 33, que sirve como ejemplo de la utilización de la librería, pero no es estable e incluso supuso un problema conseguir su correcto funcionamiento.

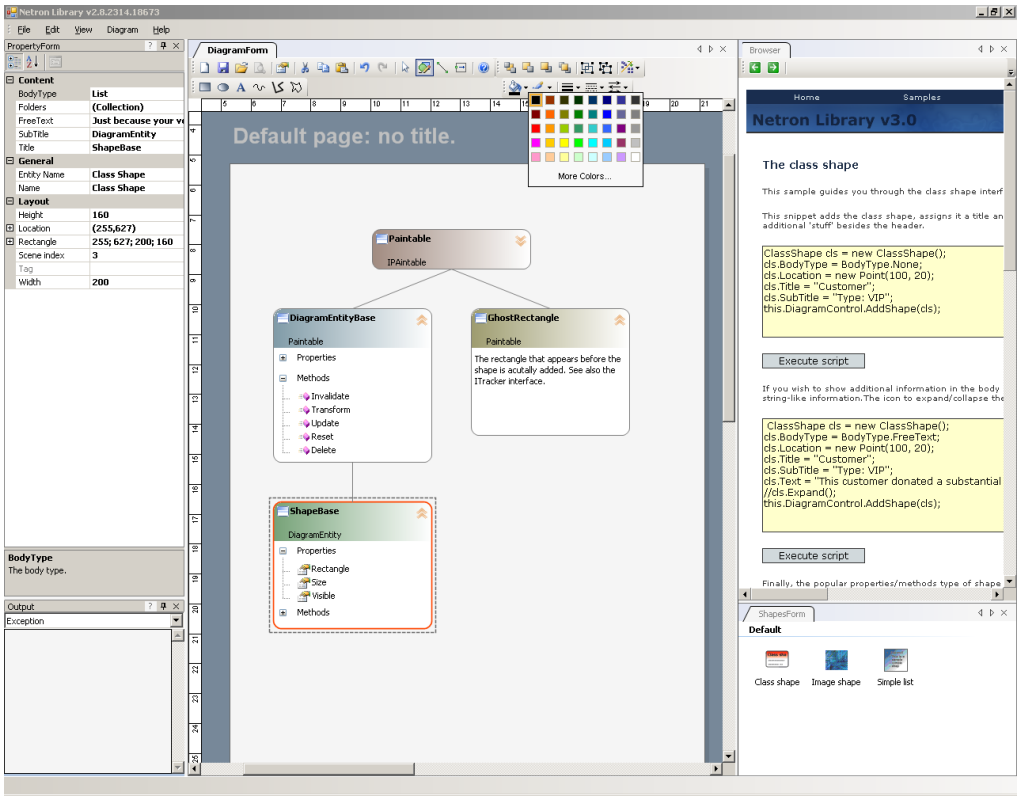


Figura 33 IDE incluido en Netron

La empresa que lo aloja, "The Orbifold", ha orientado el resto de sus productos a Silverlight y WPF, dejando de lado los programas de diagramas en WinForms. Ante la falta de soporte, ejemplos y documentación se descartó desde un principio.

4.2.6 Open Diagram

Software de código abierto licenciado bajo LGPL [32] que peca de los mismos fallos que Netron, la falta de documentación e inestabilidad del producto. Este componente no incluye gestión de layout y ofrece ejemplos muy básicos.

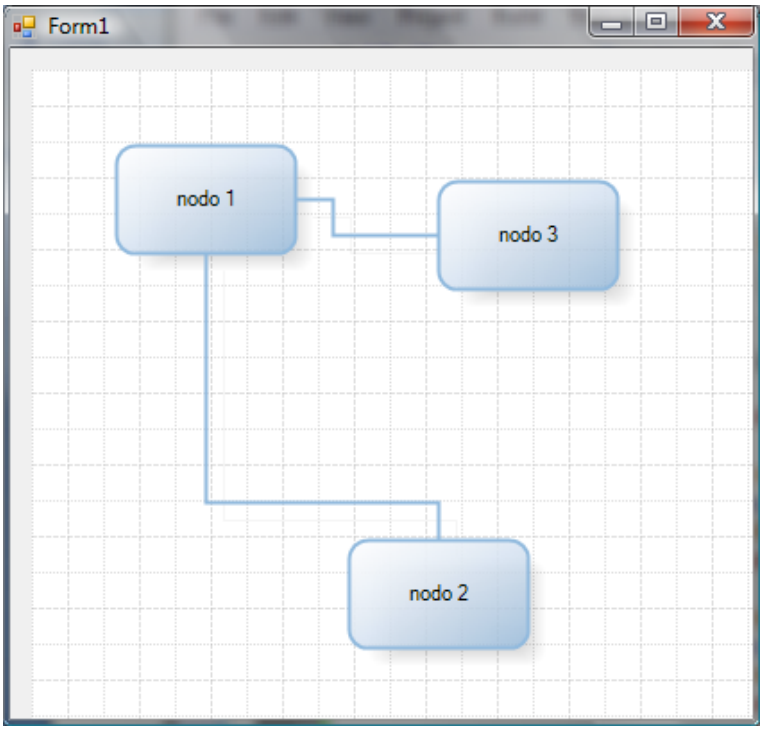


Figura 34 Ejemplo de diagramas de Open Diagram

Dentro de las carencias de Open Diagram está la gestión de documentos. Las actividades o figuras se añaden a un objeto de tipo *Model*, el cual únicamente tiene funciones de contenedor de formas. Esto implica que sería necesario implementar toda la gestión de documentos, e incluso funciones de salvado de los diagramas. Además, la variedad de nodos es escasa, con lo que también sería necesario implementarlos específicamente.

En el siguiente ejemplo se muestra como al modelo del diagrama se le añade una figura y un conector, se le aplica un tema para su visualización y finalmente se refresca el diagrama para que se muestre el resultado.

```
Model model = diagram1.Model;

Shape shape = new Shape();
shape.Location = new PointF(170, 80);
shape.Label = new TextLabel("prueba");
model.Shapes.Add(shape);

Connector connector = new Connector();
connector.Start.Location = new PointF(20, 20);
connector.End.Location = new PointF(440, 330); //300,140
connector.Avoid = true;

model.Lines.Add(connector);

model.ApplyTheme(Themes.LightBlue);

diagram1.Controller.Refresh();
```

La presentación destaca sobre la media pero sigue siendo un producto incompleto con el que se hace difícil trabajar. Además las actualizaciones que recibe son mínimas, siendo la última versión de marzo de 2009.

4.2.7 yFiles

Software comercial que consta de una completa suite de componentes para tratamiento de gráficos [41]. Incluye gestión de layout, exportación de diagramas a múltiples formatos, una herramienta de análisis de diagramas mediante algoritmos y un potente editor.

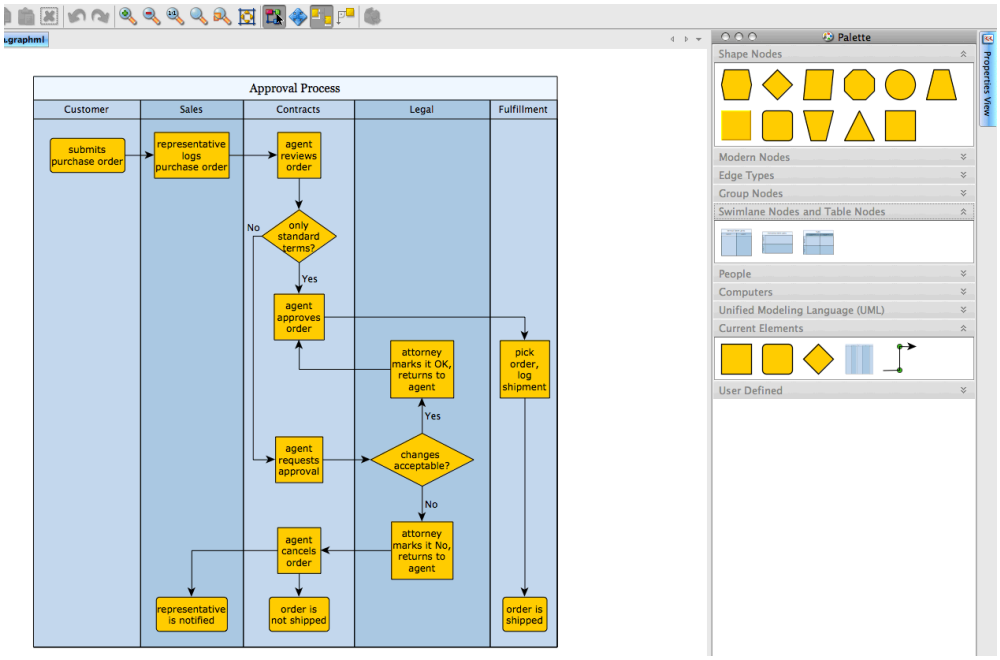


Figura 35 Editor de diagramas de yFiles

Con diferencia es el producto más completo y a su vez el de mayor precio (3600€) de todos y por ello está muy por encima de las necesidades del proyecto y del presupuesto. Por ese motivo no se profundizó más en este componente.

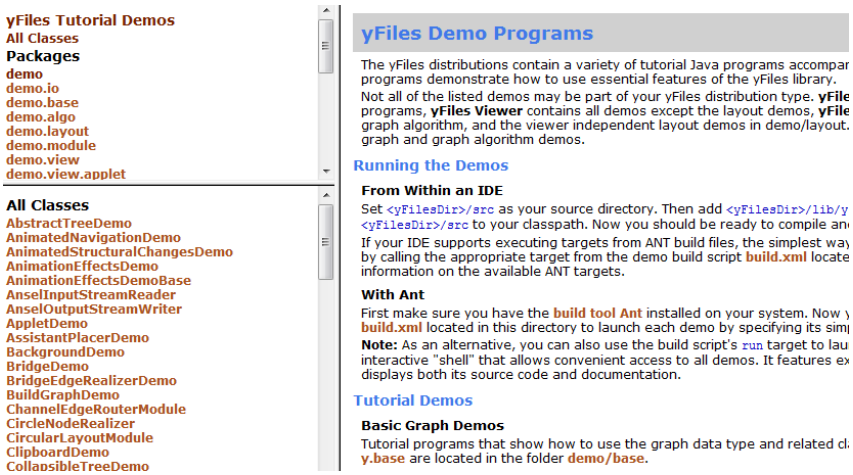


Figura 36 Documentación online de yFiles

4.3 Conclusiones

Como primera elección había que considerar la utilización de un componente comercial o uno de software libre. En este caso, adquirir un software comercial suponía tener un soporte y sobre todo una fiabilidad de la cual carecían todas las opciones de tipo software libre. Esto se ponía de manifiesto en la calidad de la documentación de los distintos componentes, siendo casi inexistente en el caso de los componentes de tipo software libre.

Los criterios de elección cambiaron durante las distintas pruebas de los componentes, dejando las *swim lanes* y la parte web fuera de los requisitos. El problema de las *swim lanes* es que no existía una implementación en ninguno de los componentes y en el caso de la parte web es que la compatibilidad entre Web y aplicación de escritorio no era total, lo que implicaba duplicar la implementación. La gestión de layout también se descartó porque la complejidad que añadía no suponía un gran beneficio.

Después de revisar y probar todos estos componentes se llegó a las siguientes conclusiones:

- Resultaba más interesante elegir una opción de tipo comercial sobre todo por la documentación.
- La herramienta elegida debía ser concisa, evitando así tener más funcionalidad de la necesaria.
- Los ejemplos suministrados debían ser suficientemente complejos como para poder utilizarlos como base.

Siguiendo estos criterios las opciones se reducían a AddFlow, Syncfusion y GoDiagram, dejando fuera las opciones de software libre. Las tres cumplían los requisitos de documentación, estabilidad, numerosos ejemplos y estaban dentro del presupuesto. Aún así no era necesaria tanta funcionalidad como la

que nos permitían. Por ello el primer descarte fue Syncfusion por ser la opción más completa de las tres.

Finalmente la opción elegida fue GoDiagram, ya que ofrece una versión "express" con funcionalidad reducida, incluye ejemplos similares al problema a resolver y tiene una API potente y sencilla.

Capítulo 5: Módulo para especificación de Workflows

5.1 Especificación de requisitos

5.1.1 Propósito.

El propósito del proyecto es la creación de un módulo de gestión visual de workflows. En este apartado se pretende definir cuáles son los requerimientos que debe tener la nueva aplicación, teniendo en cuenta que debe orientarse en TUNE-UP Process Tool.

5.1.2 Descripción del sistema actual

El sistema al que se le va a añadir el módulo es TUNE-UP Process Tool, cuyo núcleo es un motor de workflows el cual se encarga del correcto orden de las tareas, el flujo de las incidencias, la asignación de los recursos, etc.

La interacción con los propios workflows se divide en dos apartados: consulta y edición. En el caso de la consulta, en el sistema actual se realiza mediante unos archivos generados mediante Microsoft Visio para la visualización de los workflows desde el programa de gestión de tareas. La edición a su vez carece de ningún tipo de interfaz, teniendo que gestionar manualmente las tablas de la base de datos con la complejidad y los posibles errores que eso supone.

Los workflows están definidos en la base de datos por las siguientes tablas:

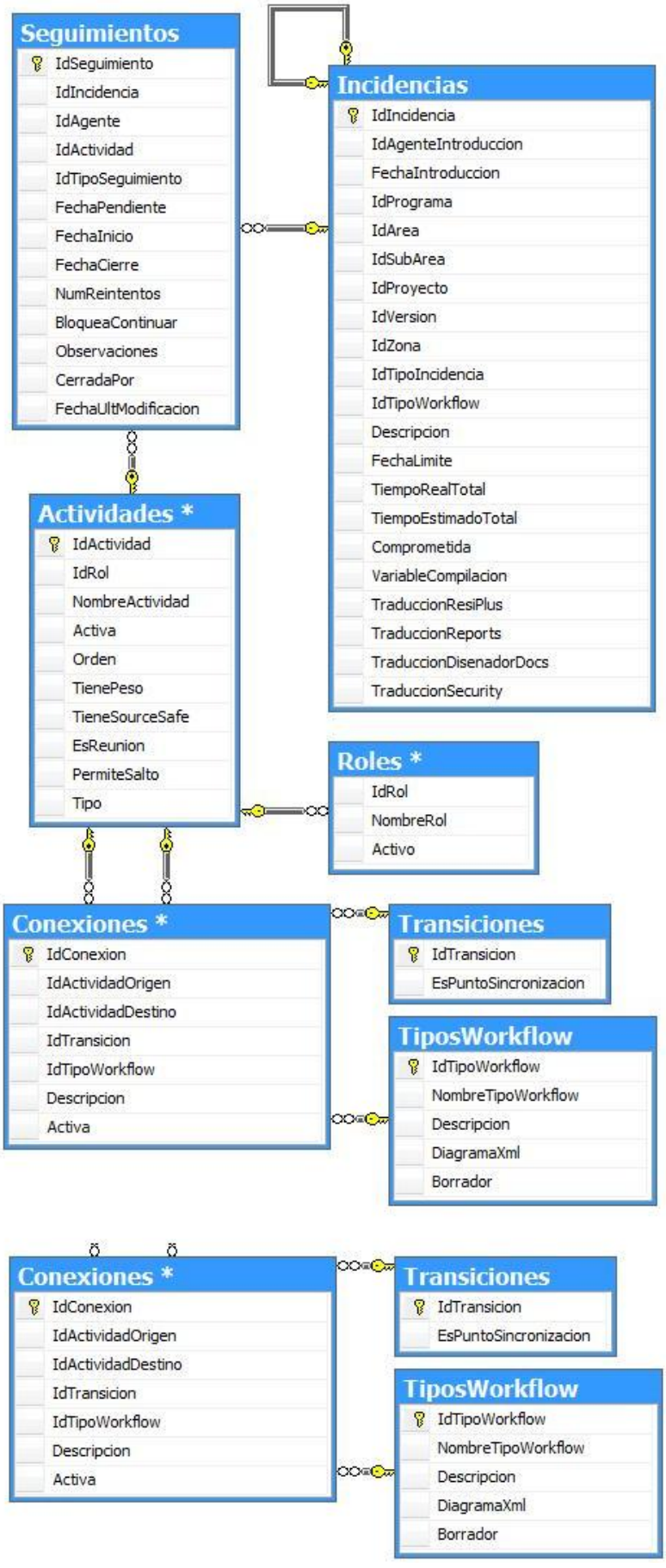


Figura 37 Diagrama relacional de la base de datos

En los workflows las actividades están unidas por transiciones, que a su vez están compuestas por conexiones. Dependiendo de si dos conexiones son parte de una misma transición, de si las conexiones tienen el mismo origen o destino y de si la transición está sincronizada es como se diferencian los distintos tipos de transiciones. Existen dos tipos:

- Decisión: cuando una unidad de trabajo puede tomar varios caminos o cuando dos caminos se unen.
- Paralelo: cuando una unidad de trabajo se divide, realizándose a la vez dos caminos distintos. A su vez también sirve para volver a unirlos.

Los dos tipos de transiciones se representan en la base de datos de distinta manera dependiendo de si es de entrada, de salida o de unión de transiciones. El convenio de representación de los workflows queda reflejado en la base de datos de la siguiente manera:

- Decisión de salida: conexiones de transiciones diferentes que tienen el mismo origen.
- Decisión de llegada: conexiones de una misma transición con mismo destino sin sincronización.
- Unión de decisiones: conexiones con mismo destino de transiciones diferentes.
- Paralelo de salida: conexiones con mismo origen y de una misma transición sin sincronización.
- Paralelo de llegada: conexiones con mismo destino y de una misma transición con sincronización.

Nótese en el diagrama relacional anterior la inclusión de los campos DiagramaXml y Borrador en la tabla TiposWorkflow, cuya finalidad es guardar los diagramas generados por la nueva aplicación.

5.2 Arquitectura propuesta

El módulo tiene como finalidad permitir la modificación de los workflows desde un entorno visual pero a su vez es necesario implementar una solución que cubra la visualización de los mismos. Por ello se desarrolló un visor de workflows que sustituyera el sistema de imágenes anterior incluido en la aplicación. Además, el visor debía ofrecer información tanto de las actividades como del propio workflow.

5.2.1 Casos de uso

Siguiendo la descripción anterior del sistema el diagrama de casos de uso sería el siguiente:

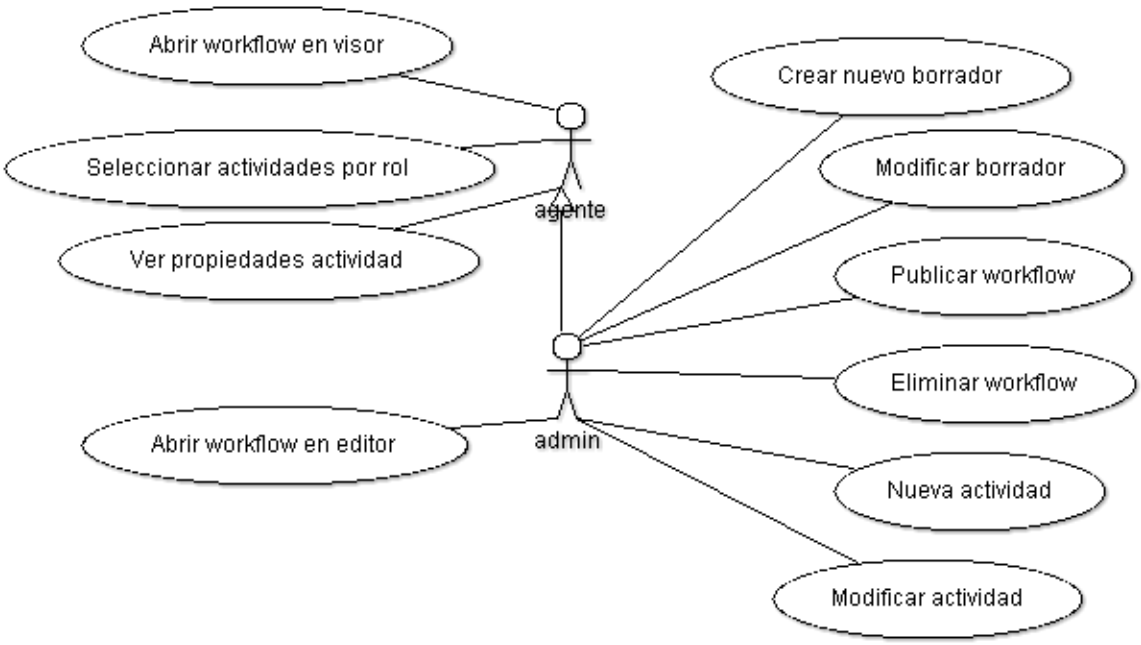


Figura 38 Diagrama de casos de uso

Los casos de uso se dividen entre las dos aplicaciones que van a utilizar los dos tipos de usuario, por un lado está el agente, el cual es el usuario del sistema que accede para visualizar los workflows desde la aplicación de gestión de tareas y por el otro está el administrador, encargado de la gestión de los workflows.

Visor

Los casos de uso de la parte del visor son los siguientes:

Caso de uso 01 – Abrir workflow en visor

Desde TUNE-UP Process Tool el usuario accede al visor de workflows desde el menú de ayuda, dónde se muestra un listado de los workflows publicados del sistema.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Debe permitir abrir el borrador de un workflow si el workflow está publicado y tiene un borrador.
- Debe abrir el workflow que se le pasa como parámetro y en caso de no especificarlo que nos muestre la lista completa de los workflows que tengan diagrama.
- Comprobar que las actividades que no están activas se muestran de otro color.
- Comprobar que no podemos modificar ningún elemento del diagrama.

Caso de uso 02 – Seleccionar actividades por rol en visor

En el visor, mediante el desplegable de roles, el usuario puede consultar las actividades que pertenecen a cada rol. Éstas se le mostrarán en otro color para resaltar cuáles son las seleccionadas.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que el listado de roles son todos los que están en el workflow.
- Comprobar que al seleccionar un rol se colorean todas las actividades del workflow que tienen ese tipo de rol.

Caso de uso 03 – Ver propiedades de una actividad

En el visor el usuario puede acceder a las propiedades de una actividad haciendo doble clic sobre ella.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que al hacer doble clic en una actividad se abre la ventana de propiedades de la actividad.
- Comprobar que no podemos modificar las propiedades de una actividad desde la ventana de propiedades de la actividad.

Editor

Los casos de uso de la parte del editor son los siguientes:

Caso de uso 04 – Abrir workflow en editor

Dentro del editor el administrador puede abrir un workflow del listado de workflows, el cual incluye workflows publicados y borradores.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Debe preguntar si queremos ver el borrador en caso de que el workflow tenga uno.
- Comprobar que muestra un cartel en el que pone "Borrador" cuando el workflow abierto es un borrador.
- Comprobar que las actividades que no están activas se muestran de otro color.

Caso de uso 05 – Crear nuevo borrador

El usuario podrá empezar un workflow desde cero y guardarlo. Al no estar publicado queda registrado como borrador.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que se añade el nuevo borrador al listado de workflows.
- Comprobar que el nuevo borrador se guarda como borrador y no como workflow publicado.
- Comprobar que no se puede guardar un borrador sin nombre.

Caso de uso 06 – Modificar workflow

El usuario puede abrir cualquiera de los workflows del sistema y modificarlo, puede ser un borrador o un workflow ya publicado, en este caso añadiéndole un borrador al mismo. Si el workflow estaba publicado comprobará que no se han eliminado actividades que tenían incidencias.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que no se guardan datos de los elementos del workflow hasta que no se publique el borrador.
- Comprobar que al cambiar el nombre del workflow no es posible dejarlo vacío.
- Comprobar que no es posible guardar un workflow con nombre duplicado.
- Comprobar que no se pueden modificar actividades que tienen asociadas una incidencia.

Caso de uso 07 – Publicar workflow

El usuario abre un workflow que tenga un borrador asociado y cambia su estado a publicado mediante la opción “publicar” del menú. Si se modifican actividades el sistema advertirá que el resto de diagramas de workflows que contienen esa actividad serán eliminados. El workflow quedará actualizado, guardando el XML generado en el campo DiagramaXml y el campo Borrador quedará vacío. A su vez se generarán todas las entradas de conexiones, transiciones y actividades en las tablas correspondientes.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que no se pueden publicar si se modifican actividades que tienen asociadas una incidencia.
- Comprobar que no se puede publicar si existen nodos sin conectar en el workflow.
- Comprobar que no se puede publicar un workflow con transiciones con varias entradas y salidas.
- Comprobar que no se puede publicar un workflow con varias conexiones entre actividades.
- Comprobar que no pueden conectarse dos nodos transición del mismo tipo.
- Comprobar que si se modifica una actividad se eliminan los diagramas de los workflows que contienen esa actividad.

Caso de uso 08 – Eliminar workflow

El usuario puede eliminar un workflow del sistema desde el menú siempre que no tenga incidencias asociadas. El workflow quedará eliminado y en caso de estar publicado también se eliminarán las conexiones y transiciones del mismo.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que no se puede eliminar un workflow con incidencias asociadas.
- Comprobar que se eliminan el workflow del listado.

Caso de uso 09 – Nueva actividad

Desde la ventana de propiedades de la actividad, es posible añadir actividades al listado mediante el botón "nueva". Se crea una nueva actividad y ésta está dentro del borrador modificado. No quedará reflejado en la base de datos hasta que se publique.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que la actividad se añade al listado de actividades.
- Comprobar que no es posible crear una actividad con nombre duplicado.
- Comprobar que no se quede vacío el campo nombre.

Caso de uso 10 – Modificar actividad

Desde la ventana de propiedades de actividad se puede modificar las propiedades de las distintas actividades registradas en el sistema. Pero supone eliminar los diagramas de los workflows de la base de datos que contengan dicha actividad. La actividad es modificada dentro del borrador modificado. No quedará reflejado en la base de datos hasta que se publique. Los diagramas de los workflows que tengan esa actividad serán eliminados.

El caso de uso debe cumplir las siguientes pruebas de aceptación:

- Comprobar que no se quede vacío el campo nombre.
- Comprobar que no se puede modificar el rol.
- Comprobar que al cancelar no realice ninguna modificación.

5.3 Diseño

5.3.1 Introducción

En este apartado se explicará el análisis y diseño correspondiente al módulo. En esta fase el objetivo es satisfacer los requisitos especificados en el modelado de requisitos y establecer una arquitectura interna que les dé soporte.

A continuación, se especificará la dinámica del sistema para realizar los casos de uso del modelo de requisitos. De este modo, finalmente, de cada una de las partes describiremos:

- Clases existentes en el sistema.
- Relaciones entre las clases.

5.3.2 Diagrama de clases

En este apartado se establece el modelo estático del sistema. Este modelo contiene las clases que formarán parte del sistema así como sus estructuras y las relaciones existentes entre ellas.

En el diagrama de la Figura 39 se pueden observar las múltiples clases y sus relaciones.

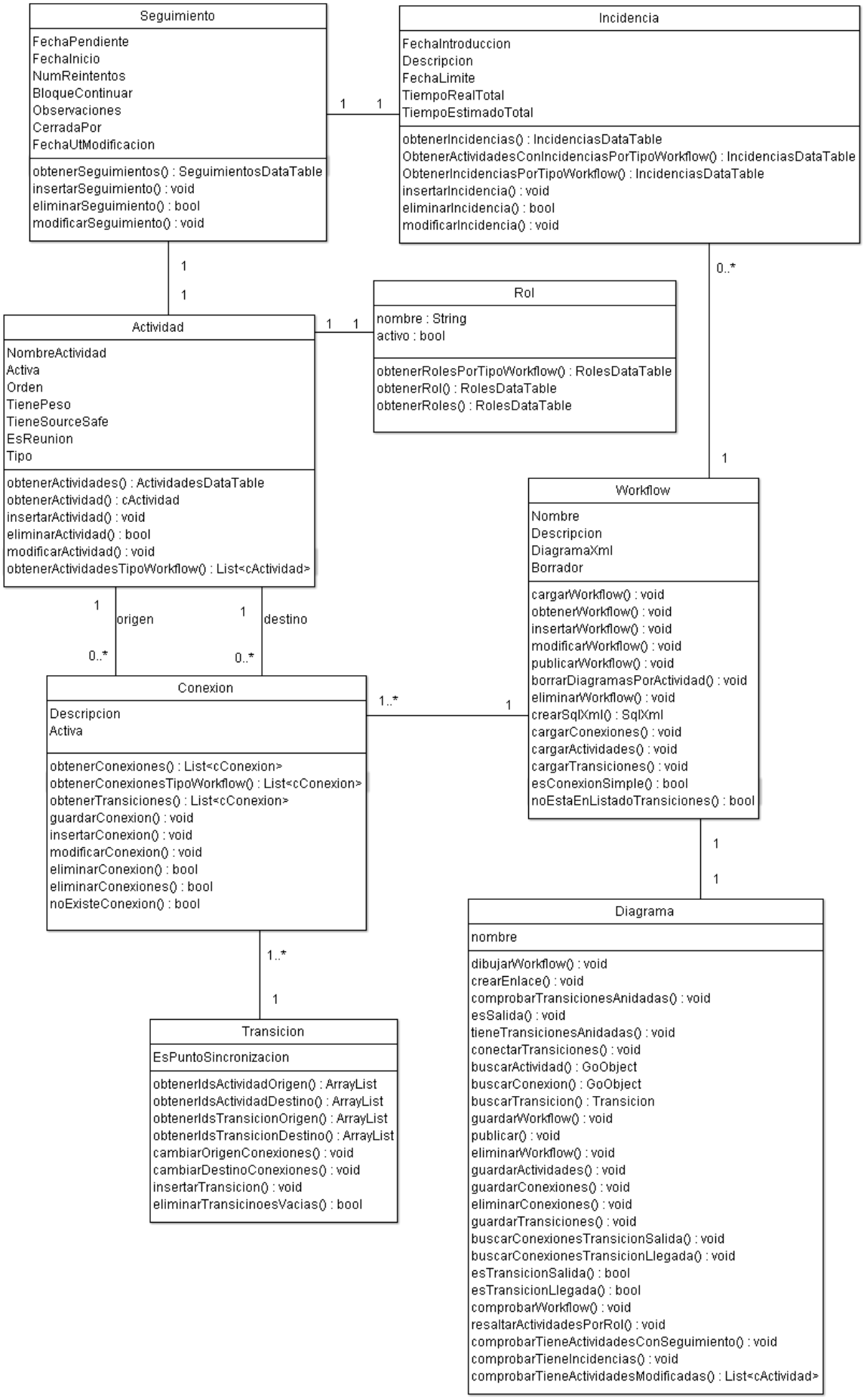


Figura 39 Diagrama de clases

La clase principal es el workflow, ya que todo el sistema girará en torno a él. Sobre él se podrán realizar múltiples acciones y el resto de clases dependerán de él.

Por otra parte están las actividades, las cuales se definen como los distintos estados por los que pasarán las unidades de trabajo. Las actividades a su vez están unidas por transiciones, que pueden ser de tipo decisión, paralelo o conexión y están compuestas de conexiones.

Las incidencias y los seguimientos no repercuten en el sistema de forma directa aunque en muchos casos dependa la modificación de un workflow de si existen incidencias asociadas.

5.3.3 Arquitectura de la aplicación

El mayor reto del módulo es la correspondencia gráfica de los workflows con la base de datos, teniendo que realizar transformaciones para convertir los datos en diagramas y viceversa. Para ello fue necesario crear una capa de abstracción de los datos y que a su vez se comunicara con la capa que se encarga de los diagramas basada en los componentes de GoDiagram.

Los componentes GoDiagram ofrecen un editor simple sobre el que implementar el módulo. Tomando como base la clase `GoDocument`, que se encarga de todo lo relacionado con la edición del documento, se construye la clase workflow. En el caso de las actividades hereda de `GraphNode`, la clase que representa los nodos en los diagramas, y para las conexiones heredan de `GraphLink`, la clase encargada de representar las conexiones entre nodos.

A continuación se van a comentar dos ejemplos de la implementación que fueron especialmente relevantes, al ser fundamentales para la correspondencia entre la representación gráfica y los datos de los workflows.

El primer ejemplo trata sobre la carga de workflows desde la base de datos cuando aún no existe un diagrama y el segundo sobre la publicación de diagramas, en cuyo caso se produce el volcado del diagrama a las tablas de la base de datos.

Cargar workflow sin diagrama

Cuando un workflow no tiene el diagrama creado, el sistema lo genera convirtiendo las tablas en objetos del diagrama. Para ello se instancia la clase `Diagrama` pasando como parámetro el identificador del workflow a mostrar almacenado en la base de datos. La case `Diagrama` se compone mayormente de un objeto que guarda el workflow representado en objetos y de un objeto que representa el documento del diagrama. En el constructor siguiente se muestra que se instancia un objeto de tipo `cWorkflow` pasando el identificador del workflow.

```
public Diagrama(int TipoWorkflow)
    : base()
{
    workflow = new cWorkflow(TipoWorkflow);
    this.Document.Name = workflow.Nombre;
    this.Document.IsModified = false;
}
```

En el constructor de la clase `cWorkflow` es donde se cargan los valores de la base de datos, rellenando las listas de actividades, transiciones y conexiones.

```
public cWorkflow(int TipoWorkflow)
{
    inicializarVariables();
    cargarWorkflow(TipoWorkflow);
}

private void cargarWorkflow(int TipoWorkflow)
{
    this.tipoWorkflow = TipoWorkflow;
    obtenerWorkflow(TipoWorkflow);
    cargarConexiones();
    cargarActividades();
    cargarTransiciones();
}
```

La carga de actividades y conexiones es trivial, pero en el caso de las transiciones es cuando hay que comprobar que se representen adecuadamente dependiendo de las combinaciones de las conexiones y que las agrupe adecuadamente. El método que se encarga de ello la siguiente forma:

```
private void cargarTransiciones()
{
    List<cConexion> conexionesTransicion =
        new cConexion().obtenerTransiciones(TipoWorkflow);
    cTransicion transParalelo, transDecision;

    //primero los paralelos y después las decisiones
    //para cada conexion comprueba cuantas son de su misma transición y las agrupa

    //Cargamos los paralelos
    foreach (cConexion conexion in conexionesTransicion)
    {
        paralelos
    }

    //Cargamos las decisiones
    foreach (cConexion conexion in conexionesTransicion)
    {
        decisiones
    }
}
```

Para las transiciones de tipo paralelo el algoritmo busca todas las conexiones distintas de una misma transición, con mismo origen o con mismo destino y que son punto de sincronización, para agruparlas en transiciones de tipo paralelo.

```
transParalelo = null;

//comprueba que no se ha metido ya la transición
if (noEstaEnListadoTransiciones(conexion))
{
    foreach (cConexion conexion2 in conexionesTransicion)
    {
        //Comprueba que no sea la misma conexión
        if (conexion.IdConexion != conexion2.IdConexion
            && noEstaEnListadoTransiciones(conexion2))
        {
            //Agrupa por transición
            if (conexion.IdTransicion == conexion2.IdTransicion &&
                (conexion.IdActividadOrigen == conexion2.IdActividadOrigen ||
                (conexion.IdActividadDestino == conexion2.IdActividadDestino &&
                conexion.EsPuntoSincronizacion == conexion2.EsPuntoSincronizacion == true)))
            {
                if (transParalelo == null)
                {
                    transParalelo = new cTransicion(GraphNodeKind.Paralelo);
                    transParalelo.IdTransicion = conexion.IdTransicion;
                    transParalelo.EsPuntoSincronizacion = conexion.EsPuntoSincronizacion;
                    transParalelo.Conexiones.Add(conexion);
                }
                transParalelo.Conexiones.Add(conexion2);
            }
        }
    }

    if (transParalelo != null)
        transiciones.Add(transParalelo);
}
```

En el caso de las transiciones de tipo decisión el algoritmo es similar pero las condiciones para agrupar conexiones en transiciones son: que tenga el mismo origen y sean de distinta transición o que tengan el mismo destino y sean de la misma transición.

```
transDecision = null;

//comprueba que no se ha metido ya la transición
if (noEstaEnListadoTransiciones(conexion))
{
    foreach (cConexion conexion2 in conexionesTransicion)
    {
        //Comprueba que no sea la misma conexión
        if (conexion.IdConexion != conexion2.IdConexion
            && noEstaEnListadoTransiciones(conexion2))
        {
            //Agrupa por transición
            if ((conexion.IdActividadOrigen == conexion2.IdActividadOrigen &&
                conexion.IdTransicion != conexion2.IdTransicion) ||
                (conexion.IdActividadDestino == conexion2.IdActividadDestino &&
                conexion.IdTransicion == conexion2.IdTransicion))
            {
                if (transDecision == null)
                {
                    transDecision = new cTransicion(GraphNodeKind.Decision);
                    transDecision.IdTransicion = conexion2.IdTransicion;
                    transDecision.EsPuntoSincronizacion = conexion.EsPuntoSincronizacion;
                    transDecision.Conexiones.Add(conexion);
                }
                transDecision.Conexiones.Add(conexion2);
            }
        }
    }
```

Si no hay dos conexiones dentro de la misma transición, crea una transición de una única conexión, para luego añadirla al listado de transiciones del workflow.

```
    if (transDecision == null
        && !esConexionSimple(conexion, conexionesTransicion))
    {
        transDecision = new cTransicion(GraphNodeKind.Decision);
        transDecision.IdTransicion = conexion.IdTransicion;
        transDecision.EsPuntoSincronizacion = conexion.EsPuntoSincronizacion;
        transDecision.Conexiones.Add(conexion);
    }

    if (transDecision != null)
        transiciones.Add(transDecision);
}
```

Una vez cargados los objetos desde las tablas el siguiente paso es dibujarlos en el diagrama, lo cual se hace en el método `dibujarWorkflow` de la clase `Diagrama`.

```
public void dibujarWorkflow()
{
    //Dibuja las actividades
    foreach (cActividad actividad in workflow.Actividades)
    {
        this.Doc.InsertNode(actividad);
    }

    //Dibuja las conexiones
    foreach (cConexion conexion in workflow.Conexiones)
    {
        cActividad origen =
            BuscarActividad(conexion.IdActividadOrigen) as cActividad;
        cActividad destino =
            BuscarActividad(conexion.IdActividadDestino) as cActividad;

        this.CreateLink(origen.RightPort, destino.LeftPort);
    }

    //Dibuja las transiciones
    foreach (cTransicion transicion in workflow.Transiciones)
    {
        this.Doc.InsertNode(transicion);
    }
```

Primero se dibujan las actividades, las conexiones no pertenecientes a una transición y los nodos de las transiciones. Después se recorren las transiciones para dibujar las conexiones entre los nodos de las actividades y los nodos de las transiciones.

```
    //Dibuja las transiciones
    foreach (cTransicion transicion in workflow.Transiciones)
    {
        this.Doc.InsertNode(transicion);
    }

    //Dibuja las conexiones de las transiciones
    foreach (cTransicion transicion in workflow.Transiciones)
    {
        //Comprueba si dos transiciones están anidadas
        comprobarTransicionesAnidadas(transicion);

        foreach (int idOrigen in transicion.obtenerIdsActividadOrigen())
        {
            //...
```

Publicar workflow

Cuando se quiere modificar la base de datos con el diagrama del workflow creado, se publica el workflow. Esto supone convertir la representación gráfica en objetos y guardarlos en la base de datos. Como en el caso anterior, el problema está en la correspondencia de los objetos con la forma en la que están guardados los elementos en la base de datos. Es la clase `Diagrama` la que incluye el método `publicar`.

```
public void publicar()
{
    //Comprueba que el diagrama sea correcto
    comprobarWorkflow();

    //Comprueba si hay actividades modificadas
    comprobarTieneActividadesModificadas();

    //Publica el workflow
    this.workflow.publicarWorkflow(workflow.Borrador, workflow.TipoWorkflow);

    //Guarda las actividades y conexiones en base de datos
    guardarActividades();
    guardarConexiones();

    //Guarda el XML del diagrama en el workflow
    this.workflow.DiagramaXml = this.Doc.Store();

    //El documento pasa de modificado a guardado
    this.Doc.IsModified = false;
}
```

Primero comprueba que los datos del diagrama son correctos, y luego comprueba que las actividades del diagrama no han sido modificadas, porque si hay actividades modificadas debe notificar que los diagramas que incluyan alguna de las actividades modificadas serán eliminados y será necesario volver a generarlos. Al llamar método `publicarWorkflow` de la clase `cWorkflow` modifica el XML del diagrama y elimina el borrador.

Después guarda las actividades y las conexiones. En el caso de las actividades no tiene mayor dificultad, pero en las conexiones es necesario hacer una conversión para las transiciones.

El siguiente método primero elimina las conexiones del workflow de la base de datos para introducirlas de nuevo. Para cada conexión primero comprueba si es una conexión simple entre actividades y en ese caso la almacena.

```
private void guardarConexiones()
{
    //Eliminar todas las conexiones y las genera otra vez.
    eliminarConexiones(this.workflow.TipoWorkflow);

    //Guarda las conexiones
    cConexion con = null;
    cTransicion trans = null;

    foreach (GoObject obj in this.Doc)
    {
        if (obj is cConexion)
        {
            con = (cConexion)obj;
            trans = null;

            //Guarda las conexiones simples entre actividades
            if (con.FromPort.Node is cActividad && con.ToPort.Node is cActividad)
            {
                con.IdActividadOrigen = ((cActividad)con.FromPort.Node).IdActividad;
                con.IdActividadDestino = ((cActividad)con.ToPort.Node).IdActividad;

                if (con.noExisteConexion(con.IdActividadOrigen, con.IdActividadDestino,
                    this.workflow.TipoWorkflow))
                {
                    trans = new cTransicion();
                    trans.insertarTransicion();
                    con.IdTransicion = trans.IdTransicion;
                    con.IdTipoWorkflow = this.workflow.TipoWorkflow;
                    con.guardarConexion();
                }
            }
```

En el caso de que sean transiciones, distingue entre transiciones de salida y de entrada, buscando las conexiones que pertenecen a una misma transición.

```
            //Guarda las conexiones con una transición de salida
            else if (con.FromPort.Node is cActividad && con.ToPort.Node is cTransicion
                && EsTransicionSalida(con))
                buscarConexionesTransicionSalida(con, ref trans);

            //Guarda las conexiones con una transición de llegada
            else if (con.FromPort.Node is cTransicion && con.ToPort.Node is cActividad
                && EsTransicionLlegada(con))
                buscarConexionesTransicionLlegada(con, ref trans);

            guardarTransicion(trans);
        }
    }
}
```

Estadísticas de la implementación

El módulo consta de 5 formularios, 13 clases y 6 tablas. Se han realizado 10 casos de uso y 29 pruebas de aceptación. La distribución de las 6121 líneas de código totales ha sido la siguiente:

Clases	Líneas de código
RelationshipTool	155
GraphNode	384
GraphDoc	403
GraphView	545
Diagrama	662
cWorkflow	378
cTransicion	156
cSeguimiento	76
cRol	54
cIncidencia	82
cConexion	180
cActividad	262
GraphView	545
Total	3882

Formularios	Líneas de código
CargarWorkflow	62
GraphViewWindow	457
MainForm	1573
NuevoWorkflow	51
PropiedadesActividad	97
Total	2239

5.4 Diseño de los Casos de Prueba

Los casos de prueba se dividen entre las dos versiones del gestor de diagramas, que son el visor y el editor. Los casos de prueba van correlacionados con los casos de uso. Son las siguientes:

5.4.1 Visor

La instanciación de las distintas pruebas es la siguiente:

<table>
<tr><th colspan="2">Caso de Prueba de Aceptación</th></tr>
<tr><td>Código Caso de Prueba: 01</td><td>Código Caso de Uso: 01</td></tr>
<tr><td colspan="2">Descripción de la Prueba:

Comprobar que la aplicación nos ofrece la opción de abrir el borrador de un workflow cuando éste tiene un borrador y un workflow publicado.</td></tr>
<tr><td colspan="2">Condiciones de Ejecución:

Debe de existir un workflow en la base de datos, que esté publicado y que tenga un borrador.</td></tr>
<tr><td colspan="2">Entrada / Pasos de Ejecución:

1- El usuario accede al visor por el menú de ayuda de la aplicación de gestión de tareas, seleccionando el primer workflow.
2- El usuario selecciona “Sí” cuando la aplicación le pregunta si quiere abrir el borrador</td></tr>
<tr><td colspan="2">Resultado Esperado:

El resultado en el paso 1 es que aparece el dialogo de la Figura 40 en el que le pide la confirmación de abrir el borrador del diagrama o el diagrama publicado.</td></tr>
</table>

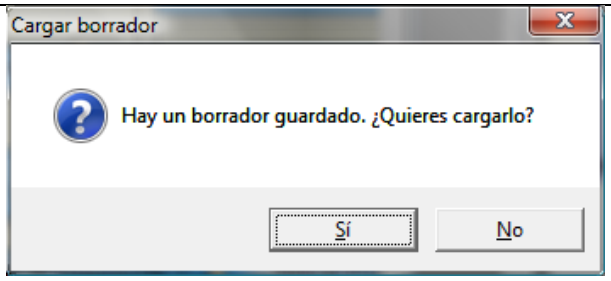


Figura 40 Diálogo de comprobación

El resultado del paso 2 es que se abre el diagrama en formato borrador, indicándolo en la ventana como se puede ver en la Figura 41.

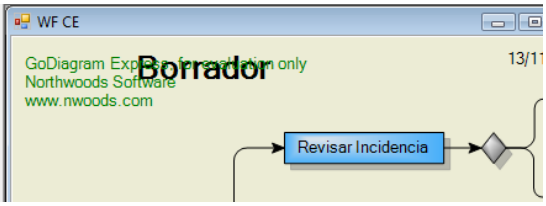


Figura 41 Ejemplo de borrador

Tabla 11. Prueba de aceptación 01

Caso de Prueba de Aceptación	
Código Caso de Prueba: 02	**Código Caso de Uso**: 01
Descripción de la Prueba: Comprobar que las actividades que no están activas se muestran de otro color.	
Condiciones de Ejecución: Debe de existir un workflow en la base de datos y que contenga una actividad en estado no activo.	
Entrada / Pasos de Ejecución: 1- El usuario accede al visor por el menú de ayuda de la aplicación de gestión de tareas, seleccionando el primer workflow. 2- El usuario hace doble clic sobre la actividad en gris.	
Resultado Esperado: El resultado en el paso 1 es que se muestra el workflow seleccionado marcando las actividades no activas en gris como se puede ver en la Figura 42.	

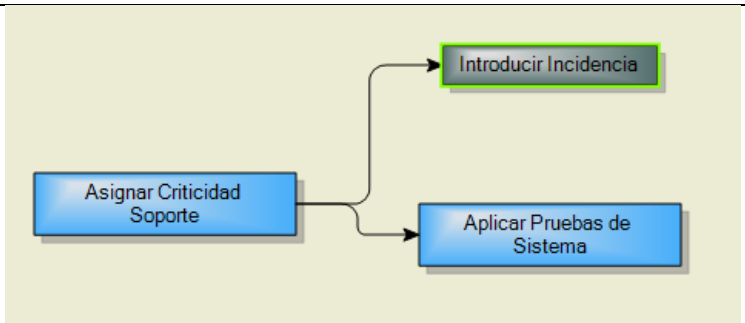


Figura 42 Diagrama con actividades inactivas

El resultado del paso 2 nos muestra la ventana de propiedades de la actividad en la que la casilla "activa" no está marcada.

Tabla 12. Prueba de aceptación 02

Caso de Prueba de Aceptación	
Código Caso de Prueba: 03	**Código Caso de Uso**: 01
Descripción de la Prueba: Comprobar que no es posible la edición del workflow.	
Condiciones de Ejecución: Debe de existir un workflow en la base de datos y que esté publicado.	
Entrada / Pasos de Ejecución: 1- El usuario accede al visor por el menú de ayuda de la aplicación de gestión de tareas, seleccionando el primer workflow. 2- El usuario selecciona con el ratón la segunda actividad del workflow y la arrastra. 3- El usuario selecciona con el ratón la conexión entre las dos primeras actividades y la arratra. 4- El usuario selecciona la primera acción del workflow y presiona la tecla suprimir en el teclado.	
Resultado Esperado: El resultado en el paso 1 es que se muestra el workflow seleccionado. El resultado del paso 2 es que la aplicación no permite el desplazamiento de actividades desde el visor, mostrando el icono de prohibido sobre la acción, como se puede ver en la Figura 43.	

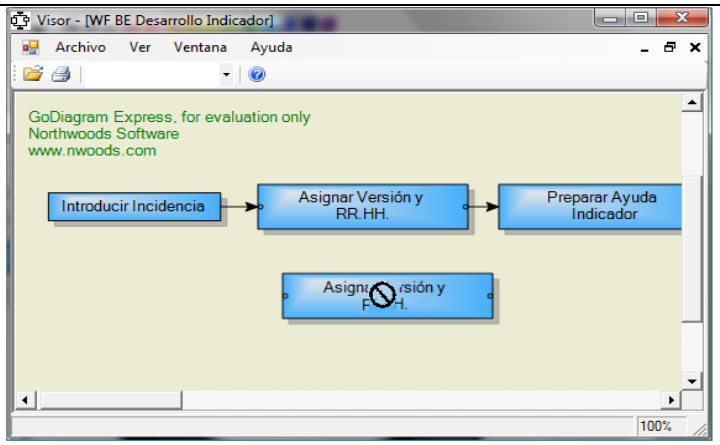


Figura 43 Ejemplo de modificación en visor

El paso 3obtiene el mismo resultado que el paso 2. El resultado del paso 4 no provoca ninguna reacción en la aplicación.

Tabla 13. Prueba de aceptación 03

Caso de Prueba de Aceptación	
Código Caso de Prueba: 04	**Código Caso de Uso**: 02
Descripción de la Prueba: Comprobar que el listado de roles del desplegable son todos los que intervienen en el workflow.	
Condiciones de Ejecución: Debe de existir un workflow en la base de datos y que esté publicado.	
Entrada / Pasos de Ejecución: 1- El usuario accede al visor por el menú de ayuda de la aplicación de gestión de tareas, seleccionando el primer workflow. 2- El usuario selecciona con el ratón el primer valor del desplegable de roles. 3- El usuario selecciona con el ratón el segundo valor del desplegable de roles. 4- El usuario selecciona con el ratón el tercer valor del desplegable de roles. 5- El usuario selecciona con el ratón el cuarto valor del desplegable de roles.	
Resultado Esperado: El resultado en el paso 1 es que se muestra el workflow seleccionado. El	

resultado del paso 2, 3, 4 y 5 es que la aplicación colorea de rojo las actividades que tienen asociado el rol seleccionado como en la Figura 44.

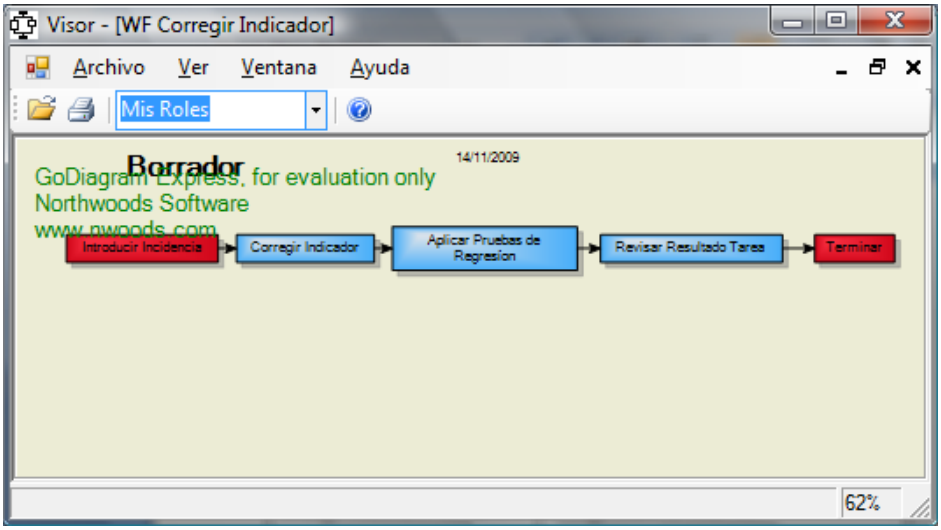


Figura 44 Ejemplo de actividades resaltadas por roles

Después de haber seleccionado todos los roles no debe haber quedado ninguna actividad por cambiar de color.

Tabla 14. Prueba de aceptación 04

Caso de Prueba de Aceptación	
Código Caso de Prueba: 05	**Código Caso de Uso**: 02
Descripción de la Prueba: Comprobar que al seleccionar un rol se colorean todas las actividades del workflow que tienen ese tipo de rol.	
Condiciones de Ejecución: Debe de existir un workflow en la base de datos y que esté publicado.	
Entrada / Pasos de Ejecución: 1- El usuario accede al visor por el menú de ayuda de la aplicación de gestión de tareas, seleccionando el primer workflow. 2- El usuario selecciona con el ratón el primer valor del desplegable de roles. 3- El usuario hace doble clic sobre la primera actividad. 4- El usuario hace doble clic sobre la segunda actividad. 5- El usuario hace doble clic sobre la tercera actividad. 6- El usuario hace doble clic sobre la cuarta actividad.	

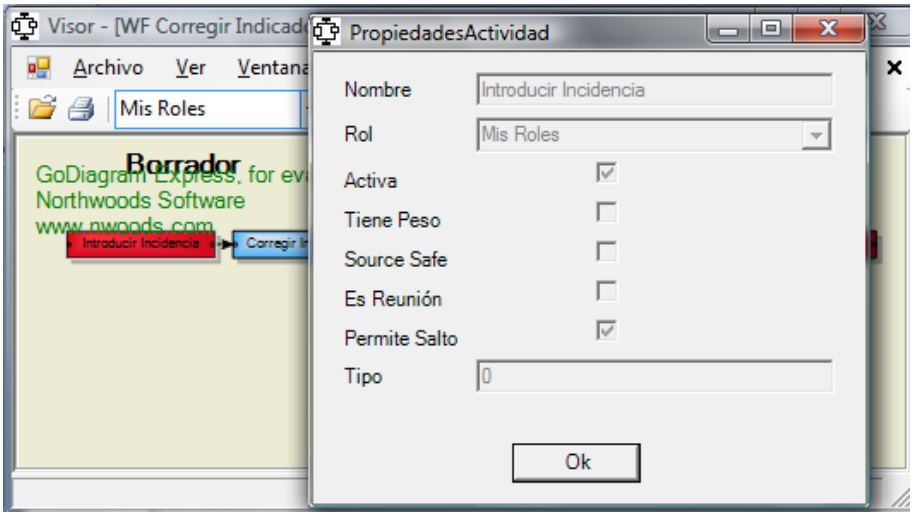
7- El usuario hace doble clic sobre la quinta actividad.
Resultado Esperado: El resultado en el paso 1 es que se muestra el workflow seleccionado. El resultado del paso 2 es que la aplicación colorea de rojo las actividades que tienen asociado el rol seleccionado.      **Figura 45 Ventana de propiedades de la actividad** El resultado de los pasos 3, 4, 5, 6 y 7 muestra, dentro de la ventana de propiedades, el rol de cada una de las acciones, siendo igual al seleccionado de el despegable en caso de que la actividad esté en color rojo, como se puede ver en la Figura 45.

Tabla 15. Prueba de aceptación 05

Caso de Prueba de Aceptación	
Código Caso de Prueba: 06	**Código Caso de Uso**: 03
Descripción de la Prueba: Comprobar que al hacer doble clic en una actividad se abre la ventana de propiedades de la actividad.	
Condiciones de Ejecución: Debe de existir un workflow en la base de datos.	

Entrada / Pasos de Ejecución:

1- El usuario accede al visor por el menú de ayuda de la aplicación de gestión de tareas, seleccionando el primer workflow.
2- El usuario hace doble clic sobre la primera actividad del workflow.

Resultado Esperado:

El resultado en el paso 1 es que se muestra el workflow seleccionado. El resultado del paso 2 se abre la ventana de propiedades de la actividad con todos los controles deshabilitados.

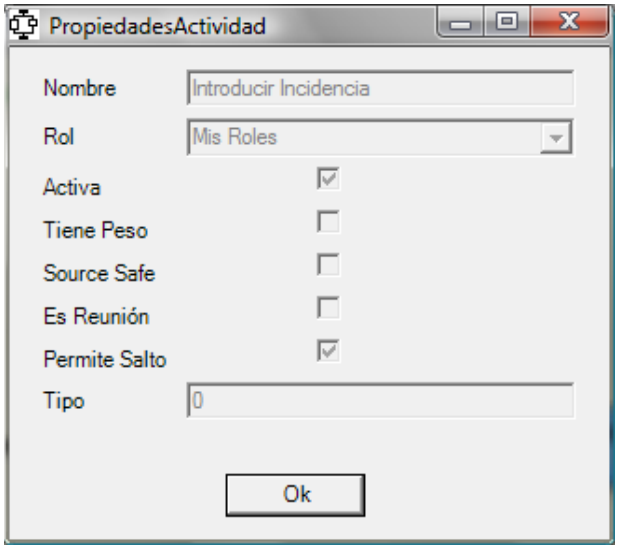
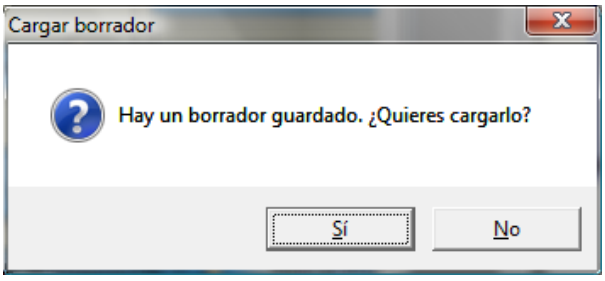


Figura 46 Ventana de propiedades de la actividad

El paso 3obtiene el mismo resultado que el paso 2. El resultado del paso 4 no provoca ninguna reacción en la aplicación.

Tabla 16. Prueba de aceptación 06

5.4.2 Editor

Caso de Prueba de Aceptación	
Código Caso de Prueba: 04	**Código Caso de Uso**: 07
Descripción de la Prueba: Comprobar que la aplicación nos ofrece la opción de abrir el borrador de un workflow cuando éste tiene un borrador y un workflow publicado.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe de existir un workflow en la base de datos, que esté publicado y que tenga un borrador.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre la ventana de selección de workflow desde la barra de herramientas. 3- El usuario selecciona el primer workflow de la lista. 4- El usuario selecciona "Sí" cuando la aplicación le pregunta si quiere abrir el borrador	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. Al realizar el paso 2 se abre la ventana de selección de workflow. El resultado en el paso 3 es que aparece un dialogo como el de la Figura 47 en el que le pide la confirmación para abrir el borrador del diagrama o el diagrama publicado.  **Figura 47 Diálogo de selección de borrador** El resultado del paso 4 es que se abre el diagrama en formato borrador, indicándolo en la ventana.	

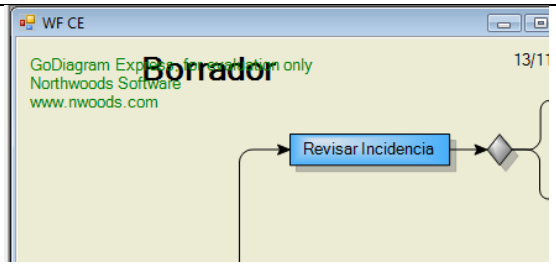


Figura 48 Borrador del diagrama

Tabla 17. Prueba de aceptación 07

Caso de Prueba de Aceptación	
Código Caso de Prueba: 08	**Código Caso de Uso**: 04
Descripción de la Prueba: Comprobar que las actividades que no están activas se muestran de otro color.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe de existir un workflow en la base de datos y que contenga una actividad en estado no activo.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre la ventana de selección de workflow desde la barra de herramientas. 3- El usuario selecciona el primer workflow de la lista. 4- El usuario hace doble clic sobre la actividad en gris.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. Al realizar el paso 2 se abre la ventana de selección de workflow. El resultado en el paso 3 es la Figura 49, en la que se muestra el workflow seleccionado marcando las actividades no activas en gris.	

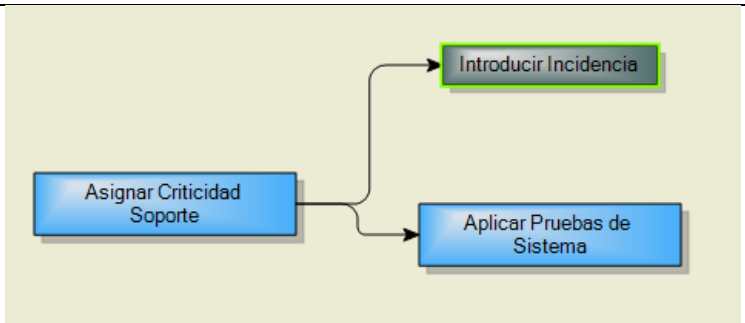


Figura 49 Diagrama con actividades no activas

El resultado del paso 4 nos muestra la ventana de propiedades de la actividad en la que la casilla "activa" no está marcada.

Tabla 18. Prueba de aceptación 08

Caso de Prueba de Aceptación	
Código Caso de Prueba: 09	**Código Caso de Uso**: 05
Descripción de la Prueba: Comprobar que se añade el nuevo borrador al listado de workflows.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario crea un nuevo borrador haciendo clic en el icono de la barra de herramientas de "nuevo". 3- El usuario guarda el borrador creado con el botón "guardar" de la barra de herramientas. 4- El usuario introduce el nombre del workflow creado como "Prueba inserción nuevo workflow" y aprieta en el botón Ok. 5- El usuario abre el listado de workflows con el botón "abrir" de la barra de herramientas.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre una nueva ventana de edición con una etiqueta que tiene por título "Nuevo" y otra en la que se indica la fecha de creación, como se puede ver en la Figura 50.	

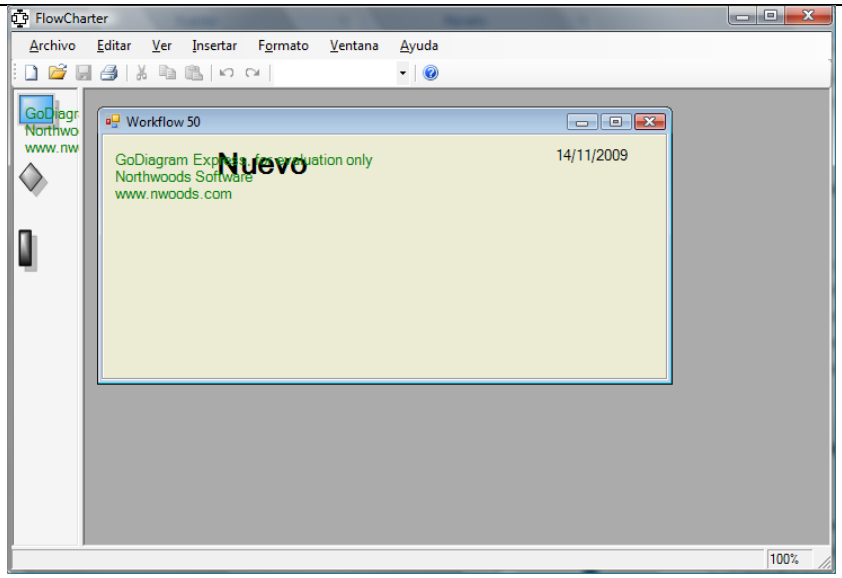


Figura 50 Nuevo workflow

Al realizar el paso 3 la aplicación nos mostrará una ventana para que introduzcamos las propiedades del workflow como la que se muestra en la Figura 51.

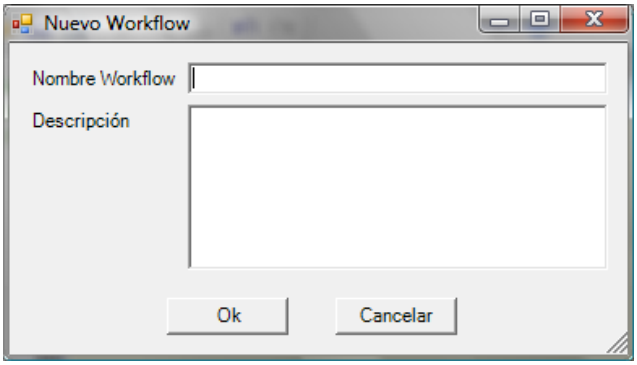


Figura 51 Propiedades del workflow

Después de realizar el paso 4, en el paso 5 nos muestra el listado de workflows de la Figura 52 y el borrador creado está al final de la lista.

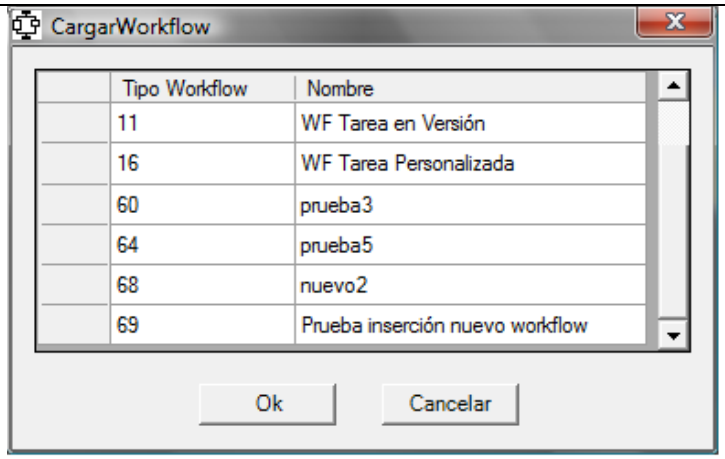


Figura 52 Listado de workflows

Tabla 19. Prueba de aceptación 09

Caso de Prueba de Aceptación	
Código Caso de Prueba: 10	**Código Caso de Uso**: 05
Descripción de la Prueba: Comprobar que el nuevo borrador se guarda como borrador y no como workflow publicado.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario crea un nuevo borrador haciendo clic en el icono de la barra de herramientas de "nuevo". 3- El usuario guarda el borrador creado con el botón "guardar" de la barra de herramientas. 4- El usuario introduce el nombre del workflow creado como "Prueba inserción nuevo workflow" y aprieta en el botón Ok. 5- El usuario abre el listado de workflows con el botón "abrir" de la barra de herramientas. 6- El usuario selecciona el workflow creado.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre una nueva ventana de edición con una etiqueta	

que tiene por título “Nuevo” y otra en la que se indica la fecha de creación, como se puede ver en la Figura 50.

Al realizar el paso 3 la aplicación nos mostrará una ventana para que introduzcamos las propiedades del workflow como la que se muestra en la Figura 51.

Después de realizar el paso 4, en el paso 5 nos muestra el listado de workflows de la Figura 52 y el borrador creado está al final de la lista.

Después de realizar el paso 6 se abrirá el workflow creado y mostrara una etiqueta con título “Borrador”.

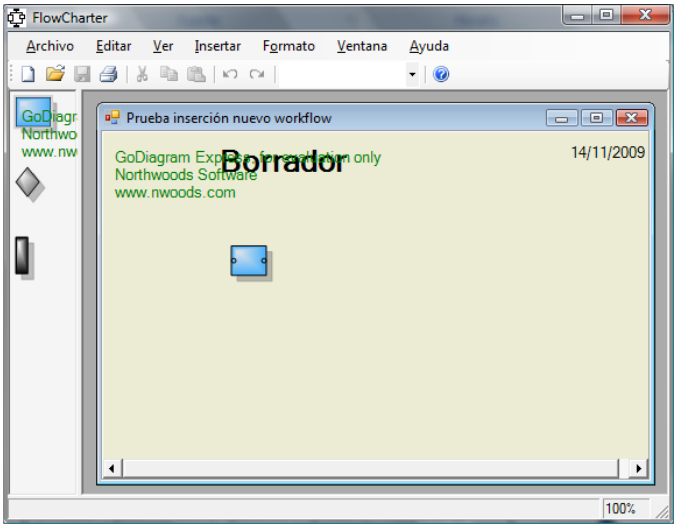


Figura 53 Ejemplo de borrador

Tabla 20. Prueba de aceptación 10

Caso de Prueba de Aceptación	
Código Caso de Prueba: 11	**Código Caso de Uso**: 05
Descripción de la Prueba: Comprobar que no se puede guardar un borrador sin nombre.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario crea un nuevo borrador haciendo clic en el icono de la	

barra de herramientas de “nuevo”.

3- El usuario guarda el borrador creado con el botón “guardar” de la barra de herramientas.
4- El usuario no escribe ningún nombre y aprieta en el botón Ok.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre una nueva ventana de edición con una etiqueta que tiene por título “Nuevo” y otra en la que se indica la fecha de creación, como se puede ver en la Figura 50.

Al realizar el paso 3 la aplicación nos mostrará una ventana para que introduzcamos las propiedades del workflow como la que se muestra en la Figura 51.

Al realizar el paso 4 el sistema nos advertirá que no es posible guardar un borrador sin nombre.

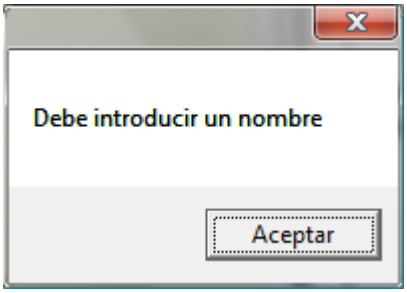


Figura 54 Mensaje de error por faltar el nombre

Tabla 21. Prueba de aceptación 11

Caso de Prueba de Aceptación	
Código Caso de Prueba: 12	**Código Caso de Uso**: 06

Descripción de la Prueba:

Comprobar que no se guardan datos de los elementos del workflow hasta que no se publique el borrador.

Condiciones de Ejecución:

El usuario debe tener acceso a la herramienta de edición. Deben existir dos workflows.

Entrada / Pasos de Ejecución:

1- El usuario accede al editor desde la aplicación de gestión de tareas.
2- El usuario accede al listado de workflows con el botón "abrir" de la barra de herramientas.
3- El usuario selecciona el primer workflow de la lista.
4- El usuario abre la ventana de propiedades de actividad haciendo doble clic sobre la primera actividad.
5- El usuario crea una nueva actividad con el botón "Nueva".
6- El usuario introduce "actividad nueva" como nombre y deja el resto de campos por defecto.
7- El usuario acepta dando al Ok.
8- El usuario guarda el borrador.
9- El usuario abre el listado de workflows.
10- El usuario selecciona el segundo workflow.
11- El usuario abre las propiedades de la primera actividad.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre el listado de workflows. El resultado del paso 3 abre el workflow seleccionado.

El resultado del paso 4 abre la interfaz de propiedades de una actividad como se puede ver en la Figura 55, en la que se incluye un listado de las actividades ya registradas en el sistema y unos controles para la modificación y adición de actividades. Los campos de la actividad seleccionada se muestran deshabilitados.

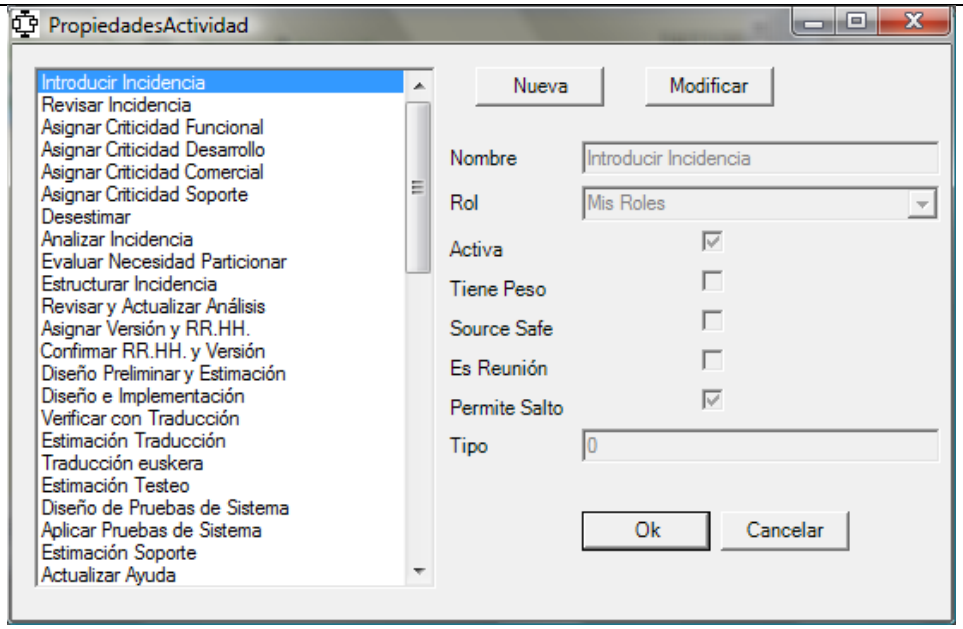


Figura 55 Ventana de propiedades de la actividad en editor

Al realizar el paso 5 los campos se vacían y se habilitan. Al aceptar en el paso 7 después de introducir el nombre, la actividad cambia su texto por el introducido.

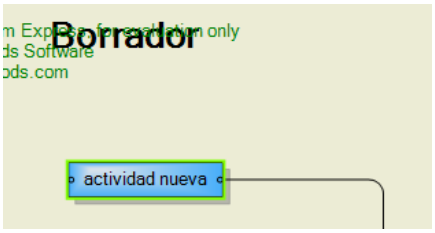


Figura 56 Actividad modificada

Al llegar al paso 11 mostrará el listado de actividades pero sin incluir la actividad añadida en el otro borrador.

Tabla 22. Prueba de aceptación 12

Caso de Prueba de Aceptación	
Código Caso de Prueba: 13	**Código Caso de Uso**: 06
Descripción de la Prueba: Comprobar que al cambiar el nombre del workflow no es posible dejarlo vacío.	

Condiciones de Ejecución:

El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow creado.

Entrada / Pasos de Ejecución:

1- El usuario accede al editor desde la aplicación de gestión de tareas.
2- El usuario abre el listado de workflows.
3- El usuario selecciona el primer workflow.
4- El usuario abre la ventana de propiedades del workflow desde el menú.
5- El usuario borra el nombre y aprieta en el botón Ok.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre el workflow para editar. Al realizar el paso 3 la aplicación nos mostrará la ventana de propiedades del workflow, quedando vacía al realizar el paso 5 como en la Figura 57.

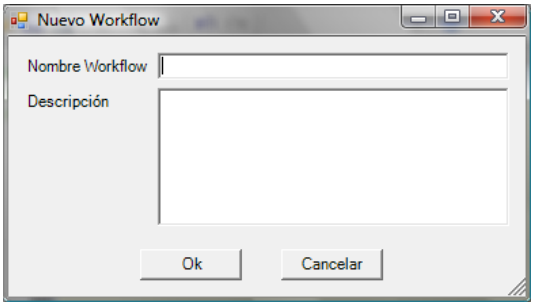


Figura 57 Ventana de propiedades del workflow con nombre vacío

Al realizar el paso 5 el sistema nos advertirá que no es posible guardar un borrador sin nombre.

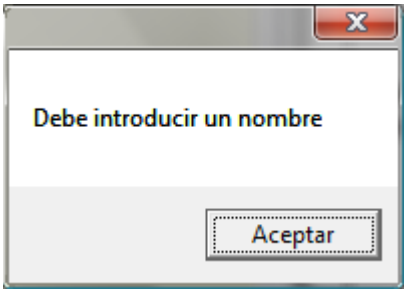


Figura 58 Mensaje de error por no introducir un nombre para el workflow

Tabla 23. Prueba de aceptación 13

Caso de Prueba de Aceptación	
Código Caso de Prueba: 14	**Código Caso de Uso**: 06
Descripción de la Prueba: Comprobar que no es posible guardar un workflow con nombre duplicado.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow creado con nombre "Prueba duplicado".	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario crea un nuevo borrador. 3- El usuario abre la ventana de propiedades del workflow desde el menú. 4- El usuario pone por nombre "Prueba duplicado" y aprieta en el botón Ok.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre el nuevo borrador. Al realizar el paso 3 la aplicación nos mostrará la ventana de propiedades del workflow. Al realizar el paso 4 el sistema advertirá que no es posible guardar dos workflows con el mismo nombre.	

Tabla 24. Prueba de aceptación 14

Caso de Prueba de Aceptación	
Código Caso de Prueba: 15	**Código Caso de Uso**: 06
Descripción de la Prueba: Comprobar que no se pueden modificar actividades que tienen asociadas una incidencia.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow creado con incidencias asociadas a sus actividades.	

Entrada / Pasos de Ejecución:

1- El usuario accede al editor desde la aplicación de gestión de tareas.
2- El usuario abre el listado de workflows.
3- El usuario selecciona el primer workflow.
4- El usuario elimina la primera actividad del workflow.
5- El usuario guarda el workflow.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre el workflow a modificar.

Al realizar el paso 3 se eliminará la primera actividad del workflow y al realizar el paso 4 el sistema advertirá que no es posible modificar un workflow de manera que queden incidencias sin actividad con el mensaje de la Figura 59.

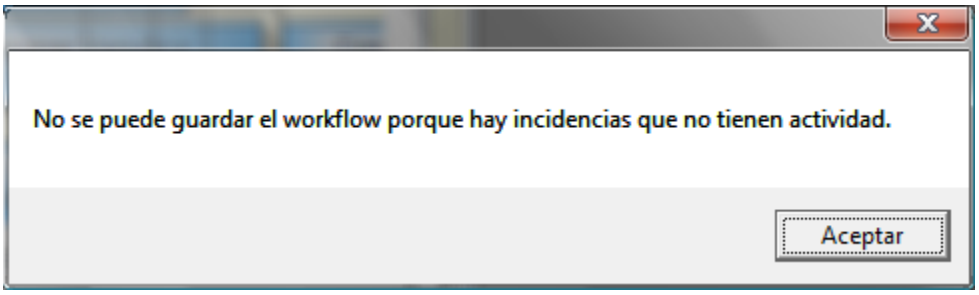


Figura 59 Mensaje de error por tener incidencias asociadas

Tabla 25. Prueba de aceptación 15

Caso de Prueba de Aceptación	
Código Caso de Prueba: 16	**Código Caso de Uso**: 07
Descripción de la Prueba: Comprobar que no se pueden publicar si se modifican actividades que tienen asociadas una incidencia.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow publicado con incidencias asociadas a sus actividades.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre el listado de workflows.	

3- El usuario selecciona el primer workflow.
4- El usuario elimina la primera actividad del workflow.
5- El usuariopublica el workflow.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre el workflow a modificar.

Al realizar el paso 3 se eliminará la primera actividad del workflow y al realizar el paso 4 el sistema advertirá que no es posible modificar un workflow de manera que queden incidencias sin actividad con el mensaje de la Figura 60.

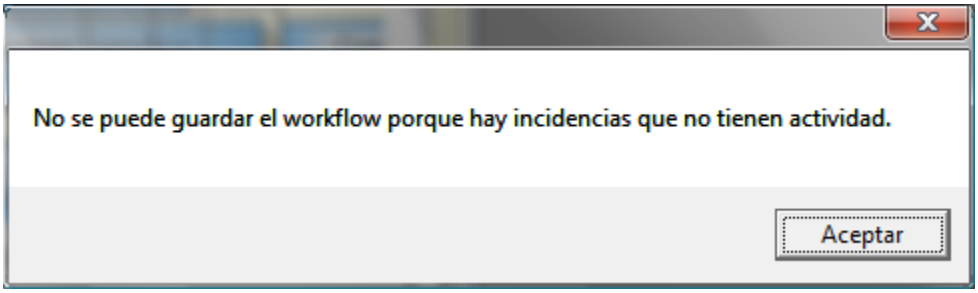


Figura 60 Mensaje de error por modificar actividades con incidencias asociadas

Tabla 26. Prueba de aceptación 16

Caso de Prueba de Aceptación	
Código Caso de Prueba: 17	**Código Caso de Uso**: 07
Descripción de la Prueba: Comprobar que al intentar publicar cuando el diagrama es incorrecto el sistema lo notifica impidiendo que se publique.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario crea un borrador. 3- El usuario añade dos actividades arrastrándolas desde la paleta de nodos. 4- El usuario publica el workflow.	

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre un nuevo borrador. En el paso 3 se añaden las dos actividades vacías al workflow y al realizar el paso 4 el sistema advertirá que no es posible publicar un workflow si hay incoherencias, en este caso nodos sin conectar. El mensaje de error es el que se muestra en la Figura 61.

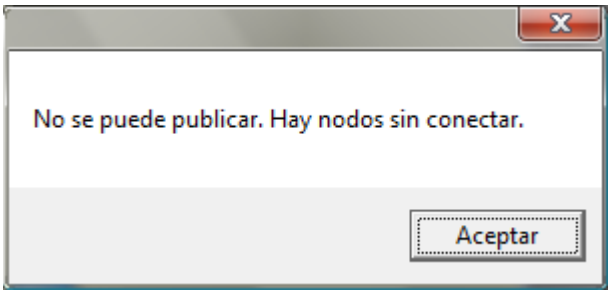


Figura 61 Mensaje de error por guardar con nodos sin conectar

Tabla 27. Prueba de aceptación 17

Caso de Prueba de Aceptación	
Código Caso de Prueba: 18	**Código Caso de Uso**: 07
Descripción de la Prueba: Comprobar que no se puede publicar un workflow con transiciones con varias entradas y salidas.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario crea un borrador. 3- El usuario añade cuatro actividades arrastrándolas desde la paleta de nodos. 4- El usuario añade un nodo de bifurcación. 5- El usuario une dos actividades a la entrada de la transición y las otras dos a la salida. 6- El usuario publica el workflow.	

Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre un nuevo borrador. Después de los pasos 3, 4 y 5 el borrador será como la Figura 62.

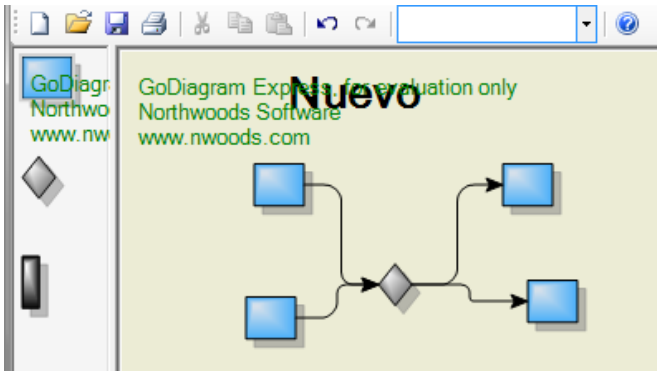


Figura 62 Diagrama con transición incorrecta

Al realizar el paso 6 el sistema advertirá con el mensaje de la Figura 63 que no es posible publicar un workflow si hay incoherencias, en este caso nodos sin conectar.

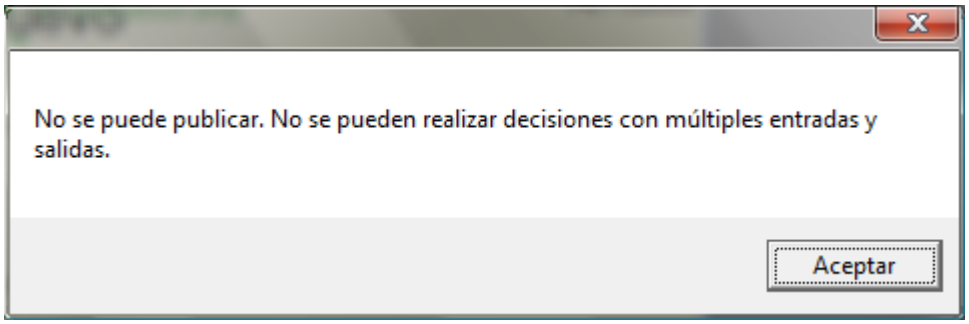


Figura 63 Mensaje de error por malformación del workflow

Tabla 28. Prueba de aceptación 18

Caso de Prueba de Aceptación	
Código Caso de Prueba: 19	**Código Caso de Uso**: 07
Descripción de la Prueba: Comprobar que no se puede publicar un workflow con varias conexiones entre actividades.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición.	

Entrada / Pasos de Ejecución:

1- El usuario accede al editor desde la aplicación de gestión de tareas.
2- El usuario crea un borrador.
3- El usuario añade tres actividades arrastrándolas desde la paleta de nodos.
4- El usuario une una actividad a las otras dos mediante conexiones.
5- El usuario publica el workflow.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre un nuevo borrador. Después de los pasos 3 y 4 el borrador quedará como la Figura 64.

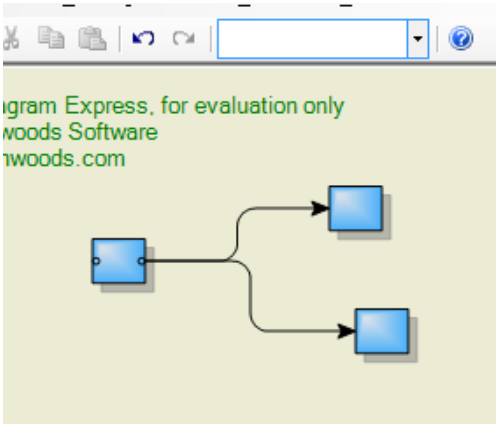


Figura 64 Diagrama con conexiones incorrectas

Al realizar el paso 5 el sistema advertirá con el mensaje de la Figura 65 que no es posible publicar un workflow si hay incoherencias, en este caso varias conexiones saliendo de una actividad.

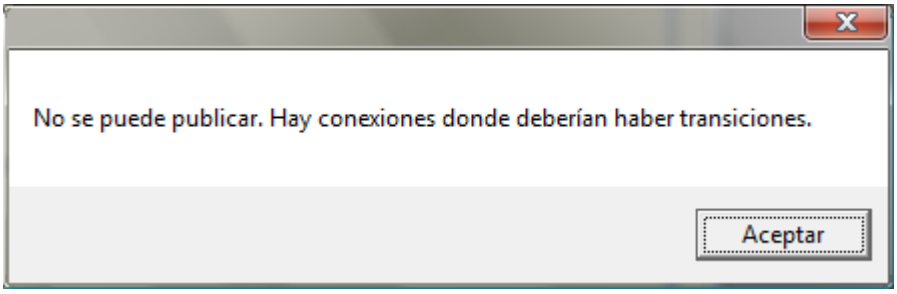


Figura 65 Mensaje de error por workflow incorrecto

Tabla 29. Prueba de aceptación 19

Caso de Prueba de Aceptación	
Código Caso de Prueba: 20	**Código Caso de Uso**: 07

Descripción de la Prueba:

Comprobar que no pueden conectarse dos nodos transición del mismo tipo.

Condiciones de Ejecución:

El usuario debe tener acceso a la herramienta de edición.

Entrada / Pasos de Ejecución:

1- El usuario accede al editor desde la aplicación de gestión de tareas.
2- El usuario crea un borrador.
3- El usuario añade cuatro actividades arrastrándolas desde la paleta de nodos.
4- El usuario añade un nodo de bifurcación.
5- El usuario une dos actividades a la entrada de la transición y las otras dos a la salida.
6- El usuario publica el workflow.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre un nuevo borrador. Después de los pasos 3, 4 y 5 el borrador será como el de la Figura 66.

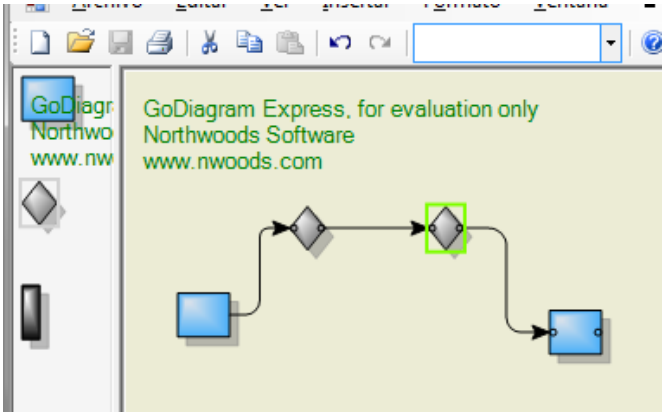


Figura 66 Diagrama con transiciones iguales anidadas

Al realizar el paso 6 el sistema advertirá que no es posible publicar un workflow si hay incoherencias, en este caso nodos sin conectar con el mensaje de la Figura 67.

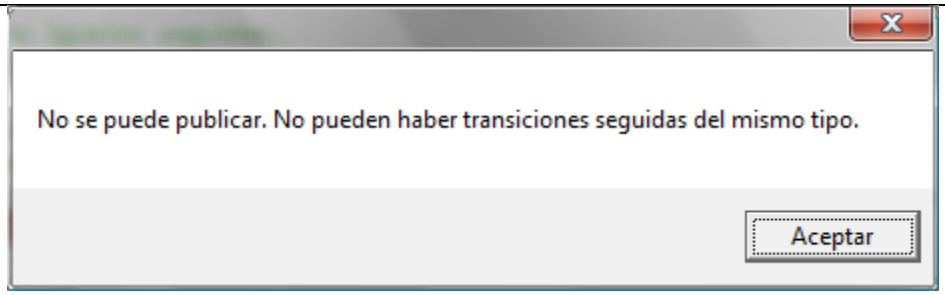


Figura 67 Mensaje de error por transiciones iguales anidadas

Tabla 30. Prueba de aceptación 20

Caso de Prueba de Aceptación	
Código Caso de Prueba: 21	**Código Caso de Uso**: 07
Descripción de la Prueba: Comprobar que al publicar una actividad modificada se eliminan los diagramas de los workflows que contienen esa actividad.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir al menos un workflow que esté publicado y que contenga la actividad a modificar.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario crea un borrador. 3- El usuario añade dos actividades arrastrándolas desde la paleta de nodos. 4- El usuario las une con una conexión. 5- El usuario asigna el tipo de actividad de la primera actividad a "nueva actividad" y aprieta en modificar. 6- El usuario acepta el diálogo de confirmación 7- El usuario modifica el nombre a "nueva actividad 2". 8- El usuario publica el workflow. 9- El usuario abre el workflow que ya estaba creado	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre un nuevo borrador. Después de los pasos 3, 4 y 5 la aplicación nos pedirá confirmación para modificar la actividad	

advirtiendo que serán eliminados los diagramas de los workflows que contengan la actividad modificada.

Al realizar el paso 9 el workflow habrá perdido su formato y estará generado por la aplicación con la actividad modificada.

Tabla 31. Prueba de aceptación 21

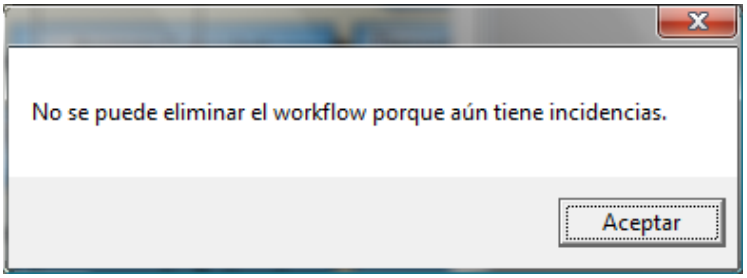
Caso de Prueba de Aceptación	
Código Caso de Prueba: 22	**Código Caso de Uso**: 08
Descripción de la Prueba: Comprobar que no se puede eliminar un workflow con incidencias asociadas.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir al menos un workflow que esté publicado y que tenga incidencias asociadas.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre el listado de workflows. 3- El usuario selecciona el primer workflow. 4- El usuario eliminar el workflow desde el menú.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El paso 3 abre el workflow en el editor. Al realizar el paso 4 la aplicación nos mostrará un mensaje de error.  **Figura 68 Mensaje de error por eliminar actividad con incidencias**	

Tabla 32. Prueba de aceptación 22

Caso de Prueba de Aceptación	
Código Caso de Prueba: 23	**Código Caso de Uso**: 08
Descripción de la Prueba: Comprobar que el workflow eliminado no aparece en el listado de workflows.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir al menos un workflow.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre el listado de workflows. 3- El usuario selecciona el primer workflow. 4- El usuario eliminar el workflow desde el menú. 5- El usuario vuelve a abrir el listado de workflows.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El paso 3 abre el workflow en el editor. Al realizar el paso 4 el workflow se cerrará. El paso 4 mostrará que el workflow eliminado ya no se encuentra en el listado de workflows.	

Tabla 33. Prueba de aceptación 23

Caso de Prueba de Aceptación	
Código Caso de Prueba: 24	**Código Caso de Uso**: 09
Descripción de la Prueba: Comprobar que la actividad se añade al listado de actividades.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir al menos un workflow.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre el listado de workflows. 3- El usuario selecciona el primer workflow. 4- El usuario abre la ventana de propiedades de actividad. 5- El usuario hace clic en “nueva”. 6- El usuario introduce como nombre “nueva actividad” y acepta. 7- El usuario publica el workflow. 8- El usuario vuelva a abrir la ventana de propiedades de actividad.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El paso 3 abre el workflow en el editor. Al realizar el paso 5 se activarán los campos de edición de actividad. Al aceptar se cierra la ventana y la actividad en el diagrama se actualizará. Después de publicar, en el paso 8, en el listado de actividades se mostrará la actividad introducida al final del listado como se puede ver en la Figura 69.	

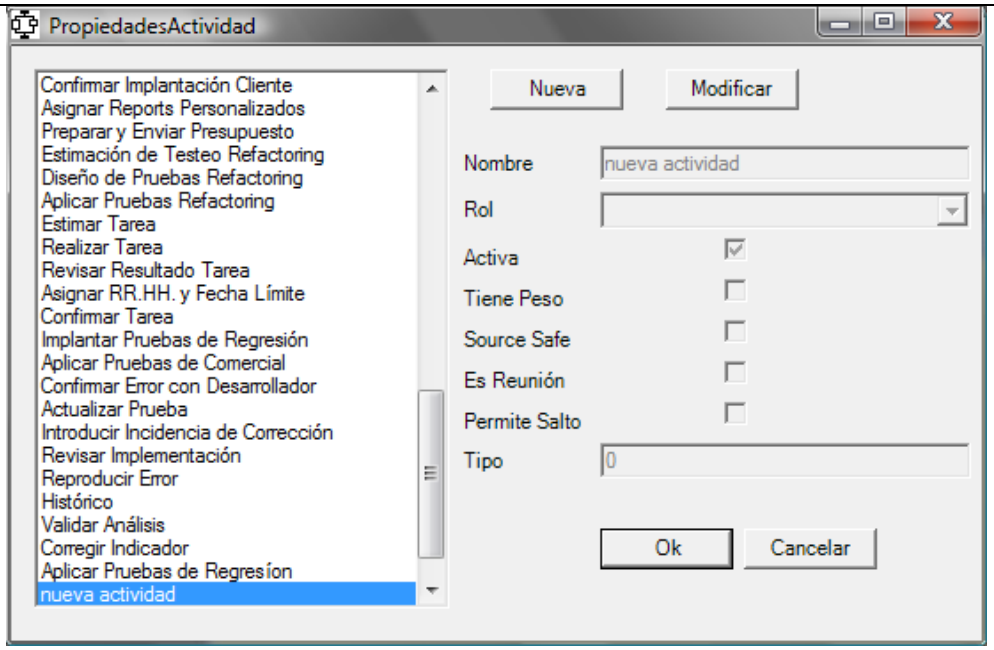


Figura 69 Ventana de propiedades de actividad con nueva actividad

Tabla 34. Prueba de aceptación 24

Caso de Prueba de Aceptación	
Código Caso de Prueba: 25	**Código Caso de Uso**: 09
Descripción de la Prueba: Comprobar que no es posible crear una actividad con nombre duplicado.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow con una actividad de nombre "actividad duplicada".	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre el listado de workflows. 3- El usuario selecciona el primer workflow. 4- El usuario abre la ventana de propiedades del workflow desde el menú. 5- El usuario crea una nueva actividad con nombre "actividad duplicada" y aprieta en el botón Ok.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El	

resultado del paso 2 abre el workflow para editar. Al realizar el paso 3 la aplicación nos mostrará la ventana de propiedades del workflow, quedando vacía al realizar el paso 5. Al realizar el paso 5 el sistema nos advertirá que no es posible guardar una actividad si ya existe otra con el mismo nombre.

Tabla 35. Prueba de aceptación 25

Caso de Prueba de Aceptación	
Código Caso de Prueba: 26	**Código Caso de Uso**: 09
Descripción de la Prueba: Comprobar que no es posible crear una actividad sin nombre.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow con una actividad de nombre “actividad duplicada”.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre el listado de workflows. 3- El usuario selecciona el primer workflow. 4- El usuario abre la ventana de propiedades del workflow desde el menú. 5- El usuario crea una nueva actividad con nombre vacío y aprieta en el botón Ok.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 2 abre el workflow para editar. Al realizar el paso 3 la aplicación nos mostrará la ventana de propiedades del workflow, quedando vacía al realizar el paso 5. Al realizar el paso 5 el sistema nos advertirá que no es posible guardar una actividad sin nombre.	

Tabla 36. Prueba de aceptación 26

Caso de Prueba de Aceptación	
Código Caso de Prueba: 27	**Código Caso de Uso**: 10
Descripción de la Prueba: Comprobar que no es posible modificar una actividad y dejar el nombre vacío.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow con una actividad.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas. 2- El usuario abre el listado de workflows. 3- El usuario selecciona el primer workflow. 4- El usuario abre las de propiedades del workflow desde el menú. 5- El usuario aprieta en modificar. 6- El usuario borra el nombre y aprieta en el botón Ok.	
Resultado Esperado: Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 3 abre el workflow para editar. Al realizar el paso 4 la aplicación nos mostrará la ventana de propiedades del workflow, habilitándose los campos al realizar el paso 5. Al realizar el paso 6 el sistema nos advertirá que no es posible guardar una actividad sin nombre.	

Tabla 37. Prueba de aceptación 27

Caso de Prueba de Aceptación	
Código Caso de Prueba: 28	**Código Caso de Uso**: 10
Descripción de la Prueba: Comprobar que no es posible modificar el rol de una actividad.	
Condiciones de Ejecución: El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow con una actividad.	
Entrada / Pasos de Ejecución: 1- El usuario accede al editor desde la aplicación de gestión de tareas.	

2- El usuario abre el listado de workflows.
3- El usuario selecciona el primer workflow.
4- El usuario abre las propiedades del workflow desde el menú.
5- El usuario aprieta en modificar.

Resultado Esperado:

Al realizar el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 3 abre el workflow para editar. Al realizar el paso 4 la aplicación nos mostrará la ventana de propiedades del workflow, habilitándose todos los campos menos el desplegable de roles al realizar el paso 5.

Tabla 38. Prueba de aceptación 28

Caso de Prueba de Aceptación	
Código Caso de Prueba: 29	**Código Caso de Uso**: 10

Descripción de la Prueba:

Comprobar que no al cancelar la edición no se efectúa ningún cambio.

Condiciones de Ejecución:

El usuario debe tener acceso a la herramienta de edición. Debe existir un workflow con una actividad.

Entrada / Pasos de Ejecución:

1- El usuario accede al editor desde la aplicación de gestión de tareas.
2- El usuario abre el listado de workflows.
3- El usuario selecciona el primer workflow.
4- El usuario abre las propiedades del workflow desde el menú.
5- El usuario aprieta en modificar.
6- El usuario cambia el nombre por “actividad modificada”
7- El usuario aprieta el botón “cancelar”.

Resultado Esperado:

Con el paso 1 se abre la aplicación de gestión de workflows. El resultado del paso 3 abre el workflow para editar. En el paso 4 la aplicación nos mostrará la ventana de propiedades del workflow, habilitándose todos los campos al realizar el paso 5. Al cancelar la modificación el texto de la actividad no ha sido modificado.

Tabla 39. Prueba de aceptación 29

Capítulo 6: Conclusiones

6.1 Conclusiones y trabajo futuro

Para una correcta gestión de proyectos es necesaria la utilización de metodologías y herramientas que ayuden a la planificación y al seguimiento de los mismos. Considerando el proceso de desarrollo de software como un proceso de negocio, y como tal automatizable, una manera efectiva de controlar el proceso es mediante la utilización de workflows. Éstos sirven como patrón ofreciendo información sobre el estado del desarrollo y los siguientes pasos a realizar.

Dentro de las metodologías asociadas a los procesos de producción de software, TUNE-UP es una metodología ágil con una visión pragmática aplicada a desarrollos con equipos pequeños.

TUNE-UP se apoya en la herramienta TUNE-UP Process Tool para una aplicación efectiva de la metodología. Ésta a su vez gestiona las distintas unidades de trabajo mediante los workflows definidos en el sistema.

Los requisitos de flexibilidad para los workflows en TUNE-UP Process Tool llevaron a desarrollar un motor sencillo pero a la vez potente en cuanto a posibilitar muchas acciones no frecuentes en workflows tradicionales.

Para la gestión de los workflows el módulo implementado supone una mejora considerable respecto al sistema manual actual, ya que es incómodo y propenso a errores. Hay que considerar que la mejora de procesos se basa en el refinamiento continuo de los workflows, con lo cual frecuentemente se están haciendo cambios en ella y añadiendo nuevos workflows.

Actualmente se tienen 20 workflows definidos en el sistema, con algunos de ellos de hasta 30 actividades, siendo éstos un factor clave en el funcionamiento de la herramienta TUNE-UP Process Tool.

Para la implementación del módulo hubo que investigar el estado del arte de los componentes para .NET dedicados a la edición de diagramas. Este estudio supuso comparar soluciones comerciales con otras de tipo software libre realizando un ejemplo con cada una de ellas para acabar seleccionando los componentes GoDiagram por su sencillez y soporte.

El módulo desarrollado actualmente se ha integrado y probado en el ambiente de desarrollo de TUNE-UP Process Tool. Se tiene previsto pasarlo a producción en la próxima versión de la herramienta.

Dentro de las futuras líneas de trabajo está la mejora de la gestión de borradores, ofreciendo al usuario la posibilidad de visualizar el borrador y el workflow publicado para poder elegir mejor qué editar. También interesaría implementar las *swim lanes* para la distinción de roles, mejorando la comprensión del workflow. Por último, sería interesante una gestión de históricos de workflows, permitiendo ver la evolución de los mismos en incluso volver a un estado anterior.

La realización de esta tesis ha permitido al autor trabajar en un proyecto real teniendo que profundizar en tecnologías de workflows, librerías gráficas, en aspectos de una metodología y herramientas que las soportan. Para ello ha sido clave la colaboración con el equipo implicado en el desarrollo de TUNE-UP Process Tool.

Anexo: Referencias

Referencias

1. AddFlow for .NET. http://www.lassalle.com
2. Adobe LiveCycle Workflow. http://www.adobe.com/products/livecycle/processmanagement/
3. Allen, R. Open Image Systems Inc., United Kingdom Chair, WfMC External Relations Comittee. *The Workflow Handbook 2001*. Workflow Management Coalition, 2001.
4. Auraportal BPM. http://www.auraportal.com
5. Beck K. *Extreme Programming Explained: Embrace Change*. Reading, Massachusetts: Addison-Wesley, 2000.
6. *Business Process Execution Language for Web Services Specification, version 1.1*. IBM Corp, 2003.
7. *Business Process Management Initiative* http://www.bpmi.org/
8. *Business Process Management Notation* http://www.bpmi.org/
9. *Business Process Reengineering and Beyond*. IBM Corp. http://www.redbooks.ibm.com/abstracts/sg242590.html
10. *Business Process Specfication Schema.* http://www.ebxml.eu.org/process.htm
11. CMMI v1.2. *Capability Maturity Model Integrated*. SEI–Software Engineering Institute. Carnegie Mellon: 2009. www.sei.cmu.edu/cmmi/models/
12. DynaFlow EZ-Process. http://www.ez-process.com
13. Fujitsu – Interstage BPM. http://www.fujitsu.com/interstage
14. GoDiagram. http://www.northwoods.com
15. Hollingsworth, D. *The Workflow Reference Model*. Workflow Management Coalition 1995.
16. Hommes, B. *The Evaluation of Business Process Modeling Techniques*. TU Delft. 2004.

17. IBM WebSphere Business Monitor. http://www.ibm.com/software/integration/wbimonitor/

18. Integrated Definition Methods. http://www.idef.com/

19. ISO-International Organization for Standardization. *ISO/IEC 15504, SPICE: Software Process Improvement and Capability dEtermination*. 1998.

20. ISO-International Organization for Standardization. *ISO/IEC 90003, Software engineering-Guidelines for the application of ISO 9001:2000 to computer software*. 2004.

21. K2.NET.http://www.k2.com

22. Kroll P., Kruchten P., Booch G. *The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP*. Addison-Wesley Professional. 2003.

23. Lotus Notes. http://www-01.ibm.com/software/es/lotus/

24. MAP-Ministerio de Administraciones Públicas de España. *Métrica Versión 3, Metodología de planificación, desarrollo y mantenimiento de sistemas de información*. 2005. www.csi.map.es/csi/metrica3/

25. Marante M, Letelier P, Suarez F. TUNE-UP: Un enfoque pragmático para la planificación y seguimiento de proyectos de desarrollo y mantenimiento de software. 2009

26. Microsoft BizTalk. http://www.microsoft.com/biztalk/

27. Netron http://www.orbifold.net/default/?page_id=1272

28. Nevron Diagram. http://www.nevron.com

29. OfBiz. The Apache Open for Business Project. http://ofbiz.apache.org/

30. OMG's Meta Object Facility. http://www.omg.org/mof/

31. Open Business Engine http://obe.sourceforge.net/

32. Open Diagram. http://www.codeplex.com/opendiagram

33. Oracle Fusion Middleware. http://www.oracle.com/lang/es/products/middleware/index.html

34. RosettaNET. http://www.rosettanet.org/

35. Schwaber K., Beedle M. *Agile Software Development with Scrum*. Prentice Hall, Series in Agile Software Development. 2001.

36. Syncfusion Essential Diagram. http://www.syncfusion.com

37. Ultimus BPM Suite. http://www.ultimus.com

38. UML Profile for Enterprise Distributed Object Computing (EDOC). http://www.omg.org/technology/documents/profile_catalog.htm#UML_for_EDOC

39. Watts, H. *Introducción al proceso de software personal*. Traducido por Pearson Education S.A. Addison-Wesley Iberoamericana. 2001.

40. White, S. *Introduction to BPMN*. IBM Corp.

41. yWorks. *yFiles Diagramming Components for the .NET platform*. http://www.yworks.com