# UNA HERRAMIENTA DE USO DOCENTE PARA LA CAPTURA DE REQUISITOS FUNCIONALES A PARTIR DE *WORKFLOWS*[*]

C. Hernández, M. A. Laguna, E. Manso, J. M. Marqués

Departamento de Informática
Universidad de Valladolid
Edificio T.I.T. – Campus Miguel Delibes,
47320 Valladolid
{chernan, mlaguna, manso, jmmc}@infor.uva.es

**Resumen**

Un objetivo de las asignaturas de Ingeniería del Software es formar a los alumnos en las principales técnicas de Análisis y Diseño, de forma que las puedan aplicar para construir un nuevo sistema informático que debe cumplir las expectativas que un usuario tiene planteadas. En este artículo se describe un método de trabajo y una herramienta de apoyo que se está utilizando en la Universidad de Valladolid para delimitar con precisión cuál debe ser la funcionalidad del tal sistema, partiendo de los flujos de trabajo del sistema actual. Este método y herramienta son independientes del paradigma de desarrollo empleado con posterioridad y pueden generar resultados en formatos reconocibles por las herramientas CASE más utilizadas.

**Abstract**

Preparing students to handle the most important techniques of System Analysis and Design is one of the main goals of the Software Engineering courses. The aim of this goal is to make them capable of building a new system that satisfies the requirements of a user. This paper describes a new method for determining, in a very precise way, the desired system functionality starting from the current system workflows. This method is complemented with a support tool that aid the students during the process. Both, the method and the tool, are being used at the Universidad de Valladolid. Neither the method nor the tool entails the further used development method or paradigm. The proposed tool produces outputs readable by the most wide used CASE tools.

**Palabras Clave**: Herramienta CASE, Ingeniería del Software, requisitos, *workflows*.

## 1. Introducción

La Ingeniería del Software constituye un área de interés fundamental en los estudios de Informática, pero nuestra experiencia docente en este campo nos dice que los inicios de los alumnos en esta materia no son fáciles. Uno de los objetivos que han de conseguir es aprender a construir un nuevo sistema informático que responda a las expectativas que un determinado usuario tiene planteadas.

La construcción de un nuevo sistema pasa, ineludiblemente, por la comprensión total de los requisitos del mismo, requisitos que han de ser suministrados, necesariamente, por el usuario que ha encargado su construcción. A partir de los requisitos, se podrá precisar la funcionalidad esperada. Siguiendo las fases de producción de un sistema software, se obtendrá, al final, el producto concreto demandado por el usuario. La

[*] Este trabajo ha sido parcialmente financiado por la Junta de Castilla y León y el Ministerio de Educación y Ciencia.

utilización adecuada de herramientas CASE contribuye, en gran manera, al éxito del proceso de desarrollo, bien como ayuda en la resolución de problemas complejos, bien generando la documentación cómodamente, etc.
Por todo lo dicho anteriormente, un objetivo esencial de las asignaturas de Ingeniería del Software es formar a los alumnos en las principales técnicas de Análisis y Diseño, de forma que las puedan aplicar a la resolución de los anteriores problemas planteados, utilizando herramientas CASE como soporte. Es en este punto donde podemos encuadrar el objeto de este artículo.
Proponemos un método de trabajo y una herramienta de apoyo para delimitar con precisión cuál debe ser la funcionalidad del nuevo sistema, partiendo de los flujos de trabajo del sistema actual. La técnica básica está soportada por una variante particular de éxito comprobado del modelado de *workflows* documentales conocida como diagrama Documentos-Tareas (dDT). Además la herramienta va a generar resultados en formatos reconocibles por las herramientas CASE más utilizadas, lo que permitirá que sea posible continuar con la construcción del nuevo sistema utilizando las herramientas CASE más conocidas, como, por ejemplo, EasyCase®.
En la siguiente sección se presenta el entorno educativo en el que se ha experimentado con el método y la herramienta propuestos. La tercera sección nos sirve para introducir brevemente los conceptos fundamentales de los dDT. En la cuarta sección presentamos la herramienta que hemos desarrollado. Finalmente, exponemos las líneas de nuestro trabajo futuro y las conclusiones de este trabajo.

## 2. Entorno Educativo

Nuestro objeto de estudio se centra en una asignatura de Ingeniería del Software ubicada en el primer cuatrimestre del tercer curso de la titulación de Ingeniería Técnica de Informática de Gestión, y más concretamente en la realización de las prácticas de dicha asignatura.
Se trata de la primera asignatura que cursan los alumnos en relación con la Ingeniería del Software. Uno de sus objetivos es introducir a los alumnos en los conceptos y principios de la Ingeniería del Software y formarlos en las principales técnicas de Análisis y Diseño Estructurado. En particular, se aborda la Fase de Análisis en la construcción de los sistemas software, siguiendo el enfoque estructurado [5, 9].
Con el fin de evaluar la capacidad de utilización de los conocimientos teóricos, se propone a los alumnos la realización de un pequeño proyecto, guiado bajo enseñanza tutorizada y que consiste en la construcción de una aplicación informática de gestión. El objeto de la situación a mecanizar es de libre elección por parte de los alumnos, que se han de organizar en grupos de tres personas para poder elaborar el trabajo. La estructura del mismo ha de adaptarse a cada una de las partes del Análisis y Diseño desarrolladas en el programa de la asignatura. Para su desarrollo se dispone de un laboratorio de herramientas CASE, entre las que se encuentra la herramienta que hemos desarrollado.
De una forma global, la metodología para la realización de este trabajo práctico en equipo se concreta en los siguientes pasos:

1. Constitución del equipo de trabajo y propuesta, por parte de éste, del sistema a mecanizar.
2. Evaluación de las propuestas de trabajo por el profesor.
3. Confirmación del trabajo a realizar.
4. Desarrollo del trabajo.
5. Entrega de la documentación.
6. Evaluación y calificación del trabajo por parte del profesor.

El paso 4 es el que constituye la etapa de ejecución del trabajo práctico, en ella cada equipo ha de realizar el análisis de requisitos y la especificación funcional del sistema software que ha propuesto.
Para llegar a definir los requisitos del nuevo sistema es necesario realizar una recopilación de la información que maneja el usuario. Existen técnicas que ayudan a obtener esta información, todas ellas encaminadas a eliminar las barreras que existen entre el usuario y el analista y que impiden una comunicación eficaz entre ambos: diferentes vocabularios, desconocimiento del entorno y/o de las posibilidades tecnológicas, etc.
En el caso del trabajo práctico que realiza el alumno, esta tarea es todavía más difícil porque es la primera vez que éste se enfrenta a la tarea de extraer, de entre la información que le proporciona el usuario, aquello que debe ser considerado como requisito para la construcción del nuevo sistema.

Nosotros pretendemos apoyar esta fase previa al desarrollo, cuando el usuario final no sabe exactamente lo que quiere o sabe lo que quiere pero no cómo lo quiere, y el alumno no tiene experiencia en la elicitación de los requisitos. En este sentido parece más práctico y realista que el analista/alumno se apoye en la experiencia que tiene el usuario sobre su forma de trabajo hasta ese momento. A partir de esta experiencia, el analista/alumno puede definir la funcionalidad del nuevo sistema, contemplando varias alternativas posibles.
Nuestra propuesta se basa en la utilización de los *workflows* como herramienta para la extracción/recolección de requisitos. Además, hemos estudiado las posibilidades que proporcionan estas herramientas para obtener mecánicamente distintos resultados útiles para la siguiente tarea a acometer, la fase inicial de especificación del sistema: listas de eventos y respuestas del sistema, diagramas de flujo de datos (DFDs) de varios niveles, etc.

## 3. El Diagrama Documentos-Tareas (dDT)

En los últimos años se ha incorporado a los sistemas de información de las empresas el concepto de flujo de trabajo o *workflow* como herramienta de modelado de procesos y, al mismo tiempo, como infraestructura (*workflow enactment*) en la que integrar las aplicaciones específicas de facturación, contabilidad, etc. La *Workflow Management Coalition* [8] ha establecido estándares para ambas posibilidades. Desde el punto de vista de la extracción/recolección de requisitos, la parte del estándar relacionada con el modelado y definición de flujos de trabajo resulta de especial interés.
El modelo estándar propuesto por la WfMC [8] establece una serie de requisitos mínimos para que una herramienta que modela y/o controla las tareas desarrolladas en un sistema de información pueda ser considerada un sistema gestor de flujos de trabajo. Concretamente define un *Workflow Management System* como "un sistema que define, gestiona y ejecuta flujos de trabajo".
El metamodelo propuesto por WfCM [8] constituye una base mínima que incorpora sus propios mecanismos de ampliación. Define básicamente actividades y transiciones que conectan esas actividades. Las construcciones que soporta incluyen rutas de actividades secuenciales, alternativas, iterativas y paralelas (ver Figura 1).

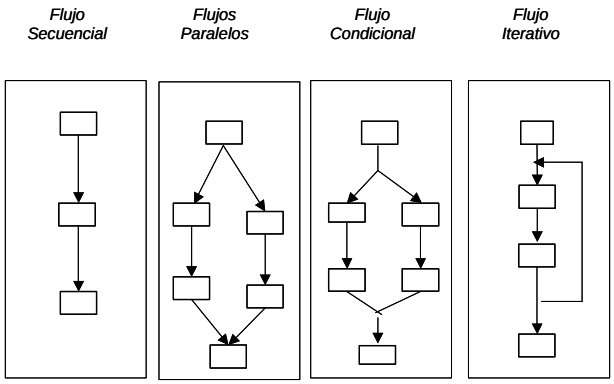


**Figura 1.** Las construcciones básicas para la definición de flujos de trabajo.

Entre las distintas ampliaciones del modelo, los flujos de trabajo documentales constituyen la variante más atractiva para el análisis de sistemas de información. Se trata de añadir al modelo, de forma explícita, los documentos manejados por el sistema. En esta línea, los diagramas Documentos-Tareas [2] se han utilizado desde hace tiempo como soporte para el análisis y la validación de los resultados obtenidos a partir de las entrevistas a los usuarios del futuro sistema.
Se trata, fundamentalmente, de un diagrama que representa el recorrido y tratamiento de la información desde que entra en un sistema hasta que sale del mismo, transformada apropiadamente.

Esta información, que físicamente puede estar soportada sobre cualquier medio (incluyendo la comunicación oral no estructurada), se representa de forma abstracta como un documento de un tipo genérico. El uso de esta técnica está muy relacionado con el enfoque sistémico de la Ingeniería del Software, del cuál es ejemplo paradigmático el método Merise [7].
Aunque se puede enriquecer con más iconos, diferenciando por ejemplo soportes de información o formas de comunicación, la versión más simple es la que más beneficios puede aportar, dados los objetivos que se pretenden. En nuestra herramienta sólo utilizaremos tres elementos básicos, *agente*, *documento* y *tarea*. Además, el sistema se puede dividir opcionalmente en tantos subsistemas como puestos de trabajo diferentes se hayan detectado.
En la Figura 2 aparece un fragmento típico de un dDT que muestra la interacción con clientes y proveedores cuando el sistema recibe un pedido del exterior y se desencadenan una serie de tareas en el sistema, con los documentos asociados a esas tareas (el documento D2 se muestra como archivado, probablemente para su utilización posterior por otras tareas).
Los elementos básicos de un dDT son los siguientes:

- **Agente externo**: Es el origen o destino final de los documentos manejados por nuestro sistema. Puede tratarse de personas individuales, organizaciones u otros sistemas. Equivale al concepto de entidad externa de los DFDs.
- **Puesto de trabajo o agente interno:** Procesador de información del sistema actual, equivalente a un trabajador tipo o conjunto de trabajadores (incluso un departamento) que realizan las mismas tareas.
- **Documento**: Cualquier transmisión de información, independientemente del formato o soporte físico utilizado. Los documentos pueden ser generados en el exterior por un agente externo o en el interior y en este caso irán asociados a una tarea e, indirectamente, a un puesto de trabajo.

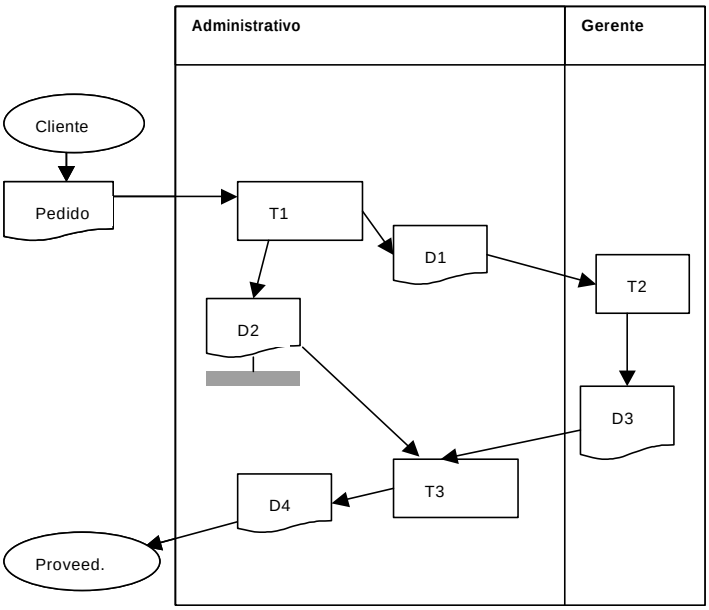


**Figura 2:** Un fragmento de un diagrama Documentos-Tareas

- **Tarea**: Conjunto de actividades realizadas en el sistema por cualquier puesto de trabajo. Una tarea se desencadena por la presencia de uno o varios documentos de entrada o un evento temporal (como alternativa, se puede considerar que un evento temporal es un documento originado por un agente externo especial que informa del tiempo transcurrido). Toda tarea debe tener algún resultado visible, es decir, debe generar algún documento.

- **Flujo**: Es un arco que une los nodos de los tres tipos precedentes. Son legales los flujos en ambos sentidos entre · un agente externo y un documento, y · un documento y una tarea.

Los elementos mencionados y su representación gráfica, que se puede apreciar de forma intuitiva en el ejemplo de la Figura 2, permiten construir de forma mecánica este tipo de diagramas sin más que seguir la descripción que hacen los usuarios de su trabajo diario: cada usuario comienza habitualmente explicando que "recibe un documento D1 del departamento W1 y lo tramita de tal forma, enviando una copia al departamento W2 y archivando una copia…". Los puestos de trabajo que componen el sistema en estudio se muestran opcionalmente como calles en el dDT, al estilo de los diagramas de actividades de UML [1].
El diagrama así construido permite, como primera ventaja, comprobar inconsistencias e ineficiencias en la forma de trabajo actual (documentos que se guardan en un cajón y nunca salen de ahí, documentos que salen hacia el departamento W2 pero el departamento W2 no recibe nunca, etc.). Así, ponemos a disposición del analista/alumno los elementos necesarios para llevar a cabo una labor de comprensión y validación de los procesos del sistema actual.
Además el diagrama va a permitir al analista/alumno definir los límites exactos del sistema que pretende construir. En efecto, una vez establecidas todas las tareas que se realizan en el sistema, y admitiendo que el flujo de tareas en sí mismo es correcto y ha sido validado por los usuarios, se pueden analizar los puestos de trabajo y si algunos de ellos realizan tareas básicamente manuales (entrega de documentos en correos, firmas

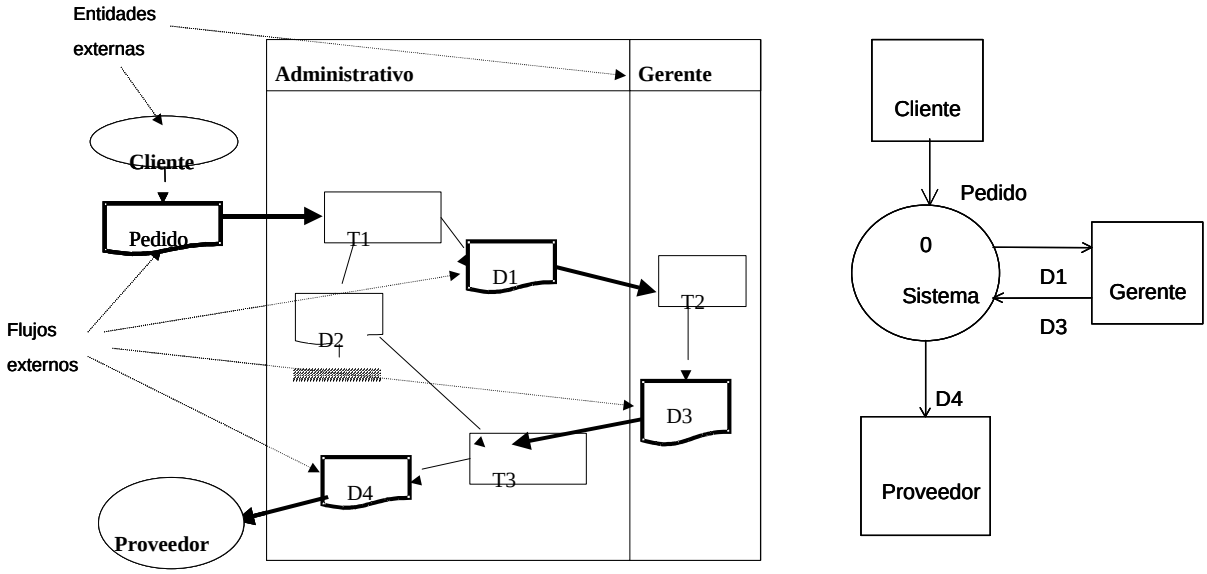


**Figura 3:** Transformación automática de un dDT en un DFD de contexto

y autorizaciones de salida de documentos, etc.) o difícilmente automatizables en un futuro inmediato, estos puestos de trabajo pueden ser etiquetados como *no automatizables* y se tratarán exactamente igual que los agentes externos.
Por otro lado, las tareas que se marcaron como disparadas por el tiempo deben ser examinadas con detalle y estudiar la posibilidad de que puedan ser disparadas por algún documento relacionado con ellas. Del mismo modo es posible que un documento archivado (un documento pasivo) pueda ser reconvertido en un documento activo, capaz de disparar una tarea que hasta ahora era de tipo temporal.
En resumen, después de esta labor de estudio se obtiene un conjunto de dDTs con distintas alternativas marcadas, sobre todo, por las fronteras de automatización que se han definido en cada caso y con dos planos de estudio según consideremos la presencia o no de los puestos de trabajo asociados a las tareas.

La utilidad de los dDTs resulta de este modo evidente. Lo que se propone ahora es incrementar la información del diagrama, añadiendo detalles de los datos que incorpora cada documento o asociando prototipos de pantallas o informes a las tareas del diagrama. Obtenemos, de este modo, unos dDT enriquecidos. Lo que planteamos en esta propuesta es que se puede obtener información muy valiosa para el modelado de sistemas a partir de estos dDTs. Proponemos, en consecuencia, una transformación de los dDTs en DFDs para desarrollo estructurado.
Siguiendo esta línea, hemos definido y desarrollado algoritmos que, a partir de estos diagramas enriquecidos, permiten obtener automáticamente resultados en forma de DFDs.
En líneas generales, para transformar un dDT en un DFD, se deben estudiar todas las posibles respuestas del sistema a cada uno de los estímulos externos para llegar, así, a una primera versión en bruto del modelo esencial [9], expresado como el DFD de contexto, la lista de eventos y respuestas del sistema y un modelo inicial de comportamiento. Si se plantean varias alternativas de automatización, para cada dDT se obtendrá un modelo esencial distinto.
La Figura 3 muestra gráficamente la equivalencia de agentes externos (o puestos no incluidos en el sistema) y entidades externas, así como la relación de los documentos externos con los flujos del diagrama de contexto. Haciendo esta transformación de forma automática se llega al DFD de contexto de la figura.

## 4. La Herramienta Gendoc

Esta herramienta permite a los alumnos dibujar y manipular de forma fácil y efectiva los dDT, para que ellos realicen, en una primera fase, la captura de requisitos del sistema que han de construir.

### 4.1. Objetivos de la herramienta.

- Ofrecer al alumno una herramienta gráfica y visual que le permita dibujar y manipular los dDT.
- Realizar comprobaciones sobre la corrección sintáctica de los dDT.

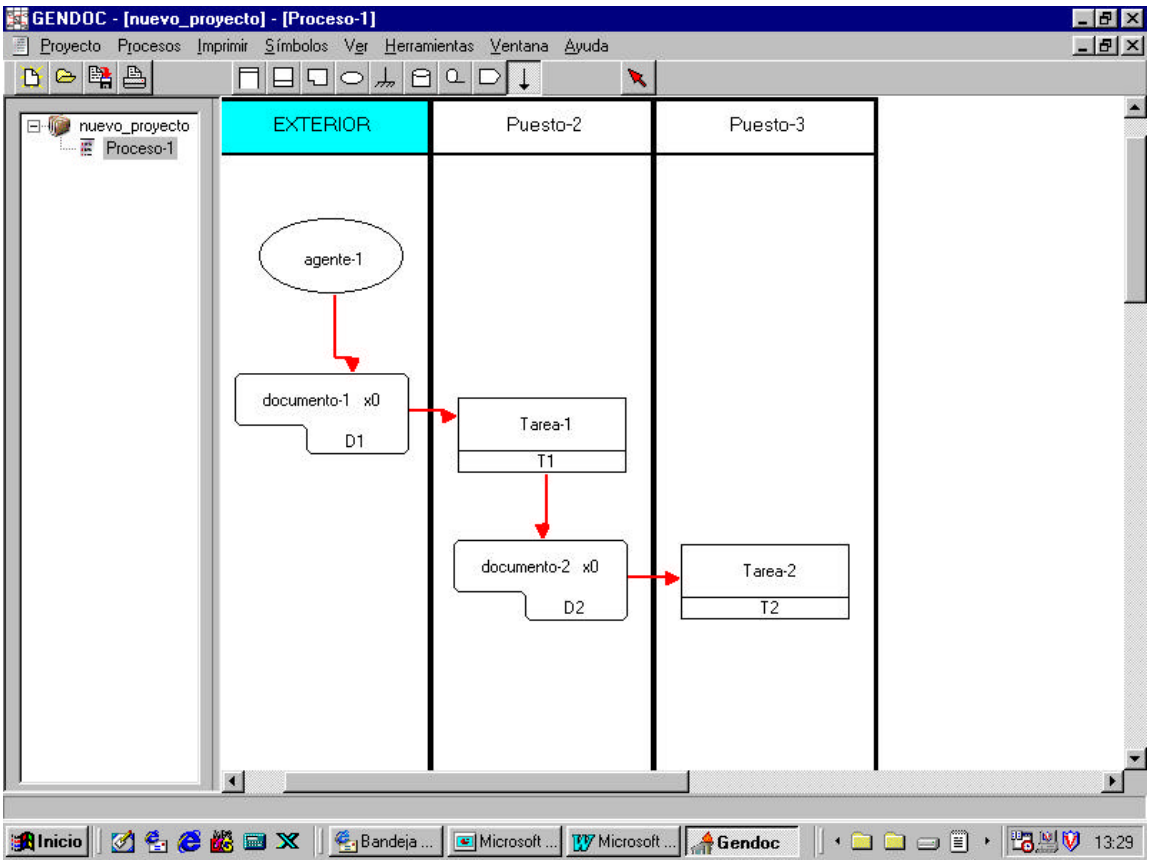


**Figura 4:** Una imagen de la herramienta en funcionamiento

- Ofrecer un entorno integrado que contemple todas las fases de la construcción de un sistema software, incluyendo la de extracción/recolección de los requisitos.
- Obtener de manera automática los DFDs de las primeras fases del Análisis.

### 4.2. Estructura de la herramienta.

Se ha desarrollado una versión inicial de un prototipo de libre disposición para el entorno Windows [4] y la funcionalidad que se ha implementado incluye la creación, mantenimiento y comprobación de la corrección sintáctica de los diagramas dDT. Así mismo, permite imprimir los diferentes diagramas construidos, la lista de eventos y otros listados suplementarios con información de los documentos de cada proceso y las tareas de cada proceso.
Los requisitos necesarios para ejecutar la herramienta son:

- Sistema Operativo Windows 95 o superior.
- 1,52 MegaBytes de espacio libre en el disco duro.
- 3,53 MegaBytes, si no se tiene instalado el gestor de base de datos de Borland (BDE), ya que la herramienta está desarrollada en Delphi.

El concepto básico con el que se trabaja es el de *proyecto*, de forma que un proyecto es un conjunto de *procesos*. Para comenzar a utilizar la herramienta, lo primero que hay que hacer es crear un proyecto. Todos los procesos que se describan, a partir de entonces, forman parte de ese proyecto. Por cada proceso del que nos informa un usuario, se crea un dDT dibujando los diferentes elementos que lo componen en la zona de la pantalla adecuada.
Se puede, básicamente:

- Manipular procesos.
- Manipular proyectos.
- Manipular y editar los elementos de un proceso (puestos de trabajo, documentos, tareas, agentes, flujos).
- Imprimir informes y diagramas.
- Comprobar y validar diagramas.

La Figura 4 muestra una imagen de la herramienta en funcionamiento.

## 5. Trabajo Futuro

Una vez expuesta la manera de construir y explotar los dDTs, resulta bastante evidente que si se dispone de una herramienta de apoyo que permita dibujar y sobre todo manipular de forma fácil y efectiva este tipo de diagramas, podemos obtener resultados exportables a otras herramientas CASE. Es necesario incluir en la herramienta las utilidades que permitan importar y exportar los diagramas de flujo de trabajo en alguno de los estándares de definición de *workflows* más extendidos [6].
Nuestro trabajo futuro se encuentra, precisamente, en ampliar la funcionalidad de este prototipo de modo que genere no solamente texto (catálogo de eventos) o versiones básicas de DFDs, sino versiones más depuradas y con mayor información de estos.
De forma general, nuestra línea de trabajo se está basando en las siguientes consideraciones: Para representar una lista de eventos se necesita únicamente un fichero de texto, en cualquier formato estándar, puesto que básicamente se trata de un producto interno de la herramienta.
Cada documento se debe definir con la mayor precisión posible, utilizando algún tipo de plantilla, de modo que su descripción se pueda convertir automáticamente en un requisito de información. Las tareas deben describirse de un modo similar, utilizando una plantilla que recoja las actividades que incluye, en un formato tabular y con la posibilidad de utilizar construcciones iterativas o alternativas. La propuesta de descripción de requisitos de Durán [3] se puede utilizar como guía.
Queda pendiente, además, medir la efectividad de dicha herramienta en el aprendizaje de los alumnos. El diseño del experimento para llevar a cabo esta tarea nos hace pensar en la comparación de dos tratamientos:

aprendizaje usando la herramienta y aprendizaje sin usarla (como se estaba haciendo antes de incorporarla). Esta tarea se desarrollará en los próximos meses.

## 6. Conclusiones

En este trabajo se ha presentado una propuesta metodológica que se está aplicando con resultados positivos en la docencia de la asignatura de Ingeniería del Software I, del primer cuatrimestre del tercer curso de la titulación de Ingeniería Técnica de Informática de Gestión, que permite la obtención y tratamiento de requisitos funcionales, independiente del paradigma de desarrollo posterior y con resultados de fácil adaptación al análisis estructurado, en particular.
Se trata de una técnica fácil de entender por los alumnos y los potenciales usuarios finales y que permite validar sobre la marcha las informaciones parciales que recibimos de estos. La segunda ventaja de la técnica es que permite simular varias alternativas de análisis, en particular las fronteras del nuevo sistema que se pretende construir. Para cada una de esas alternativas se pueden generar de forma automática los DFDs de las primeras fases del Análisis, con las evidentes ventajas que ello comporta.

## 7. Referencias


[1] **G. Booch, I. Jacobson, J. Rumbaugh, "***The Unified Modelling Language User Guide"*, Addison-Wesley, 1999.
[2] **A. Collongues, J. Huges, B. Laroche**, "*Merise. Methode de conception"*, Dunod, 1987.
[3] **A. Durán, B. Bernárdez, A. Ruiz, M. Toro,** "A Requirements Elicitation Approach Based in Templates and Patterns" in *WER'99 Proceedings*, Buenos Aires, 1999.
[4] **J.M. Guirles, "***Herramienta para el diseño de diagramas Documentos-Tarea"*, P.F.C., Universidad de Valladolid, 1998.
[5] **Ministerio de las Administraciones Públicas**, "*Metodología Métrica v.2.1"*, Madrid, 1995.
[6] **M.T. Schmidt,** "The evolution of workflow standards", *IEEE Concurr*. 7(3), 1999, pp. 44-52.
[7] **H. Tardieu, A. Rochfeld, R. Colleti**, "*La Methode Merise"*, Les Editions d'Organization,1983.
[8] **WfMC**, "*Workflow Management Coalition Terminology and Glosary (WFMC-TC-1011)"*, Technical Report, Workflow Management Coalition, disponible en http://www.wfmc.org, 1996.
[9] **E. Yourdon, "***Modern Structured Analysis"*, Prentice-Hall, 1989.