José Alfonso Aguilar Calderón

A goal-oriented approach for managing requirements in the development of Web applications

– PhD. Thesis –

Advisor: Irene Garrigós, Jose-Norberto Mazón López

Depto. Lenguajes y Sistemas Informáticos
Universidad de Alicante

During your life, never stop dreaming.
No one can take away your dreams.

Tupac Shakur.

There's gonna be some stuff that you gonna see
that's gonna make it hard to smile in the future, no doubt,
but through whatever you see, through all the rain and all the pain,
you gotta keep your sense of humor.

Tupac Shakur, de su canción "Smile for me now".

-ladran sancho!- Señal que cabalgamos!

De la película Don Quijote, de Orson Welles.

Agradecimientos

La tesis doctoral es el resultado de un sueño, un sueño que se nació de una simple conversación entre alumnos y profesores de la Facultad de Informática de la Universidad Autónoma de Sinaloa, cuando era solo un estudiante de Licenciatura. A partir de ese momento, nació en mí la ilusión de combinar la práctica profesional y la investigación. Paso a paso se va cumpliendo ese sueño, primero la Licenciatura, seguida de la práctica profesional compaginada con los estudios de Maestría y ahora el Doctorado. Se perfectamente que este es solo el comienzo, sin embargo, quiero agradecer a aquellas personas que han contribuído para que quién escribe estas líneas sea capaz de ir cumpliendo sus sueños paso a paso, gracias.....

A mis PADRES por haberme formado, por sus regaños, por sus consejos, por sus cuidados, por sus frases de aliento, por su ayuda cuando decidí salirme de casa para vivir en Culiacán, después en Alicante! y por todo el apoyo que me han dado para que pueda cumplir con mis objetivos profesionales y personales, esto es de ustedes.

A mis hermanos, Pedro, Pablo, Mirna, perdón por no estar ahí en reuniones, cumpleaños y momentos especiales, pero ante todo, gracias por su apoyo total.

A mi abuela, tios, primos, gracias por el ánimo y sus buenos deseos.

A mi tia Regina, gracias por sus oraciones y su apoyo. Regina, Jose Antonio, por todos los momentos compartidos, por los consejos, por hacerme sentir como en casa cuando estaba en Barcelona, por no dejarme sentir solo, gracias. Por último, por llevarme a ver al mejor jugador del mundo!!! y al mejor equipo (al momento de escribir la tesis).

A la familia Bailón Sanchez y Bailón Di Croce, Antonio, Mirtha, Alejandro, Eugenia, Lola, gracias por el apoyo y por todo lo que aprendí de la Argentina che's.

A mis amigos, que más de alguno faltará.. Abraham, Roberto, David, Julio, Angel, Mayra, Lupita, Deisy, Xinly, Adriana Z., Adriana, Gaby, Luis, Rafa, Irving, Raúl, Amhed, Gabriela, Blanca, Alex y Angeles...gracias por las palabras de aliento, buenos deseos, la despedida y los mariscos!!! Y gracias a tí shaparrita, por los ánimos!!!

A mis compañeros del grupo de investigación Lucentia del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Alicante, Alejandro, Lily, Jose Jacobo, por las discusiones de trabajo y toda la asesoría brindada. A Juan Carlos Trujillo, por incorporarme al grupo Lucentia, por asignarme a mis tutores y por el apoyo otorgado ante el CONACyT. A Paúl, Octavio por la discusión y análisis sobre la lógica de las transformaciones entre modelos. A Esteban Robles, por esas conversaciones filosóficas de la vida y acerca de ingeniería Web.

A los miembros del grupo de investigación IWAD del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Alicante, en especial a Santi, José Javier, Sergio, Juan Antonio por el apoyo recibido durante el período de investigación.

A mis directores de tesis Irene y Jose Norberto, xiquets!! gracias por guiarme durante este proceso, por lo que he aprendido de vosotros, por su mano firme en los momentos necesarios, por su predisposición permanente e incondicional en aclarar mis dudas, su tiempo, su dedicación, gracias, sin vosotros SIN DUDA ALGUNA esto no sería posible...sigamos adelante!!

Prefacio

En los últimos años, han surgido nuevas propuestas para abordar el desarrollo de aplicaciones Web, algunas de ellas se centran principalmente en la representación de la aplicación Web en un cierto nivel de abstracción (modelo conceptual), otros por su parte, se centran en tareas específicas de la proceso de desarrollo dejando de lado la fase de requisitos. Además, debido a la creciente complejidad de las aplicaciones Web (es decir, los cambios en la tecnología de implementación) y las múltiples audiencias implicadas en su uso (audiencia heterogénea), la fase de requisitos es más difícil de realizar y mantener. Como resultado, un problema surge en estas propuestas: la ausencia de una guía de diseño que facilite el desarrollo de las aplicaciones Web basada en las necesidades y expectativas del usuario.

Para superar la falta de dicho proceso, la tesis doctoral presenta una aproximación dirigida por modelos para el desarrollo de aplicaciones Web. En concreto, se propone la especificación de los requisitos de la aplicación Web un modelo conceptual basado en el marco de modelado orientado a objetivos *i**. Al realizarse la especificación de los requisitos en un modelo, la derivación automática de los modelos conceptuales que constituyen una aplicaciones Web es posible. Además, se presenta el soporte para la gestión de requisitos con lo cual será factible evitar problemas a nivel conceptual con (i) trazabilidad de requisitos, (ii) análisis de impacto y (iii) las decisiones de diseño que esten basadas en la maximización de los requisitos no-funcionales. Por último, como una prueba de concepto se ha implementado la propuesta en la plataforma *Eclipse* y se demuestra su aplicabilidad en un caso de estudio.

La tesis doctoral se compone de un conjunto de artículos publicados y algunos en proceso de revisión. Para escribir la tesis como una colección de artículos se deben de tener en cuenta una serie de requisitos según lo estipulado por la Universidad de Alicante. Con respecto al contenido, la tesis doctoral debe incluir un resumen, las hipótesis iniciales, los objetivos de la investigación y la colección de publicaciones. Finalmente, el resumen corresponde con la parte I de la tesis doctoral.

Es importante destacar que la tesis doctoral se ha desarrollado dentro del programa de doctorado *"Aplicaciones de la Informática"* del Departamento de Lenguajes y Sistemas Informáticos (DLSI) de la Universidad de Alicante. El trabajo doctoral ha sido financiado por el CONACYT (*Consejo Nacional de Ciencia y Tecnología, México*) y la Universidad Autónoma de Sinaloa, México.

Finalmente, la investigación se desarrolló bajo los siguientes proyectos: MANTRA (GRE09-17) de la Universidad de Alicante, SERENIDAD (PEII-11-0327-7035) de la Junta de Comunidades de Castilla La Mancha (España), MANTRA (GV/2011/035) del Ministerio de Valencia, MESOLAP (TIN2010 -14.860) del Ministerio Español de Educación y Ciencia y por el proyecto QUASIMODO (PAC08-0157-0668) del Ministerio de Educación y Ciencia de Castilla-La Mancha (España).

Alicante, Julio 2011 *José Alfonso Aguilar Calderón*

Índice

Parte I

Síntesis

1

Síntesis

A razón de que la tesis doctoral se ha realizado mediante la modalidad de compendio de artículos, este capítulo está dedicado a describir los objetivos, hipótesis y el conjunto de trabajos que la conforman. Además, es resumido el contenido científico de la tesis por medio de una síntesis global de los resultados obtenidos así como de las conclusiones finales.

1.1. Tesis Doctoral como Compendio de Artículos

Los requisitos que debe cumplir una tesis doctoral para ser realizada en la Universidad de Alicante mediante un compendio de publicaciones fueron definidos por el Pleno de la Comisión de Doctorado de fecha 2 de marzo de 2005. A continuación, se exponen aquellos directamente relacionados con el contenido de la tesis:

1. *"La tesis debe incluir una síntesis, en una de las dos lenguas oficiales de esta Comunidad Autónoma, en la que se presenten los objetivos, hipótesis, los trabajos presentados y se justifique la unidad temática."*
2. *"Esta síntesis debe incorporar un resumen global de los resultados obtenidos, de la discusión de estos resultados y de las conclusiones finales. Esta síntesis deberá dar una idea precisa del contenido de la tesis."*
3. *"Los trabajos deben ser publicados, o aceptados para la publicación, con posterioridad al inicio de los estudios de doctorado. Los artículos en periodo de revisión pueden formar parte de la tesis como apéndices del documento, que debe presentarse adjunta a los artículos publicados."*

Con el propósito de satisfacer los requisitos, la estructura de la tesis queda constituida en 3 partes. La primera parte (Parte I) consiste en una síntesis de la tesis. La Parte II presenta el conjunto de artículos publicados que forman el núcleo de la tesis. Finalmente, la Parte III consiste en un apéndice donde se presentan tres trabajos, los cuales se encuentran actualmente en proceso de revisión.

Asimismo, es muy importante subrayar que la tesis doctoral ha sido materializada gracias al apoyo económico otorgado por el Consejo Nacional de Ciencia y Tecnología (CONACyT) de México, por medio del Programa de Becas de Estudios de Posgrado en el Extranjero. Finalmente, es necesario destacar el interés y apoyo otorgado por parte de la Universidad Autónoma de Sinaloa México, a través del Programa de Formación de Recursos Humanos en Áreas Estratégicas.

1.1.1. Publicaciones Pertenecientes a la Tesis Doctoral

A continuación, se describen brevemente las cuatro publicaciones seleccionadas para que formen parte de la tesis doctoral. El criterio utilizado para la selección consistió en la relevancia y contribución científica de cada una de las publicaciones. Es decir, fueron seleccionados los

artículos publicados en revistas indexadas en JCR (*Journal Citation Report*[1]) y en congresos ubicados en la clasificación CORE (*Computer Research and Education*[2]).

Capítulo 2

J.A. Aguilar, I. Garrigós, J.-N. Mazón, J. Trujillo. Web Engineering Approaches for Requirements Analysis - A Systematic Literature Review. 6th Web Information Systems and Technologies (WEBIST 2010), April 7-10, 2010, Valencia, Spain. Vol. 2, pp. 187-190, 2010.

El capítulo presenta una revisión sistemática de la literatura con el fin de obtener el estado de la cuestión en lo referente a métodos para la especificación, análisis y modelado de requisitos en ingeniería Web así como las herramientas de soporte ofrecidas por cada uno de los métodos considerados. Los resultados obtenidos muestran, entre otras cosas, que gran parte de las metodologías Web no ofrecen un soporte integral en la etapa de análisis y especificación de requisitos.

Capítulo 3

J.A. Aguilar, I. Garrigós, J.-N. Mazón, J. Trujillo. An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering. Journal of Universal Computer Science (J.UCS), 16(17): 2475-2494 (2010).

El capítulo describe la propuesta base de la tesis, la cual consiste en el desarrollo de una metodología para la gestión de requisitos en ingeniería Web. En el capítulo anterior, se realizó una revisión sistemática de la literatura para estudiar las técnicas ingenieriles en el desarrollo de aplicaciones Web, los resultados demuestran que la mayoría de las aproximaciones se enfocan en las etapas de análisis y diseño, por tanto, no ofrecen un soporte integral a la fase de requisitos. La aproximación descrita en este capítulo, ha tomado como sustento las carencias detectadas en el capítulo anterior para desarrollar una aproximación basada en el marco de modelado orientado a objetivos *i** y en MDA (*Model-Driven Architecture*). La propuesta le permite al diseñador de la aplicación Web derivar la estructura de los modelos conceptuales que conforman la aplicación a partir de la especificación de requisitos.

Capítulo 4

J.A. Aguilar, I. Garrigós, J.-N. Mazón. Impact Analysis of Goal-Oriented Requirements in Web Engineering. The 11th International Conference on Computational Science and Its Applications (ICCSA 2011), June 20-23, 2011, Santander, Spain. Part V, Lecture Notes in Computer Science, Vol. 6786, pp. 421-436, 2011.

En capítulos anteriores se ha resaltado la importancia de la etapa de análisis y especificación de requisitos en la ingeniería Web, obligada, principalmente, por las características particulares de este tipo de aplicaciones, tales como su audiencia heterogénea y por la evolución constante en las tecnologías de implementación. Este tipo de características originan que la aplicación Web sea propensa a sufrir cambios, por eso, es importante conocer en qué medida impactarán los cambios a los requisitos, así como qué partes de la aplicación Web se verán afectadas. Para lograrlo, es necesario comprender y analizar las dependencias entre los requisitos, es decir, qué requisitos están relacionados o cuáles dependen uno del otro para cumplirse y con ello brindar soporte al diseñador por medio de una mejor gestión y mantenimiento de la aplicación Web.

Concretamente, en este capítulo se presenta un algoritmo para manejar las dependencias entre los requisitos funcionales y los requisitos no-funcionales de la aplicación Web en un contexto orientado a objetivos. Con el algoritmo, es posible comprender cuál es el impacto en los requisitos procedente de un cambio en la estructura de los modelos conceptuales que conforman la aplicación Web, así como saber qué requisitos necesitan ser implementados para

[1] http://www.thomsonreuters.com/
[2] http://www.core.edu.au/

cumplir con los propósitos establecidos en el análisis orientado a objetivos a la vez que se satisfacen los requisitos no-funcionales en la medida de lo posible.

Capítulo 5

J.A. Aguilar, I. Garrigós, J.-N. Mazón. A Goal-Oriented Approach for Optimizing Non-Functional Requirements in Web Applications. The 8th th International Workshop on Web Information Systems Modeling (WISM 2011), held in conjunction with the International Conference on Conceptual Modeling (ER 2011), 31 October - 03 November 2011, Brussels, Belgium. Lecture Notes in Computer Science, Vol. 6999, pp 14-23, 2011.

La idea de considerar a los requisitos no-funcionales desde las etapas tempranas del proceso de desarrollo con el fin de mejorar la calidad de la aplicación a desarrollar, ha sido objeto de investigación en el contexto del desarrollo dirigido por modelos. Para lograrlo, es necesario considerar a los requisitos funcionales así como a los no-funcionales desde la etapa de análisis y especificación de requisitos a razón de que los dos poseen la capacidad de satisfacer las necesidades de los *stakeholders*[3] y por lo tanto, ambos requisitos afectan la calidad de un producto de *software*, según lo establecido en la ISO/IEC 9126-1:2001[4], en la sección de *Product Quality*, específicamente en la parte de *Quality Model*.

En este capítulo se presenta una adaptación del algoritmo Optimización de Pareto para evaluar y seleccionar la configuración de requisitos óptima que maximice o balancee a los requisitos no-funcionales de la aplicación Web. La idea del capítulo se fundamenta en estudiar y mostrar cómo la implementación de los requisitos funcionales afecta o beneficia a los requisitos no-funcionales. Para esto, los requisitos no-funcionales deben de ser priorizados de acuerdo al contexto de los usuarios de la aplicación Web. Finalmente, la solución del algoritmo proporciona al diseñador de la aplicación un conjunto de configuraciones de entre las cuales podrá elegir qué requisitos funcionales implementar (configuración óptima) considerando la prioridad establecida por los *stakeholders* sobre los requisitos no-funcionales.

1.1.2. Artículos en Proceso de Revisión Pertenecientes a la Tesis Doctoral

En este apartado, se presentan tres trabajos que forman parte de la tesis doctoral y que actualmente se encuentran bajo proceso de revisión o envío.

Apéndice A

Requirements in Web engineering: a systematic literature review. Este artículo se ha enviado a la revista Journal of Web Engineering (JWE).

En el artículo se realiza una profunda revisión del estado de la cuestión en lo referente al análisis, especificación y trazabilidad de requisitos en ingeniería Web. Concretamente, se analizan: (i) las técnicas utilizadas por las metodologías ingenieriles en la etapa de análisis y especificación de requisitos, (ii) el tipo de requisitos y la terminología utilizada por cada metodología, (iii) el soporte para trazabilidad y (iv) las herramientas de soporte que ofrecen.

Cabe destacar que el artículo es una extensión del Capítulo 2, en el que se destaca la importancia de considerar a los requisitos en el desarrollo de sistemas Web. En particular, en la revisión sistemática de la literatura presentada en el apéndice, se ha mejorado el trabajo descrito en el Capítulo 2 de la siguiente forma:

- La estrategia de búsqueda ha sido mejorada, por lo tanto se han añadido más métodos a la revisión.
- El estudio se ha centrado en analizar el proceso de ingeniería de requisitos en el desarrollo de aplicaciones Web, es decir, en qué forma los requisitos son tratados en lo que respecta al análisis, especificación, validación y la gestión de los mismos.

[3] Las personas u organizaciones que afectan o son afectados directa o indirectamente por el proyecto de desarrollo de *software* en una forma positiva o negativa [62].

[4] http://www.iso.org/

- La revisión sistemática analiza el vocabulario que ha sido adoptado por cada método de ingeniería Web de una manera metódica y completa mediante el uso de la clasificación propuesta por Escalona y Koch [25].

Apéndice B

Dealing with dependencies among functional and non-functional requirements for impact analysis in Web engineering. Este artículo se ha enviado a International Journal of Open Source Software and Processes (IJOSSP).

El apéndice es una extensión del Capítulo 4 acerca de la importancia de considerar el análisis de impacto en nuestro método orientado a objetivos para el análisis y especificación de requisitos en Web. En particular, la novedad de la extensión consiste en: (i) la implementación de un metamodelo para adaptar el marco de modelado *i** al dominio Web, (ii) el desarrollo de un prototipo de herramienta para la especificación de requisitos Web como prueba de concepto de nuestra propuesta, (iii) la implementación de las reglas de transformación con un alto grado de automatización para derivar los modelos conceptuales de la aplicación Web, y (iv) la implementación de un algoritmo para analizar el impacto en el modelo de requisitos orientado a objetivos derivado de un cambio en la aplicación Web.

Apéndice C

A Goal-Oriented Requirements Engineering Approach to Distribute Functionality in RIAs. Para ser enviado a 24th International Conference on Advanced Information Systems Engineering (CAiSE 2012).

Como es sabido, las aplicaciones Web evolucionan constantemente a razón del desarrollo gradual de las tecnologías de implementación, entre otros factores [55]. Una pieza de esta evolución son las RIAs (*Rich Internet Applications*), las cuales ofrecen, entre otras cosas, una mejor interactividad con el usuario, similar a la ofrecida por las aplicaciones de *software* de escritorio. En este trabajo, se presenta la adaptación de la propuesta descrita en el Capítulo 5 para auxiliar al diseñador Web en la distribución de la funcionalidad de la RIA entre el cliente y el servidor. Para lograrlo, se adaptó el algoritmo Optimización de Pareto para obtener un conjunto de soluciones óptimas, de entre las cuales, de acuerdo con la prioridad establecida por parte del *stakeholder* sobre los requisitos no-funcionales, el diseñador de la aplicación Web será capaz de optimizar los requisitos no-funcionales mediante la distribución de los requisitos funcionales entre el cliente y el servidor.

1.1.3. Otras Publicaciones

En el transcurso de la investigación asociada a la tesis doctoral se han publicado cinco artículos en distintos eventos nacionales e internacionales. Cabe destacar que los trabajos no han sido incluídos en el núcleo de la tesis, sin embargo, complementan el progreso de la investigación. Los artículos son listados a continuación:

J.A. Aguilar, I. Garrigós, J.-N. Mazón. Aproximaciones en Ingeniería Web para el Análisis de Requisitos: una Revisión Sistemática de la Literatura. *Actas del IV Congreso Nacional de Informática y Ciencias de la Computación (CNICC 2009), Mazatlán, Sinaloa, México, 2009.*

J.A. Aguilar, I. Garrigós, J.-N. Mazón. Modelos de *weaving* para Trazabilidad de Requisitos Web en A-OOH. *Actas del VII Taller de Desarrollo de Software Dirigido por Modelos (DSDM 2010) en XV Jornadas de Ingeniería de Software y Bases de Datos (JISBD 2010), en conjunto con el Congreso Español de Informática (CEDI), pp. 146-155. SISTEDES, Valencia, España, 2010.* ISSN 19883455.

J.A. Aguilar, I. Garrigós, J.-N. Mazón. Modelo Requisitos y Modelo de Dominio, trazabilidad mediante modelos de *Weaving. Actas de VIII Jornadas para el Desarrollo de Grandes Aplicaciones de Red (JDARE 2010). GrupoM, Alicante, España, 2010.* ISBN: 978-84-614-3720-7.

J.A. Aguilar, I. Garrigós, J.-N. Mazón. Automatic Generation of Conceptual Models from Requirements Specification in Web Engineering using ATL. *Actas de IX Jornadas para el Desarrollo de Grandes Aplicaciones de Red (JDARE 2011). GrupoM, Alicante, España, 2011. Aceptado.*

J.A. Aguilar, I. Garrigós, S. Casteleyn, J.-N. Mazón. Una Propuesta Orientada a Objetivos para el Análisis de Requisitos en RIAs. *Actas de XVI Jornadas de Ingeniería de Software y Bases de Datos (JISBD 2011), pp. 211-224. La Coruña, España, 2011. ISBN: 978-84-9749-486-1.*

1.2. Hipótesis Inicial y Objetivos de Investigación

De forma similar a los sistemas *software* desarrollados exclusivamente para un entorno de escritorio, los sistemas Web necesitan la aplicación de conceptos de ingeniería para obtener éxito en la aplicación final. Para lograrlo, es necesario definir técnicas y enfoques que consideren la gran variedad de usuarios, plataformas y entornos para su implementación. En este sentido, la ingeniería de requisitos es uno de los factores de éxito más importantes en el desarrollo de *software* a razón de que permite descubrir, analizar, documentar y verificar los servicios que deben ser proporcionados por un sistema *software* junto con sus limitaciones operativas. De acuerdo con [49], el proceso de ingeniería de requisitos se divide en cinco pasos: elicitación, análisis, especificación, validación y gestión. La elicitación de los requisitos es el paso inicial en el proceso de desarrollo, tiene como objetivo descubrir qué problemas necesitan ser resueltos por medio del *software* y permite identificar a los *stakeholders*. El análisis de los requisitos permite entender la estructura organizacional, las reglas de negocio y las responsabilidades de los *stakeholders* con el fin de evitar conflictos entre ellos. La especificación de los requisitos, por su parte, consiste en una descripción integral del comportamiento del sistema *software* a desarrollar. La validación de los requisitos tiene el propósito de establecer si los requisitos son una representación exacta de las necesidades de los *stakeholders*. Finalmente, la gestión de requisitos es el proceso de comprender y controlar los cambios en los requisitos de la aplicación [62].

Sin embargo, en el desarrollo de *software* en ingeniería Web llevar a cabo un correcto proceso de ingeniería de requisitos es una tarea complicada. Principalmente, esto se debe a que la ingeniería Web enfrenta continuos cambios en lo que respecta a la tecnología de implementación los cuales dificultan las etapas del proceso de ingeniería de requisitos, como el análisis y la especificación, esto se debe a las características particulares de las aplicaciones Web, como el caso de: (i) la gran cantidad de información que ofrecen (contenido), (ii) el acceso a los diferentes escenarios donde ofrecen esa información (navegación), (iii) cómo proveer dicha información al usuario o grupos de usuarios (funcionalidad) del sitio Web y (iv), la audiencia heterogénea que tiene acceso a la Web. Como consecuencia de estos factores, los *stakeholders* en sus diferentes roles (analistas, desarrolladores y diseñadores) se enfrentan a retos cada vez más complejos para gestionar el diseño y mantenimiento de las aplicaciones Web. Por lo tanto, definir los requisitos que el sistema debe cumplir para satisfacer las necesidades de los usuarios es una tarea que necesita ser priorizada.

Actualmente, existen una notable cantidad de métodos para el desarrollo de aplicaciones Web (A-OOH [27], UWE [37], NTD [23], OOWS [52], etc.) que toman en cuenta la aplicación de distintas técnicas para llevar a cabo el proceso de desarrollo, la mayoría de ellas utilizan reconocidas técnicas de ingeniería de *software* para gestionar correctamente los requisitos de los usuarios, como el caso de UWE que utiliza casos de uso [38]. Lamentablemente, la mayoría de las técnicas utilizadas por las metodologías resultan insuficientes para representar características muy particulares de las aplicaciones Web, tales como: la navegación y la audiencia heterogénea. Por lo tanto, es necesaria la inclusión de nuevas técnicas que permitan lidiar con las características particulares de las aplicaciones Web y que además posibiliten la correcta especificación de las necesidades de los diferentes actores implicados.

Por otro lado, la ingeniería dirigida por modelos (*Model-Driven Engineering*, MDE) es una metodología de trabajo que incorpora un conjunto de métodos, técnicas y tecnologías

para llevar a cabo el ciclo de vida del proyecto de desarrollo en base a modelos. MDE guía el proyecto de desarrollo en base a conceptos y reglas tomados directamente de una área de interés determinada, es decir, del dominio del problema [59]. Para esto, MDE incorpora conceptos de la ingeniería de *software* e ingeniería de requisitos, tales como los modelos de madurez [61], los marcos de trabajo conceptuales como las metodologías ágiles, trazabilidad de los productos de trabajo[5] derivados del proceso de desarrollo [61] y la evolución del *software* [62].

No obstante, el desarrollo dirigido por modelos (*Model Driven Development*, MDD), se manifiesta como un paradigma de desarrollo que abstrae MDE mediante la utilización de modelos como artefactos principales en el proceso de desarrollo de *software*. MDD se ha convertido en una alternativa valiosa para resolver los problemas asociados con el desarrollo de *software* de manera sistemática, estructurada, integrada y completa por medio del modelado del sistema *software* y su generación a partir de los modelos [2]. Es importante resaltar que MDD sólo proporciona una estrategia general a seguir en el desarrollo de *software* dirigido por modelos, pero no define las técnicas a utilizar.

En este contexto, la arquitectura dirigida por modelos (*Model Driven Architecture*, MDA) [47] surge como un estándar del OMG (*Object Management Group*) [51] que promueve el MDD. MDA está formada por un conjunto de capas y transformaciones que proporcionan un marco conceptual de trabajo en donde encontramos tres tipos de modelos, el primero de ellos es modelo independiente de la computación (*Computational Independent Model*, CIM), utilizado para la especificación de los requisitos de la aplicación a desarrollar, el segundo es el modelo independiente de la plataforma (*Platform Independent Model*, PIM), como su nombre lo indica, se caracteriza por ser independiente de la plataforma de implementación de la aplicación, por ejemplo, un diagrama de clases, y el modelo específico de la plataforma (*Platform Specific Model*, PSM), el cual es obtenido del PIM y contiene la información sobre una plataforma de desarrollo o alguna tecnología en específico donde será implementada la aplicación final, por ejemplo, el código fuente de la aplicación [47]. El supuesto básico de MDA es la consideración de los modelos como entidades de primera clase que impulsan el proceso de desarrollo desde el análisis de requisitos hasta la implementación final. Básicamente, cada paso del proceso consiste en la generación de uno o más modelos de salida a partir de uno o más modelos de entrada. Por lo tanto, las transformaciones entre modelos son la clave para completar cada paso del proceso de desarrollo dirigido por modelos.

MDA ha tenido un gran impacto en la comunidad de ingeniería Web, esto es debido a las ventajas que ofrece llevar a cabo el proceso de desarrollo de aplicaciones Web mediante el uso de modelos conceptuales, por ejemplo, acortar el tiempo de desarrollo de la aplicación Web, lo cual puede resultar en un ahorro en el costo del proyecto. El impacto de MDA en la ingeniería Web ha permitido la llegada de la ingeniería Web dirigida por modelos (*Model Driven Web Engineering*, MDWE) como una nueva aproximación para el desarrollo de aplicaciones Web [39, 48].

En la actualidad, MDA ha sido utilizado para el desarrollo de aplicaciones Web por parte de ciertas metodologías tales como OOWS [57], NDT [24] y UWE [41]. Lamentablemente, a pesar que se ha resaltado la importancia de la etapa de análisis de requisitos en Web en distintos trabajos de gran trascendencia, por ejemplo en [25], algunas metodologías dan poca importancia a la etapa de análisis y especificación de requisitos (OOWS [52], OOHDM [60], WSDM [18] y HERA [15]) y otras, por su parte, consideran la especificación de requisitos a nivel CIM de MDA utilizando técnicas como los casos de uso (UWE [37]) y plantillas textuales [24], técnicas que resultan insuficientes en algunos casos para representar características de las aplicaciones Web como la navegación cuando se trata de aplicaciones muy complejas [66]. En este aspecto, el trabajo presentado en la tesis doctoral aborda el modelado conceptual de aplicaciones Web a partir del nivel CIM de MDA utilizando técnicas orientadas a objetivos (por medio del marco de trabajo *i** [33]) [4].

El uso del enfoque orientado a objetivos (*Goal-oriented Requirements Engineering*, GORE) para la especificación de requisitos permite la elaboración, estructuración, especificación, análi-

[5] Un producto de trabajo es cualquier artefacto producido por un proceso. Estos artefactos pueden incluir archivos, documentos, piezas de los productos, servicios, procesos y especificaciones [34].

sis, negociación, documentación y la modificación de los requisitos [6]. Tal uso se basa en un modelo visual que muestra cómo los objetivos organizacionales, los actores, los escenarios, las tareas y propiedades del dominio están relacionados entre sí y gracias a eso es posible analizar cómo los actores efectúan las tareas necesarias para que los objetivos organizacionales puedan ser alcanzados [68].

A razón de que MDWE utiliza modelos como artefactos principales en el proceso de desarrollo, es posible la obtención, con alto grado de automatización, de la estructura de los modelos conceptuales a nivel PIM directamente de un modelo de requisitos orientado a objetivos. Una de las principales ventajas es que se asegura que los modelos conceptuales obtenidos sean correctos semánticamente y que reflejen los requisitos modelados en el CIM. Además, gracias al análisis orientado a objetivos, es posible ofrecer al diseñador soporte para la gestión de los requisitos, por ejemplo, la visualización de la trazabilidad. Asimismo, gracias a la visualización de las relaciones entre los elementos del modelo, es posible analizar el impacto derivado de la eliminación de una tarea u objetivo.

La **hipótesis de partida** de la investigación asociada a la tesis doctoral consiste en que sí es factible la aplicabilidad del enfoque orientado a objetivos en el desarrollo de una metodología MDD-MDA que contemple una etapa integral para el análisis y especificación de requisitos que asista al diseñador en: (i) la comprensión de los objetivos y expectativas de la audiencia heterogénea de una aplicación Web, (ii) brinde soporte en algunos aspectos de la gestión de los requisitos como la trazabilidad y el análisis de impacto y (iii) sea capaz de proveer un conjunto de alternativas de diseño basadas en la prioridad de los requisitos no-funcionales.

El **objetivo de investigación** de la tesis doctoral es la propuesta de un método para el análisis y especificación de requisitos para aplicaciones Web, que considere:

1. Una etapa de análisis de requisitos orientada a objetivos para representar las expectativas reales de los *stakeholders* de la aplicación Web y las relaciones entre ellos.
2. Mecanismos para la comprensión de los objetivos de negocio, los cuales, gracias al uso del análisis de requisitos orientada a objetivos, deben de ser alcanzados por medio de la aplicación Web.
3. Un alto grado de automatización en el desarrollo de aplicaciones Web, por medio de un conjunto de transformaciones para obtener la estructura de los modelos conceptuales a partir de la especificación de los requisitos y con esto brindar soporte a la generación del código de la aplicación Web.
4. Soporte integral para la gestión de los requisitos en aspectos como la trazabilidad y el análisis de impacto. Esto permitirá verificar que los requisitos han sido reflejados en la aplicación final, así como analizar el impacto en los requisitos a razón de la evolución de la aplicación Web.
5. Asistir al diseñador al momento de la selección de los requisitos funcionales a implementar a través de alternativas de diseño que consideren el balance y maximización de los requisitos no-funcionales, de esta forma los requisitos no-funcionales son considerados desde la etapa de análisis y especificación de requisitos con el fin de mejorar la calidad de la aplicación Web a desarrollar.

1.3. Resumen del Contenido de la Tesis Doctoral

El objetivo de investigación de la tesis doctoral se aborda en dos etapas, la primera es la definición de una propuesta orientada a objetivos para el análisis y especificación de requisitos en ingeniería Web así como un conjunto de transformaciones para la obtención de la estructura de los modelos conceptuales de la aplicación Web. La segunda, consiste en la especificación y aplicación de técnicas para la gestión de requisitos, concretamente, aquellas relacionadas con la trazabilidad de requisitos y el análisis de impacto, además de ofrecer alternativas de diseño en base a la maximización de requisitos no-funcionales al diseñador Web.

La Figura 1.1 muestra la visión global de la propuesta que se presenta en la tesis doctoral. La propuesta consiste en un método para al análisis y especificación de requisitos en ingeniería

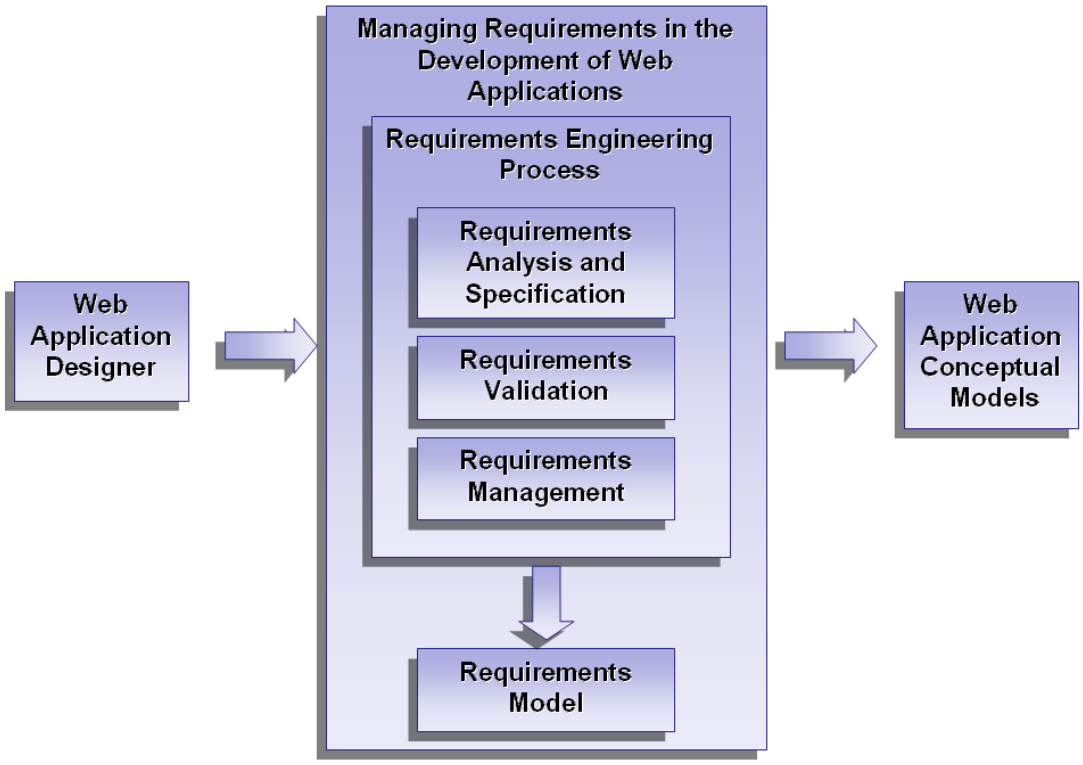


Figura 1.1. Visión global de la propuesta de la tesis doctoral.

Web. Como se puede ver en la imagen, al final del proceso se obtendrá un modelo conceptual con la especificación de los requisitos del cual se podrán obtener las estructuras de los modelos conceptuales que conforman una aplicación Web.

1.3.1. Una Aproximación Orientada a Objetivos para el Análisis de Requisitos en Ingeniería Web

En este apartado, se resume la propuesta para el análisis y especificación de requisitos en ingeniería Web aplicada por medio del marco de modelado orientado a objetivos *i** y el método de ingeniería Web A-OOH (*Adaptive Object-Oriented Hypermedia*) [4, 26, 27]. En el Capítulo 3 se ofrece una explicación más detallada del método Web A-OOH.

A-OOH [26], es la extensión del método OOH (*Object-Oriented Hypermedia*) [30] con soporte de personalización. El proceso de desarrollo de A-OOH está basado en MDA [47], es decir, la aplicación Web es obtenida a partir de una serie de modelos conceptuales. Los modelos conceptuales corresponden al nivel PIM de la arquitectura MDA como puede verse en la Fig. 1.2, estos son:

- **Modelo de dominio**. El modelo de dominio de A-OOH se expresa como un diagrama de clases UML (*Unified Modeling Language*) [64]. El modelo refleja la parte estática de la aplicación Web encapsulando su estructura y funcionalidad. Los elementos principales para el modelado de un diagrama de clases son las clases (con sus atributos y operaciones) y sus relaciones.
- **Modelo de navegación**. El modelo de navegación de A-OOH se compone de nodos de navegación y las relaciones entre ellos. El modelo indica los caminos de navegación que el usuario puede seguir en la Web (enlaces de navegación). Hay tres tipos de nodos: (i) clases navegacionales (que son vistas parciales de las clases de dominio), (ii) destinos navegacionales (que agrupan elementos del modelo que colaboran en el cumplimiento de uno

o más requisitos de navegación del usuario) y (iii) colecciones (que son estructuras, posiblemente jerárquicas, que se definen entre clases de navegación o destinos navegacionales). La colección más común es la colección de clasificación (*C-collection*), que actúa como un mecanismo de abstracción para el concepto de *menú* agrupando enlaces de navegación. Con respecto a los enlaces de navegación, A-OOH define dos tipos principales: enlaces de travesía (*Transversal-links*) (definidos entre dos nodos de navegación) y enlaces de servicio (*Service-links*), en donde la navegación sucede al activar una operación que modifica la lógica de negocio y además implica la navegación a un nodo mostrando información cuando la ejecución del servicio ha finalizado.

- **Modelo de presentación**. El modelo permite definir la interfaz gráfica de la aplicación Web, por ejemplo, el tipo de fuente utilizada en el texto, el color, etc.
- **Modelo de personalización**. El modelo es utilizado para la especificación de estrategias de personalización.
- **Modelo de usuario**. Permite la descripción de los usuarios en términos de información personal, sus relaciones con un dominio en particular y las acciones de navegación realizadas en tiempo de ejecución. El modelo de usuario también describe la estructura de la información necesaria para las estrategias de personalización.

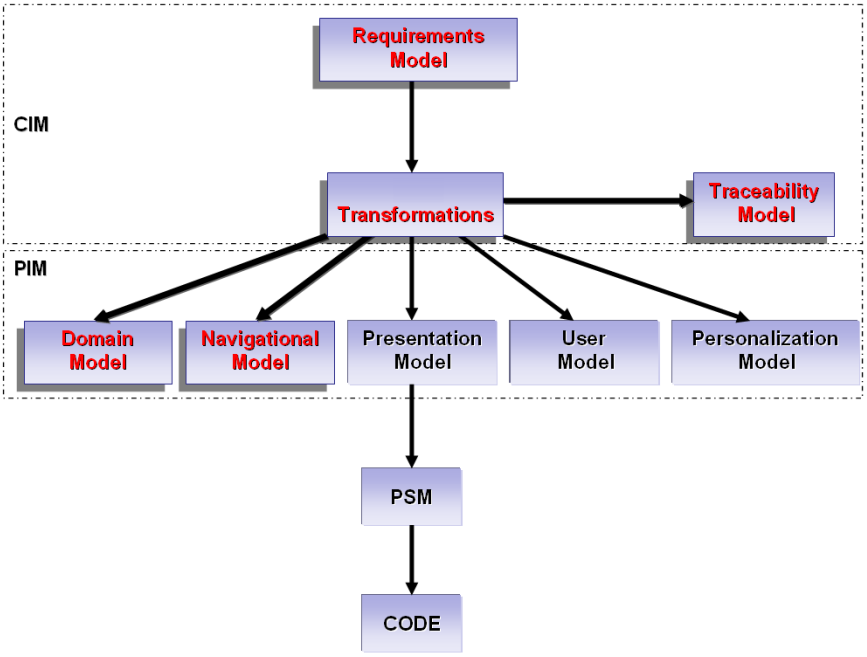


Figura 1.2. La propuesta de la tesis doctoral integrada como CIM en el método Web A-OOH.

La Fig. 1.2 muestra la ubicación de la propuesta de la tesis doctoral en la arquitectura MDA del método Web A-OOH. Los aportes de la tesis se encuentran resaltados en la imagen, estos son el modelo de requisitos y la generación de los modelos conceptuales de dominio, navegación y trazabilidad con sus respectivas reglas de transformación. Por lo tanto, la propuesta presentada en la tesis se ubica en el nivel CIM de MDA.

Especificación del Modelo de Requisitos a Nivel CIM

El primer paso de la propuesta es la especificación y el modelado de los requisitos de la aplicación Web. Para una explicación más amplia el lector puede referirse al Capítulo 3 de la tesis doctoral.

Los requisitos Web se definen en un modelo independiente de computación (CIM) utilizando el marco de modelado de requisitos *i**. El marco de modelado *i** [73, 74] es uno de los más utilizados para analizar los objetivos de los *stakeholders*, nombrados actores en *i**, y cómo el sistema a diseñar debería satisfacerlos. Además, *i** permite razonar acerca de cómo estos objetivos pueden contribuir a la selección de diferentes alternativas de diseño según su viabilidad. Con el fin de motivar esta parte de la investigación, se ha realizado una revisión del estado de la cuestión (ver Capítulo 2 y Apéndice A) acerca de las técnicas utilizadas para el análisis y especificación de requisitos en ingeniería Web.

El marco de modelado *i** consiste básicamente en dos modelos: el modelo SD (*Strategic Dependency*) para especificar las relaciones de dependencia entre varios actores en un contexto organizacional y el modelo SR (*Strategic Rationale*), utilizado para describir los intereses y preocupaciones del actor y cómo podrían abordarse. El modelo SR permite el modelado de las relaciones de asociación entre cada actor y sus dependencias, por tanto, proporciona información acerca de cómo los actores llegan a sus objetivos. El modelo SR incluye sólo los elementos considerados lo suficientemente importantes como para influir en el alcance de un objetivo. El modelo SR muestra las dependencias de los actores mediante la inclusión del modelo SD. En torno a estas dependencias, el modelo SR especifica elementos intencionales, tales como objetivos (*Goals*), tareas (*Tasks*), recursos (*Resources*) y *Softgoals* (Tabla 1.1). En comparación con el modelo SD, los modelos SR proporcionan un nivel de modelado más detallado debido a que permiten modelar las relaciones internas e intencionales de cada actores. Las relaciones intencionales son enlaces del tipo *Means-end* y representan formas alternativas para satisfacer objetivos; los *Decomposition-links* representan los elementos necesarios para que una tarea sea realizada; o los *Contribution-links* que sirven para modelar cómo un elemento intencional contribuye a la satisfacción de una *Softgoal*. En la Tabla 1.1 se describen los elementos más importantes del marco de modelado *i**, para información más extendida el lector puede consultar [33, 73, 74] y el Capítulo 3.

Tabla 1.1. Principales elementos para el modelado en *i**

Elemento	Icono	Descripción
Actor		Es una entidad que lleva a cabo acciones para cumplir con sus objetivos. Se relaciona con varios elementos intencionales (objetivo, tarea o recurso).
Goal		Representa una condición o estado que el actor le gustaría alcanzar. Sin embargo, no detalla como se satisfará el objetivo.
Task		Representa una manera particular de hacer algo (alcanzar un objetivo).
Softgoal		Representan criterios de calidad. Son utilizadas cuando los objetivos del *stakeholder* no son precisos o sus criterios de éxito no están claramente definidos.
Resource		Es una entidad que debe estar disponible para su uso (datos).
Means-ends		Son asociaciones que describen cómo se alcanzan los objetivos, es decir, los posibles caminos para satisfacer un objetivo.
Decomposition		Son asociaciones que definen elementos adicionales necesarios para llevar a cabo una tarea.
Contribution		Son asociaciones que sirven para modelar cómo los elementos intencionales contribuyen a la satisfacción de una *Softgoal*.

Por otra parte, el marco de modelado *i** resulta insuficiente para representarlas por sí solo las características particulares de las aplicaciones Web, tales como el contenido que ofrecen, la navegación, la funcionalidad y la audiencia heterogénea (para más información consultar Capítulo 2 y Apéndice A). Por tal motivo, *i** debe de adaptarse al dominio Web para poder ser utilizado en la especificación y análisis de requisitos. Esto permitirá modelar a los *stakeholders* con sus objetivos y las relaciones existentes entre ellos para satisfacerlos. Para realizar la adaptación del marco de modelado *i** al dominio Web nuestra propuesta utiliza la taxonomía de requisitos Web presentada en [25], la cual clasifica a los requisitos Web en seis tipos. Estos son descritos a continuación:

- **Requisitos de contenido (*Content*)**. Con este tipo de requisitos se define el contenido que el sitio Web presenta a sus usuarios. Considerando como ejemplo base una librería *on-line*, algunos ejemplos pueden ser: "información del libro" o "categorías del producto".
- **Requisitos de servicio (*Service*)**. Este tipo de requisito hace referencia a la funcionalidad interna que la aplicación Web debe proveer a los usuarios. Continuando con el ejemplo de la librería *on-line*, ejemplo de requisitos de servicio son: "registrar un nuevo cliente", "agregar un producto", etc.
- **Requisitos de navegación (*Navigational*)**. Un sistema Web debe también definir caminos de navegación disponibles para los usuarios. Algunos ejemplos, en base a la librería *on-line* son: "consultar productos por categoría", "consultar el carrito de la compra", etc.
- **Requisitos de interfaz (*Layout*)**. Los requisitos también pueden definir la interfaz visual para los usuarios. Por ejemplo: "el color de fondo" y "el uso de *frames*".
- **Requisitos de personalización (*Personalization*)**. El diseñador puede especificar las acciones de personalización a ser ejecutadas en el sitio Web. Por ejemplo: "adaptar el estilo de la fuente para las personas con deficiencia visual".
- **Requisitos no funcionales (*Non-functional requirements*)**. Estos requisitos representan criterios de calidad que el sistema debe conseguir. Algunos ejemplos de estos requisitos pueden ser: "eficiencia", "atraer más usuarios" y "buena experiencia del usuario".

Finalmente, para poder utilizar el marco de modelado *i** dentro de MDA se ha implementado un metamodelo. El metamodelo se implementó utilizando la tecnología EMF (*Eclipse Modeling Framework*) de Eclipse [21] (Fig. 1.3). Los elementos intencionales de *i** fueron extendidos con nuevas clases para representar cada uno de los tipos de requisitos Web descritos anteriormente (*Navigational*, *Service*, *Personalization*, *Layout* y *Content*), es decir, los requisitos *Navigational*, *Service*, *Personalization* y *Layout* son instancias del elemento *Task* del marco de modelado *i** de tal forma que se representarán visualmente por medio de la figura (⬭). El requisito *Content* es una instancia del elemento *Resource*, por lo que será representado por la forma (▭) de *i**.

Por último, es importante destacar que los requisitos no-funcionales de la aplicación Web se modelan directamente utilizando el elemento *Softgoal*, por tanto, cuando se lleve a cabo la especificación de requisitos utilizando esta propuesta, el concepto de requisito no-funcional corresponderá al concepto de *Softgoal* del marco de modelado *i** y se representará visualmente con el icono (☁).

A continuación, se resume la generación de la estructura de los modelos conceptuales de dominio y navegación de A-OOH. Se remite al lector a los capítulos específicos de la tesis doctoral para una explicación más detallada (Capítulo 3).

Generación de los Modelos Conceptuales a nivel PIM

Definidos los requisitos en un CIM, el siguiente paso consiste en utilizarlos para derivar la estructura de los modelos conceptuales de dominio y navegación de la aplicación Web. Para lograrlo, es necesario que los modelos cumplan con la sintaxis abstracta de un dominio específico, es decir, que sean conformes a un metamodelo. A continuación se describen los metamodelos utilizados para la derivación de los modelos conceptuales a nivel PIM de MDA. Es importante mencionar que en la tesis doctoral se han considerado únicamente los modelos de

dominio y navegación del método Web A-OOH con el fin de mostrar la utilidad de la propuesta pero es fácilmente extensible al resto de los modelos conceptuales.

UML (*Unified Modeling Language*) [64] es el lenguaje de modelado estándar utilizado para la derivación del modelo de dominio de A-OOH. El metamodelo de UML describe los objetos, atributos y relaciones necesarias para representar los conceptos de UML dentro de una aplicación de *software*. Los modelos estáticos son presentados en diagramas llamados Diagramas de Clases. El propósito de un diagrama de clases es el de representar a las clases de un dominio en específico dentro de un modelo. Las clases tienen atributos (variables miembros), operaciones (funciones miembro) y relaciones con otras clases. Estas características aplican perfectamente para la representación del modelo de dominio de A-OOH.

Por otra parte, A-OOH dispone de un metamodelo para representar las rutas de navegación que el usuario puede seguir durante su interacción con la aplicación Web [26]. En la Figura 1.4 se muestra el metamodelo de navegación utilizado en A-OOH. Los elementos principales del metamodelo son: (i) Nodo Navegacional (*Navigational Node*) y (ii) Enlace Navegacional (*Navigational Link*).

El Nodo Navegacional representa vistas restringidas de los conceptos del dominio y sus relaciones indican las rutas de navegación que el usuario de la aplicación Web puede seguir. Existen tres tipos diferentes de Nodos Navegacionales, los cuales se describen a continuación:

- **Clases de Navegación** (*Navigational Classes*), son clases del dominio enriquecidas con atributos y métodos cuya visibilidad ha sido restringida, dependiendo de los permisos de acceso del usuario y de los requisitos de navegación. Es representada por una clase UML estereotipada como *NavigationalClass*.
- **Objetivos de Navegación** (*Navigational Targets*), agrupan los elementos del modelo que colaboran en el cumplimiento de cada requisito de navegación del usuario. Son representados utilizando la notación UML de paquetes con el estereotipo *NavigationalTarget*.
- **Colecciones** (*Collections*), son estructuras jerárquicas definidas en Clases Navegacionales u Objetivos Navegacionales. Proveen al usuario de la aplicación Web nuevas formas de accesar a la información. La colección más común es *C-Collection* (*Clasifier Collection*), la cual actúa como un mecanismo de abstracción para el concepto de *menú*, agrupa de esta forma Enlaces Navegacionales (*Navigational Links*). Otra colección importante es la llamada *S-Collection* (*Selector Collection*) mediante la cual podemos representar un mecanismo de selección. Las colecciones son representadas por medio de una clase UML estereotipada como *NavigationalC-Collection* o *NavigationalS-Collection*.

El Enlace Navegacional define las rutas de navegación que el usuario puede seguir a través de la aplicación Web. A-OOH en su metamodelo de navegación define dos tipos principales de enlaces (*links*):

- **T-Links** (*Transversal Links*), son enlaces definidos entre dos nodos navegacionales (clases navegacionales, colecciones u objetivos navegacionales). La navegación es realizada para mostrar información a través de la interfaz del usuario sin modificar en absoluto la lógica de negocio. Son representados por el estereotipo *TransversalLink*.

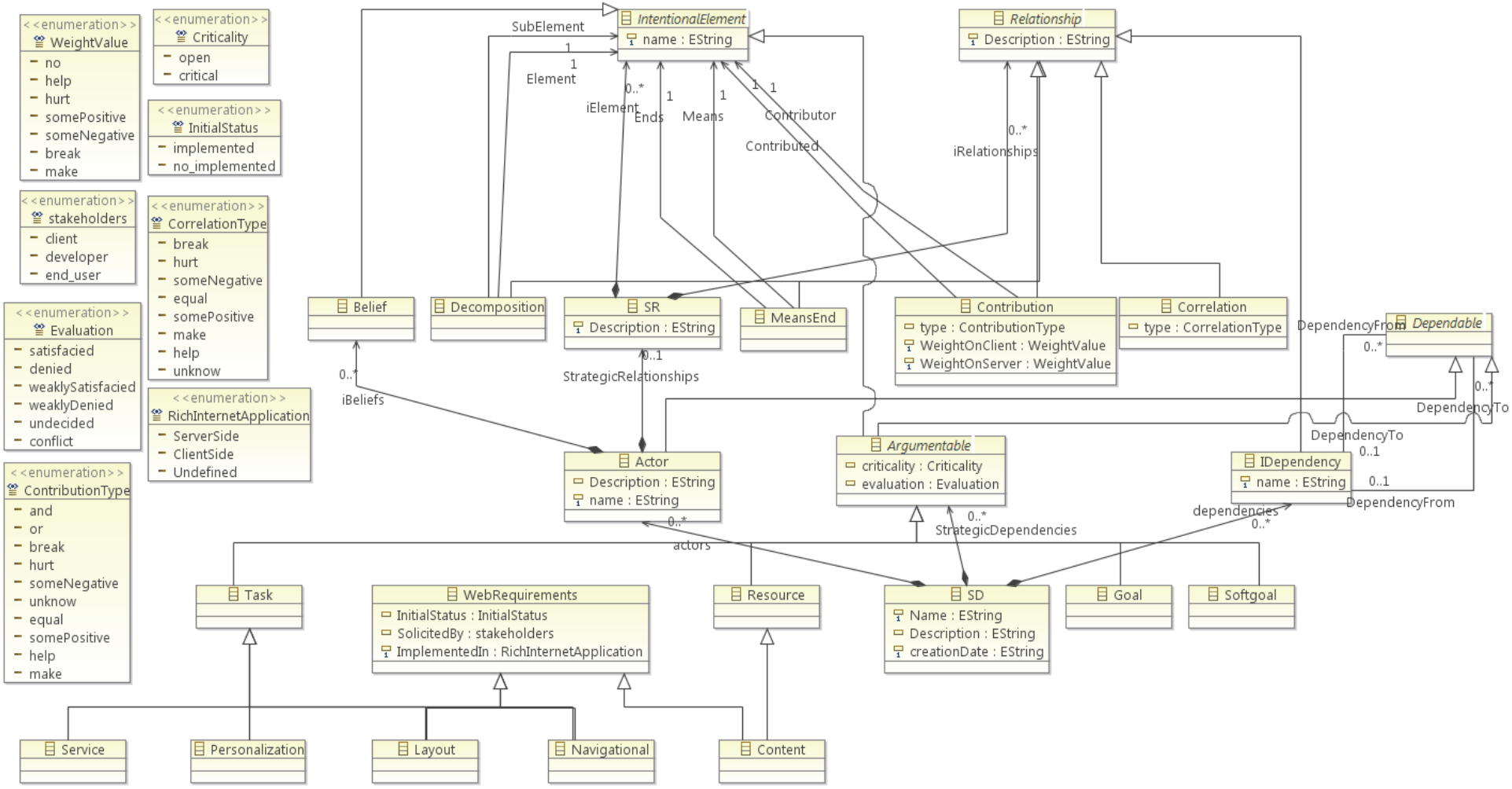


Figura 1.3. Metamodelo para la especificación de requisitos Web con *i**.

- **S-Link** (*Service Links*), estos tipos de enlaces activan una operación, la cual, en forma opuesta a los *T-Links* modifica la lógica del negocio y además implica que la navegación a un nodo muestre información cuando termine la ejecución del servicio. Se establece cuando un servicio de la clase navegacional es activado. Estos tipos de ligas son representados por el estereotipo *ServiceLink* y tiene asociado el nombre del servicio que lo invocó.

A partir de los metamodelos de dominio y navegación de A-OOH, se pueden definir sus respectivos modelos. El modelo de dominio en A-OOH encapsula la estructura y funcionalidad de los conceptos relevantes del dominio de la aplicación Web y también refleja la parte estática de la misma, se representa como un diagrama de clases en notación UML [64]. El objetivo consiste en obtener el esqueleto del modelo de dominio de A-OOH a partir del modelo de requisitos en *i** por medio de un conjunto de reglas de transformación modelo a modelo (M2M) descritas por medio del lenguaje QVT (*Query/View/Transformation*)[58] (ver Capítulo 3 y Apéndice B).

El lenguaje QVT es un estándar propuesto por el OMG para la definición de transformaciones formales y automáticas entre modelos. Definir transformaciones utilizando reglas QVT tiene varias ventajas, por ejemplo, las transformaciones son establecidas en un modelo por lo que son fáciles de comprender, utilizar, reutilizar, mantener y pueden ser integradas fácilmente en un método basado en MDA. QVT se divide en dos partes: declarativo e imperativo. La parte declarativa del lenguaje provee mecanismos para definir las relaciones que deben mantenerse entre los elementos del modelo, el cual pertenece a un conjunto de modelos candidatos, es decir, modelos origen (entrada) y destino (salida). La parte imperativa permite definir mapeos operacionales para extender la parte declarativa de QVT con implementaciones imperativas cuando resulta difícil especificar una relación puramente declarativa.

En la tesis doctoral nos centramos en la capa de relaciones de QVT. Esta capa permite la especificación de las relaciones que se debe mantener entre los modelos a través de un lenguaje de relaciones. Una relación se define por los siguientes elementos:

- Dos o más dominios: cada dominio es un conjunto de elementos de un metamodelo relacionados con un modelo de entrada o de salida. El tipo de relaciones entre dominios debe de ser especificado de dos formas: *checkonly* (C), es decir, que sólo se comprueba si la relación se mantiene o no y *enforced* (E), es decir, el modelo de salida puede ser modificado para satisfacer la relación.
- Cláusula When: establece las condiciones que debe de cumplir la relación (pre-condición) para poder realizarse.
- Cláusula Where: se utiliza para establecer las condiciones que deben cumplirse por todos los elementos de los modelos que participan en la relación (post-condición).

A continuación se explican las relaciones QVT establecidas para la generación del modelo de dominio de A-OOH a partir del modelo de requisitos *i**. Las transformaciones están definidas en una sola dirección, es decir, del modelo de requisitos *i** (modelo de entrada) al modelo de dominio de A-OOH (modelo de salida). Más información acerca de las relaciones QVT se encuentra en el Capítulo 3.

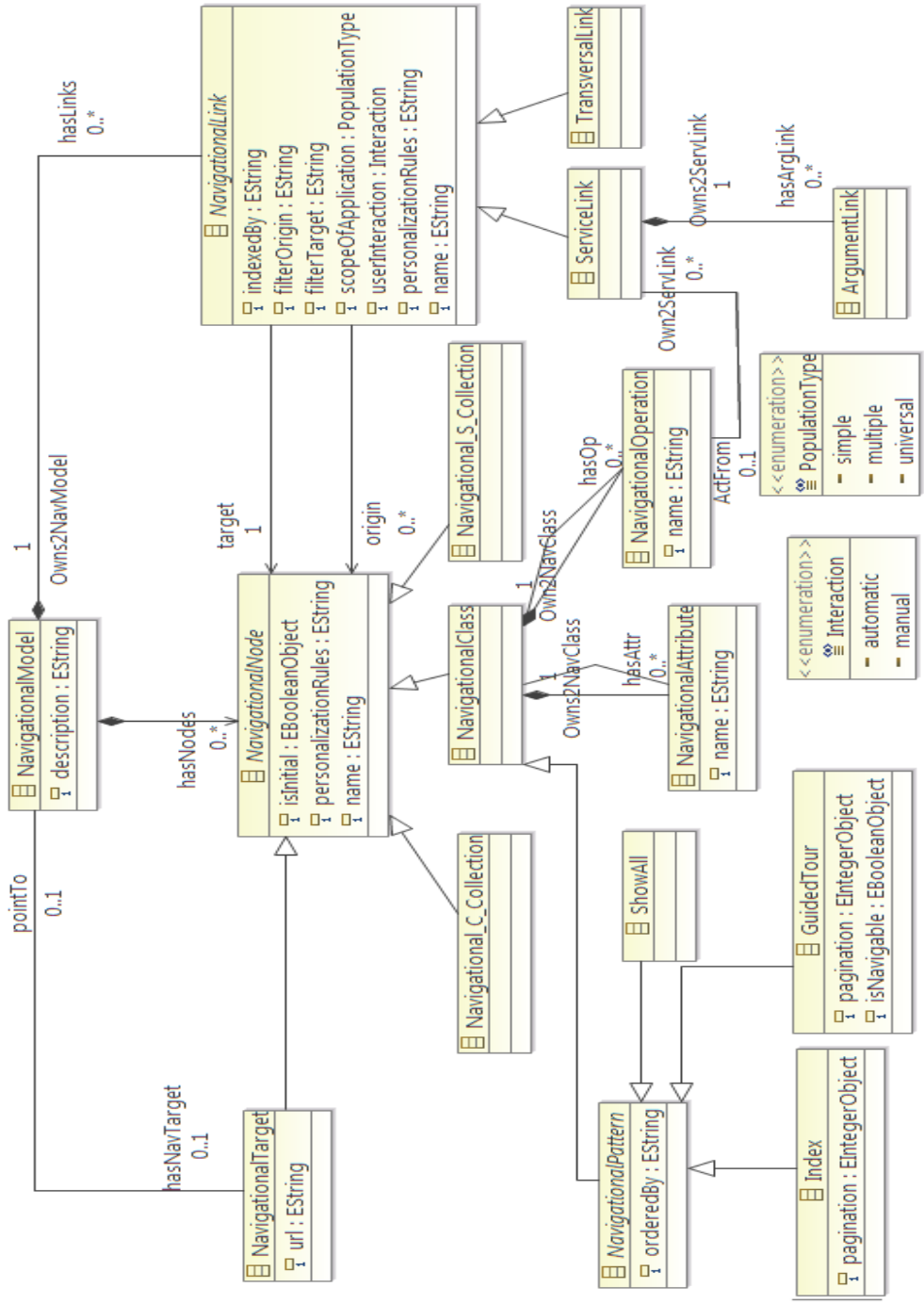


Figura 1.4. Metamodelo de Navegación de A-OOH.

- ***Content2DomainClass***. Esta regla de transformación permite crear las clases del modelo de dominio de A-OOH. Para esto, la regla detecta en el dominio origen (modelo de requisitos) el conjunto de elementos que representan un requisito del tipo *Content* y por cada uno de los elementos encontrados en el dominio origen, se creará una clase tipo *Class* UML en el dominio destino (modelo de dominio) (Figura 1.5).

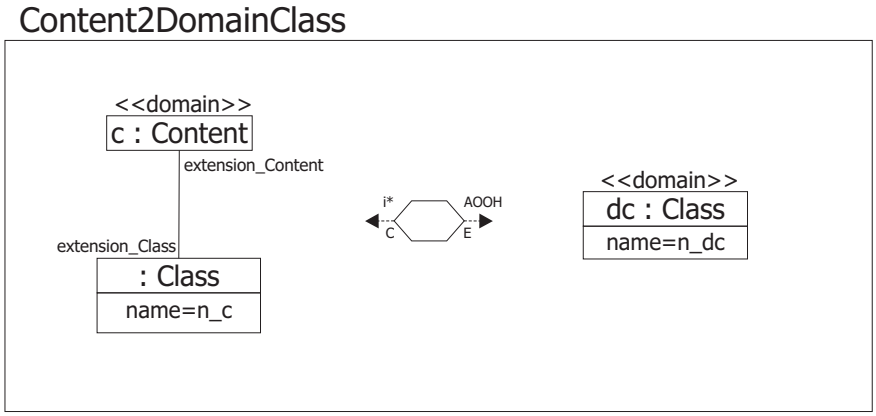


Figura 1.5. Regla QVT para obtener las clases del modelo de dominio.

- ***Service2Operation***. Las operaciones dentro de las clases del modelo de dominio de A-OOH son creadas por medio de esta regla de transformación. La regla detecta un conjunto de elementos en el dominio origen que corresponden con un requisito de tipo *Service*, el cual deberá estar asociado a un requisito *Content*. Una vez detectado este patrón de elementos, se creará dentro de la debida clase en el dominio destino una operación del tipo UML *Operation* (Figura 1.6).

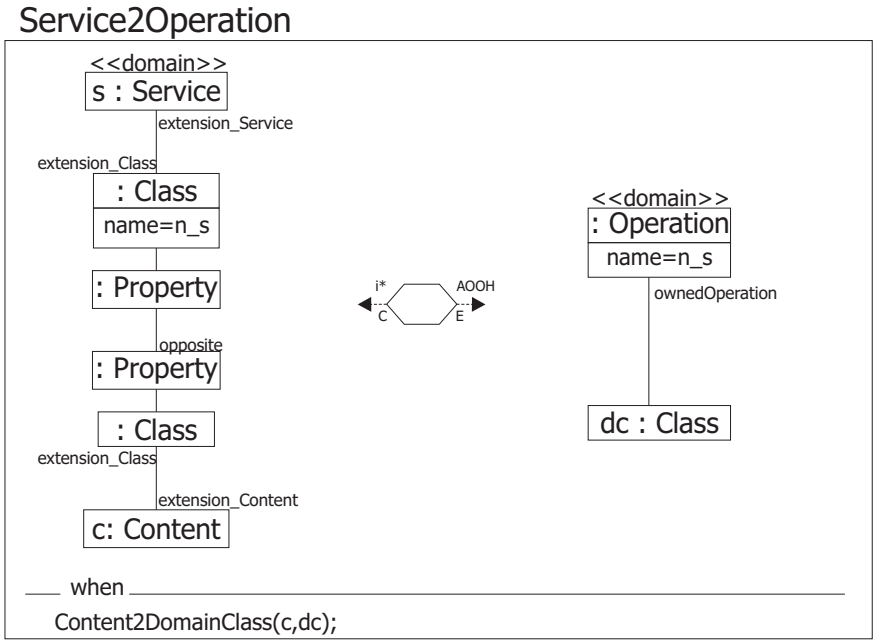


Figura 1.6. Regla QVT para obtener las operaciones de las clases del modelo de dominio.

- ***Navigation2Relationship***. Esta relación permite crear asociaciones entre las clases del modelo de dominio. Existen dos requisitos *Content* como origen, si los dos requisitos se usan para cumplir el mismo requisito de navegación, entonces se crea una asociación de tipo UML *association* entre las clases del modelo de dominio que las representan (Figura 1.7).

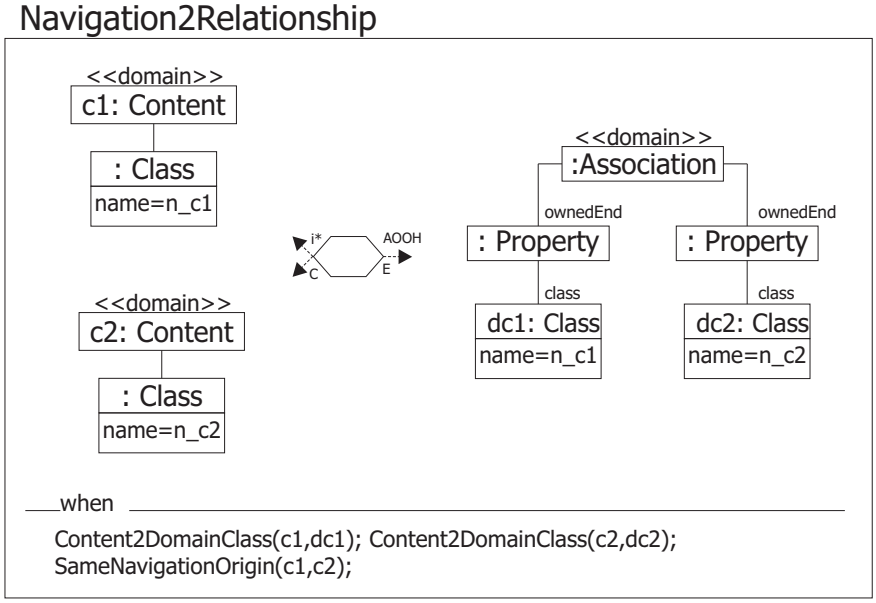


Figura 1.7. Regla QVT para obtener las relaciones entre las clases del modelo de dominio A-OOH.

Como se mencionó anteriormente, el modelo de navegación está compuesto de nodos navegacionales y sus respectivas relaciones para indicar las rutas de navegación de la aplicación Web. Para derivar el modelo de navegación los requisitos tomados en cuenta para generar un conjunto de reglas de transformación QVT son los requisitos de contenido (*Content*), servicio (*Service*), navegación (*Navigational*) y personalización (*Personalization*). En este caso, el dominio origen continúa siento el modelo de requisitos mientras que el dominio destino corresponderá al modelo de navegación de A-OOH. Las reglas QVT para la obtención del modelo de navegación son las siguientes:

- ***Navigation2NavClass***. Esta regla de transformación permite crear las clases navegacionales del modelo de navegación de A-OOH. Cuando se detecta en el dominio origen un requisito *Navigational* unido a un requisito *Content*, se creará una clase estereotipada como *NavigationalClass* en el dominio destino. Además, cada una de las nuevas clases es el destino de una nueva asociación *TransversalLink* desde una *C-Collection* previamente creada por la función *createMenu* que se encuentra en la cláusula *When* (Figura 1.8).
- ***Personalization2NavClass***. La regla de transformación es similar a la regla anterior, pero detectando los requisitos de personalización que se encuentren asociados a un requisito de contenido. Se derivan en el modelo de navegación los mismos elementos que en la regla anterior.
- ***Navigation2TransversalLink***. Esta regla de transformación permite crear asociaciones entre clases navegacionales en el modelo de navegación. Existen dos requisitos *Content* como origen, si los dos requisitos se usan para cumplir con el mismo requisito de navegación (comprobado en la cláusula *When* con la operación *SameNavigationOrigin)*, entonces se crea una asociación *TransversalLink* entre las clases del modelo de navegación que las representan.

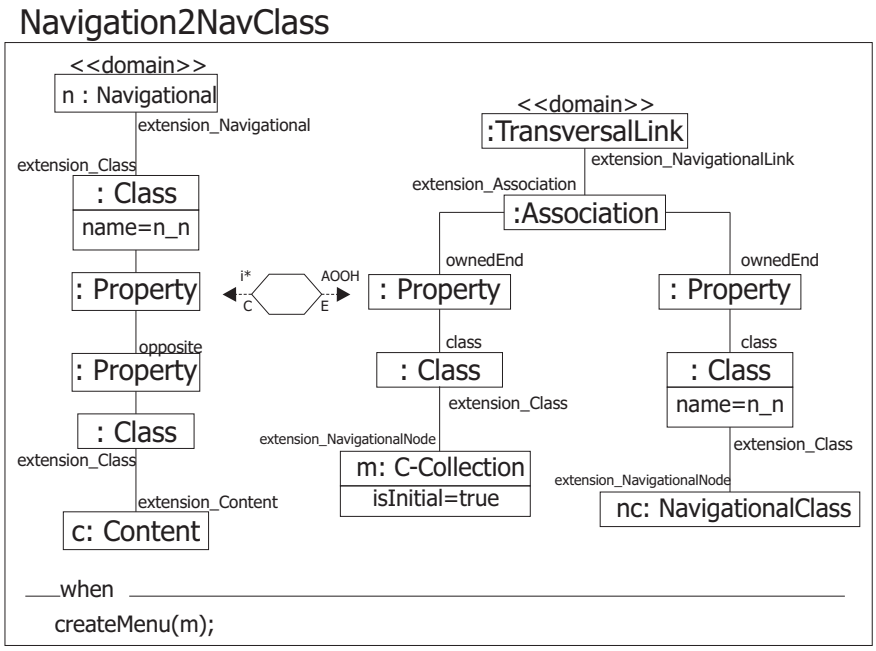


Figura 1.8. Regla QVT para obtener las clases navegacionales.

- ***Service2ServiceAndSLink***. Esta relación detecta un conjunto de elementos en el dominio origen que corresponden con un requisito del tipo *Service* asociado a un requisito *Content*. Una vez detectado este patrón de elementos, se crea en el dominio destino una clase *Operation* en la clase del modelo de navegación correspondiente. Además, se crea una asociación *ServiceLink* para cada operación añadida. El nuevo enlace de servicio se asocia a una nueva clase navegacional (*NavigationalClass*) destino. Antes de ejecutarse la transformación, se debe verificar que se cumple con las sentencias dispuestas en la cláusula *When*, en este caso *Navigation2NavClass* y *Personalization2NavClass*, con el fin de crear en el modelo de navegación todas las posibles clases de navegación a partir de cada uno de los requisitos *Content* detectados en el modelo de requisitos.

Es importante mencionar que los modelos conceptuales derivados son modelos iniciales que servirán como punto de partida para el diseñador Web, quien deberá perfeccionarlos posteriormente. Finalmente, el proceso presentado en este apartado demuestra que es posible proveer un alto grado de automatización en el proceso de desarrollo de aplicaciones Web gracias a la ingeniería Web dirigida por modelos.

1.3.2. Gestión de Requisitos en A-OOH

Las metodologías para el desarrollo de aplicaciones Web aún no responden a las exigencias actuales de este tipo de *software* [5], por lo que las necesidades y expectativas de los *stakeholders* no son captadas satisfactoriamente. De ahí que una considerable cantidad de proyectos de desarrollo no alcancen a cumplir sus objetivos y como consecuencia de esto, la aplicación Web no cumple con las expectativas reales de los usuarios [12, 25].

Las principales causas de estos problemas son la gestión insuficiente de los requisitos funcionales y no-funcionales, la comunicación ambigua e imprecisa entre los *stakeholders*, las inconsistencias no detectadas entre los requisitos, diseño y programación, así como la propagación de cambios sin analizar su impacto. En este sentido, es necesario recordar que los errores más comunes y más costosos de reparar, así como los que más tiempo consumen, se deben a una inadecuada ingeniería de requisitos [29]. Actividades propias de esta área, como la especificación

o la gestión de requisitos, son algunas de las consideradas más críticas en ingeniería Web y en la ingeniería de *software* en general.

Por otra parte, el uso de técnicas orientadas a objetivos [73] para la especificación de requisitos en ingeniería Web [12, 27] permite reflejar desde las etapas iniciales del proceso de desarrollo los objetivos, tareas y relaciones de los *stakeholders*, lo que proporciona los elementos necesarios para que se consideren las necesidades y objetivos del usuario de la aplicación Web. Sin embargo, a pesar de que los requisitos son especificados en un modelo conceptual, los *stakeholders* necesitan observar que han sido reflejados correctamente en la aplicación Web final. Una forma de brindar soporte a esta necesidad es proveer al diseñador Web con una etapa de requisitos que considere la gestión de los mismos.

Por tanto, la propuesta presentada anteriormente (ampliada en el Capítulo 3) se ha extendido para proveer al diseñador con soporte para: (i) trazabilidad de requisitos (CIM-PIM) [2] (Sección 1.1.3 y Apéndice B), (ii) evaluar el impacto derivado de un cambio en los modelos conceptuales (análisis de impacto) [3] (Capítulo 4) y (iii) seleccionar alternativas de diseño considerando la maximización y/o balance de los requisitos no-funcionales [1] (Capítulo 5).

Trazabilidad de Requisitos en A-OOH

En el campo de la ingeniería Web dirigida por modelos, realizar el seguimiento de los requisitos durante la etapa de desarrollo de la aplicación Web hasta su implementación final es una tarea compleja [3]. Esto se debe a que en ingeniería Web se deben generar varios modelos conceptuales a partir de los requisitos como son el modelo de dominio o el de navegación. Además, debido al desarrollo gradual de las necesidades de los usuarios de la aplicación Web, los modelos cambian constantemente, por lo que la trazabilidad de los requisitos se hace indispensable.

La trazabilidad de requisitos se define como la capacidad de describir y seguir la vida de un requisito, en ambas direcciones [31]: (i) determinar qué partes del modelo están relacionadas con cada uno de los requisitos, y (ii) determinar qué requisitos dieron origen a qué partes del modelo. En la actualidad, existen dos estrategias para gestionar y almacenar la información para la trazabilidad entre modelos: (i) la información se puede integrar en los modelos a los que se refiere y (ii) la información de trazabilidad se puede almacenar por separado en otro modelo [9]. La primera de estas dos opciones tiene como desventaja que si la información es almacenada en el mismo modelo, el modelo será contaminado con información poco relevante para el contexto del modelo y por lo tanto, será difícil de mantener y utilizar. Por otro lado, la segunda estrategia consiste en almacenar la información en un modelo aparte. De esta forma se pueden corregir las desventajas mencionadas.

Con el fin de brindar soporte para la trazabilidad de requisitos en A-OOH [2] se ha utilizado el concepto de modelo de *weaving*. Un modelo de *weaving* es un tipo de modelo formado por enlaces y referencias a elementos, los enlaces están dirigidos a las referencias de los elementos de un modelo origen y de un modelo destino [10]. A continuación, se presenta el metamodelo base para *weaving* [19] y una extensión para proveer a dicho metamodelo con elementos útiles para representar la trazabilidad entre modelos [9]. El metamodelo se muestra en la Figura 1.9:

- ***WElement***. Es el elemento base del cual los elementos restantes heredan, esta formado por los atributos nombre y descripción.
- ***WModel***. Representa el elemento raíz que contiene a todos los elementos del modelo, se compone de las referencias y relaciones a los modelos de entrada y salida.
- ***WLink***. Sirve para representar un enlace entre los elementos de los modelos de entrada y salida.
- ***WLinkEnd***. Este elemento representa la referencia origen o destino de un *WLink*.
- ***WElementRef***. Este elemento se asocia a una función de identificación, creando un identificador único para cada uno de elementos de los modelos de entrada y salida, por tanto *WElementRef* permite referenciar el mismo elemento de los modelos de entrada y salida por diversos elementos *WLinkEnd*.
- ***WModelRef***. Representa un identificador único de un modelo.

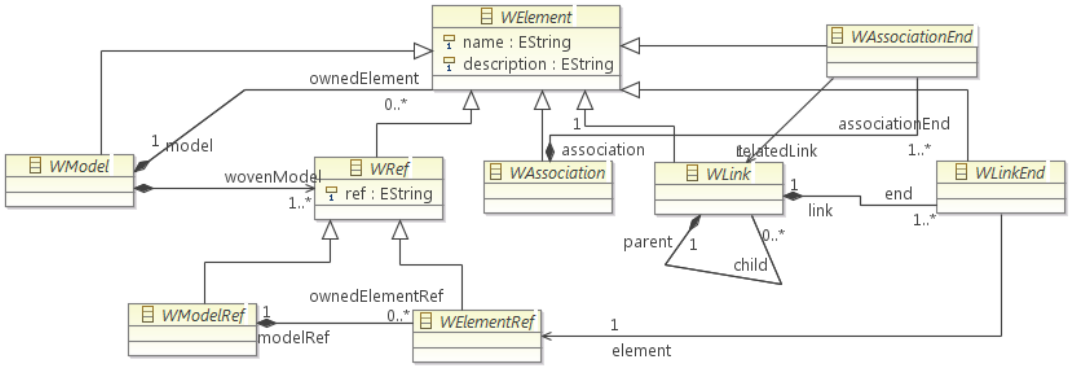


Figura 1.9. Metamodelo de *weaving*.

Por otra parte, en la Figura 1.10, se ilustra la extensión del metamodelo para *weaving* que permite la representación de la trazabilidad. La extensión la forman los siguientes elementos:

- ***TraceModel***. Es el elemento que representa al modelo de trazabilidad, está integrado por referencias a otros modelos.
- ***TraceModelRef***. Representa la referencia a otros modelos, es decir, es un único identificador para los modelos que conforman el modelo de trazabilidad.
- ***ElementRef***. Es un identificador para señalar cada elemento que integran los modelos ligados.

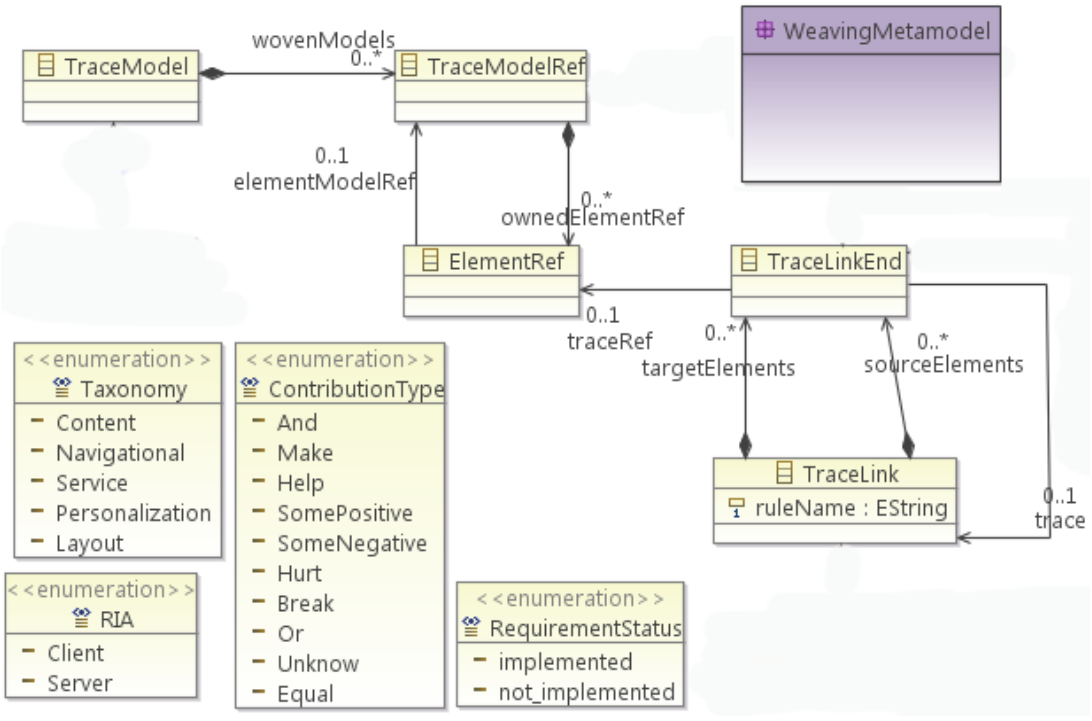


Figura 1.10. Extensión del metamodelo de *weaving* para trazabilidad.

- ***TraceLink***. Un enlace de rastreo, utilizado para representar las correspondencias entre las referencias de los elementos de los modelos enlazados. Como información de trazabilidad, almacena el nombre de la regla de transformación que ha sido ejecutada.
- ***TraceLinkEnd***. Su función es similar al elemento *WLinkEnd* del cual hereda, pues permite crear una relación uno a muchos (1-N) entre las referencias de los elementos del modelo de entrada (*sourceElements*) y los del modelo de salida (*targetElement*).

Los *trace links* han sido introducidos dentro de las reglas de transformación que generan los elementos de los modelos conceptuales de A-OOH, lo cual permite que sea generado un modelo de trazabilidad (*trace model*) al ejecutarse el conjunto de transformaciones por primera vez. Para un mejor entendimiento, se ha utilizado el lenguaje estándar QVT [58] para representar en la Figura 1.11 la incorporación de los *trace links* dentro de la regla *Content2DomainClass*. En la figura, se observa que por cada vez que se encuentre un requisito *Content* en el modelo de entrada (modelo de requisitos *i**), se creará en el modelo de dominio una clase del tipo *Class* de UML, y al mismo tiempo, para referenciar a los elementos de los modelos de entrada y salida, se creará un (*trace link*) en el modelo de trazabilidad.

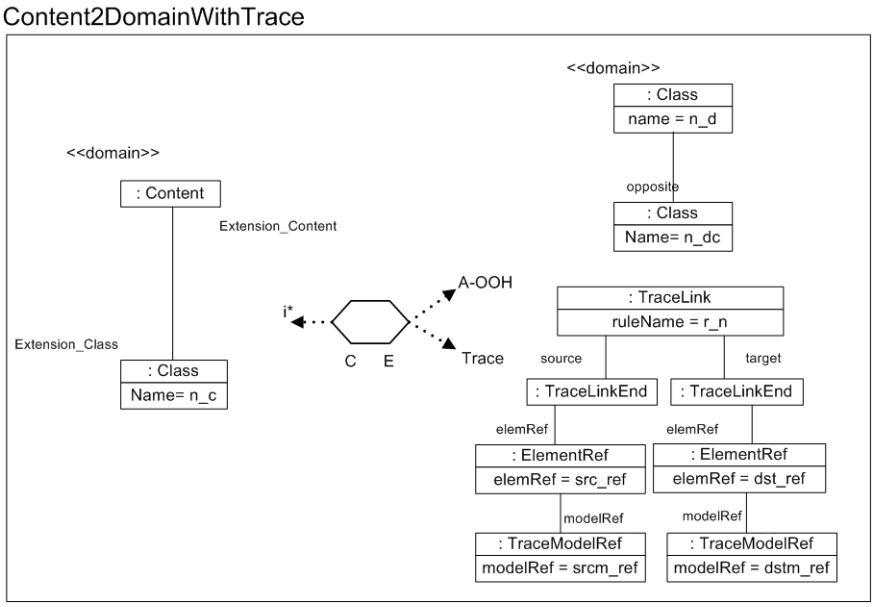


Figura 1.11. Incorporación de los *trace links* en la generación de los modelos conceptuales de A-OOH.

Análisis de Impacto en A-OOH

Los requisitos evolucionan constantemente a razón de la naturaleza dinámica de la Web. Debido a esto, es común encontrar inconsistencias entre los modelos conceptuales de la aplicación Web y los requisitos. Una de las ventajas ofrecidas por el soporte para trazabilidad en A-OOH es el de conocer las dependencias entre los elementos de los modelos conceptuales y los requisitos. Por tanto, es posible conocer los requisitos afectados debido a un cambio en alguno de los modelos conceptuales.

Análisis de impacto, conocido también como *Change Impact Analysis*, es la tarea de identificar las consecuencias potenciales de un cambio o estimar qué es necesario modificar para llevar a cabo el cambio [7], por ejemplo, una modificación en los modelos conceptuales de la aplicación Web o en la especificación de los requisitos. Comúnmente, el análisis de impacto se ha realizado de forma intuitiva por los diseñadores de la aplicación Web por medio de un

análisis superficial del código y documentación de la aplicación [29]. Esto quizá sea suficiente para aplicaciones Web no muy grandes, pero no es suficiente para aplicaciones Web sofisticadas. Asimismo, trabajos de investigación empírica como el presentado en [44] y más recientemente en [29], demuestran que incluso los desarrolladores más experimentados en la industria, deducen un análisis de impacto incompleto.

Para paliar esta limitante, se ha definido un algoritmo para analizar las dependencias entre los requisitos funcionales. El algoritmo analiza el modelo de requisitos para conocer las dependencias entre los requisitos funcionales además de saber qué requisitos no-funcionales se ven afectados (ver Capítulo 4). De esta forma, a través del soporte para trazabilidad es posible conocer qué otros elementos de los modelos conceptuales son afectados debido a un cambio en la aplicación Web. Finalmente, debido a que el modelo de requisitos es orientado a objetivos [4, 27], el algoritmo puede mostrar al diseñador un camino alternativo en el cual se indique qué requisitos funcionales tienen que ser implementados para seguir cumpliento con el objetivo (*Goal*). A continuación se presenta un ejemplo de la aplicación del algoritmo.

Supongamos que el cliente ha solicitado la implementación de una aplicación Web que permita la aplicación de encuestas *on-line*. En primer lugar es necesario que el diseñador especifique los requisitos de la aplicación Web, para esta demostración, solo se ha definido el actor que representa a la aplicación Web (Figura 1.12).

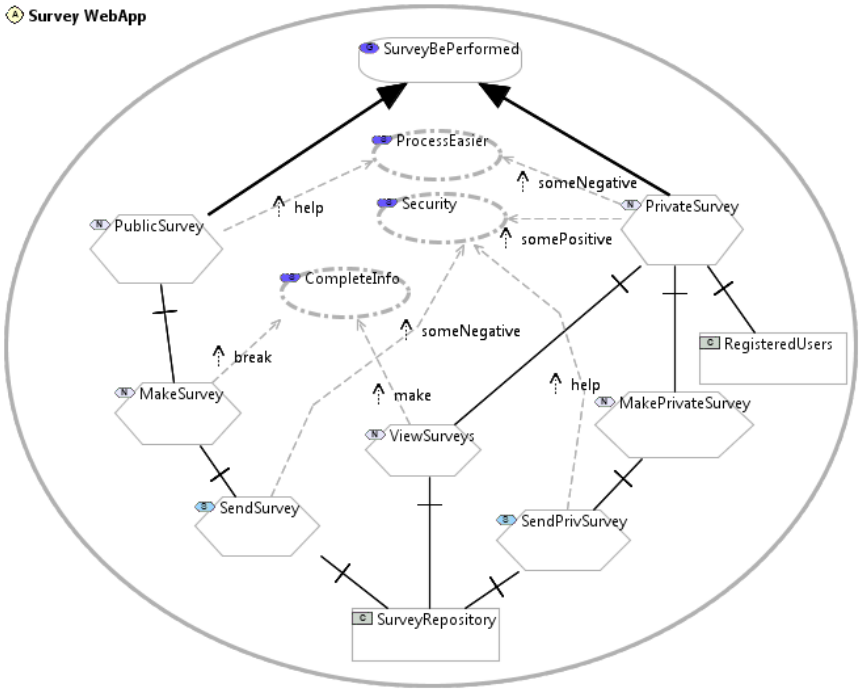


Figura 1.12. El modelo de requisitos de la aplicación *"Survey WebApp"*.

El actor "*Survey WebApp*" tiene que cumplir con el objetivo "*Survey be performed*", para lograrlo dispone de dos vías, por medio de la aplicación de la encuesta pública "*Public survey*" o la privada "*Private survey*", ambos requisitos de navegación. Cada una de las vías necesita de uno o más requisitos para cumplirse, tal es el caso del requisito "*Private survey*" debido a que necesita de los requisitos de navegación "*Make private survey*" y "*View survey*" así como del requisito de contenido "*Registered users*".

En este sentido, algunos de los requisitos afectan positiva o negativamente a las *Softgoals*, por ejemplo, el requisito navegacional "*Private survey*" afecta positivame a la *Softgoal* "*Security*" (encargada de la seguridad de la aplicación Web), pero también afecta de forma negativa

a la *Softgoal* "*Process Easier*", la cual representa el nivel de simplicidad con que deberá realizarse el proceso de revisión utilizando la aplicación Web (usabilidad), por último, el requisito navegacional "*View Surveys*" afecta de forma positiva a la *Softgoal* "*Complete Info*", es decir que si el requisito navegacional es implementado, el usuario de la aplicación Web podrá obtener información más detallada acerca del artículo asignado para su revisión.

El tipo de contribuciones que realizan los requisitos funcionales a las *Softgoals* es relevante para la ejecución del algoritmo a razón de que son utilizadas para decidir qué requisitos funcionales será necesario implementar. En el ejemplo, sólo los requisitos funcionales relacionados con el requisito de navegación "*Public survey*" se encuentran implementados en los modelos conceptuales (Figura 1.13).

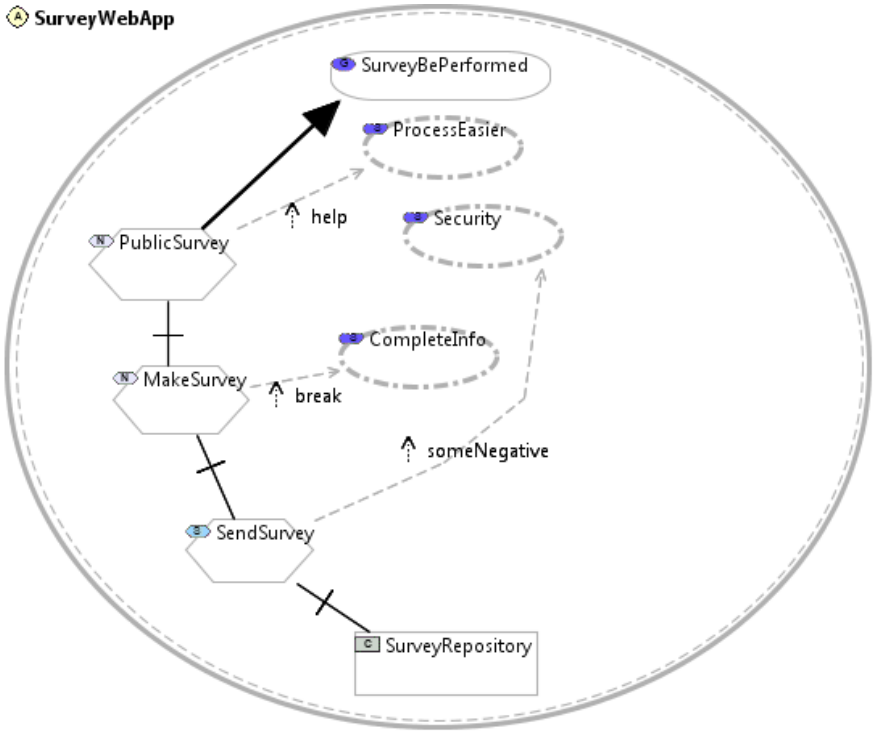


Figura 1.13. Requisitos de la aplicación *"Survey WebApp"* implementados inicialmente.

Ahora bien, supongamos el siguiente escenario: el administrador de la base de datos presenta una solicitud de cambio en la estructura de la base de datos de la aplicación Web, por tal motivo, el diseñador Web necesita eliminar un elemento del modelo de dominio A-OOH, gracias al soporte para trazabilidad, es posible conocer el requisito afectado por la clase del modelo de dominio que ha sido eliminada, como puede verse en la Figura 1.13, la clase corresponde con el requisito navegacional "*Public survey*". Por tanto, el objetivo "*Survey be performed*" no se puede cumplir. Por tal motivo, es necesario conocer:

- ¿Qué requisitos son afectados por este cambio en el modelo de requisitos?
- ¿Qué elementos de los modelos conceptuales son afectados?

Por último, es necesario encontrar un camino alternativo en el modelo de requisitos (si es que existe) para continuar cumpliendo con el objetivo "*Survey be performed*".

El algoritmo inicia una vez que se han cumplido un conjunto de pre-condiciones, las cuales pueden consultarse con detalle en el Capítulo 4. El primer paso del algoritmo consiste en realizar un listado de todos aquellos requisitos funcionales que se encuentren implementados o no (en este caso, reflejados en el modelo de dominio), y que además realicen alguna contribución

positiva o negativa a cualquier *Softgoal*, el listado quedará como lo muestra la Tabla 1.2. El requisito afectado se muestra resaltado en negrita. Con el listado generado y el soporte para trazabilidad de A-OOH, es posible saber qué elementos del modelo conceptual de dominio resultan afectados por la eliminación del requisito navegacional "*Public survey*".

Tabla 1.2. La contribución de los requisitos funcionales a cada requisito no-funcional.

Requirements	***"Process easier"***	***"Complete info"***	***"Security"***
"Public Survey"	**Help**	**-**	**Hurt**
"Make Survey"	-	Break	-
"Private Survey"	Some-	-	Some+
"Send Survey"	-	-	Some-
"View Surveys"	-	Make	-
"Send Private Survey"	-	-	Help

El siguiente paso consiste en determinar un camino alternativo para la satisfacción del objetivo de la aplicación ("*Survey be performed*"). Por cada *Softgoal* (requisito no-funcional) que recibe una contribución por parte del requisito afectado (Tabla 1.2) es necesario buscar un requisito funcional no implementado del cual su contribución compense la eliminación del requisito a remover. En este caso, el requisito navegacional *"Private Survey"* realiza dos contribuciones a las *Softgoals "Process easier"* y *"Security"*, por lo tanto se puede implementar. Para poder determinar si el requisito se puede implementar o no, es necesario aplicar un conjunto de heurísticas (definidas en detalle en el Capítulo 4). El requisito navegacional *"Private Survey"* se puede implementar gracias a las heurísticas número 2 y 3. Este paso es iterativo y finaliza cuando no hay más requisitos (no implementados) que realicen contribuciones a las *Softgoals*. El paso finaliza cuando es detectado el requisito navegacional *"Send Private Survey"*.

Finalmente, es necesario aplicar una post-condición, la cual establece que si los requisitos a implementar (*"Private Survey"* y *"Send Private Survey"*) tienen uno o más requisitos funcionales asociados, deben ser implementadas de forma automática. Por tanto, los requisitos *"View Surveys"* y *"Registered Users"* deben de ser implementados. En la Figura 1.14 se muestran los requisitos funcionales a implementar.

Optimización de Requisitos No-Funcionales en Aplicaciones Web

Como se ha motivado anteriormente, las aplicaciones Web tienen una audiencia amplia y heterogénea [2, 26], debido a esto, el diseñador se enfrenta a una problemática muy particular: ¿cómo diseñar la aplicación optimizando el máximo número posible de requisitos no-funcionales, de tal forma que la aplicación Web sea capaz de satisfacer, en medida de lo posible, a la amplia y heterogénea audiencia?, una opción consiste en enriquecer a las metodologías de diseño para que sean capaces de asistir al diseñador en la etapa de requisitos, es decir, que permitan seleccionar distintas opciones de diseño en base a la audiencia de la aplicación Web.

Por lo tanto, en A-OOH se ha implementado el algoritmo Optimización de Pareto [63] para proveer al diseñador con un conjunto de configuraciones de diseño basadas en implementación de los requisitos funcionales considerando la prioridad de los requisitos no-funcionales. Estas configuraciones de diseño se basan en el estado de los requisitos funcionales, es decir si los requisitos son implementados o no son implementados. La Optimización de Pareto, llamada así en honor de su introductor, Vilfredo Pareto, es un concepto de la economía con aplicación tanto en esa disciplina como en ciencias sociales e ingeniería [42]. El concepto está relacionado con estudios de eficiencia económica y distribución del ingreso y establece como eficiente aquella situación en la cual se cumple que no es posible beneficiar a más individuos en un sistema sin perjudicar a otros.

La Optimización de Pareto ha sido ampliamente aplicada en la ingeniería de *software* [72], principalmente, en problemas en los que hay dos o más objetivos, este tipo de problema es conocido como problema multi-objetivo (PMO) [56]. En un problema de optimización, se trata de encontrar una solución que represente el valor óptimo para una función objetivo [28].

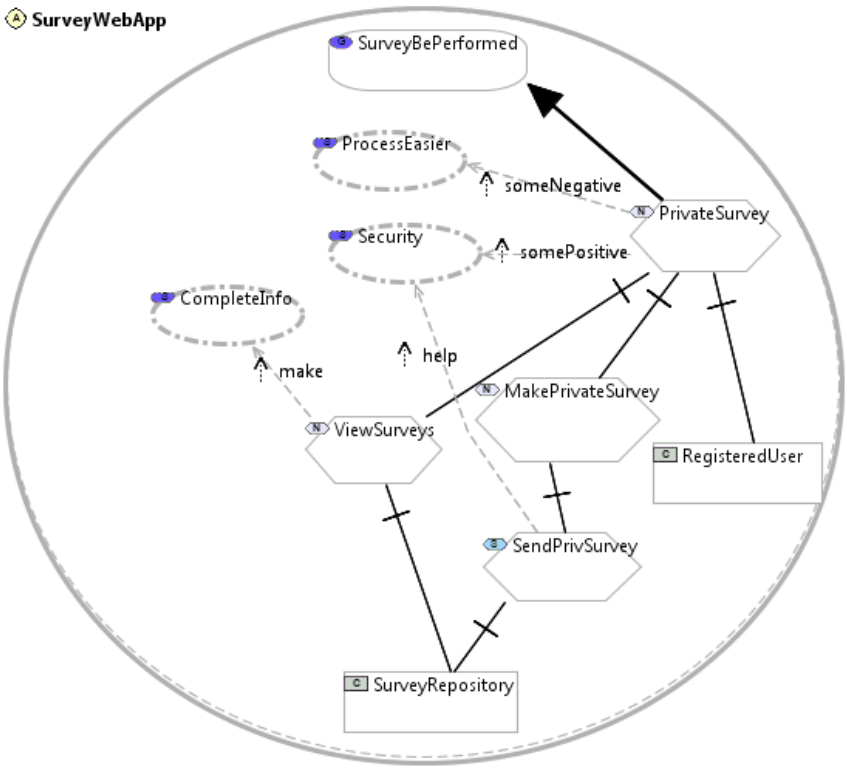


Figura 1.14. Los nuevos requisitos a implementar en el modelo de dominio.

Una característica de los PMO es que, como regla general, tienen un conjunto de soluciones, que inclusive puede ser infinito, y entonces los algoritmos deben describir lo mejor posible este conjunto. Por lo tanto, en este trabajo, se han realizado algunas modificaciones que permiten obtener mejores representaciones de los conjuntos solución. Información más detallada se encuentra en el Capítulo 5 de la tesis doctoral.

Para un mejor entendimiento de la propuesta presentada en este apartado, se han adaptado una serie de definiciones para la aplicación del algoritmo Óptimo de Pareto, estas son presentadas a continuación:

Definición. 1 ***Optimización de Pareto:*** *dado un conjunto de configuraciones y un conjunto de requisitos, la configuración "A" es una optimización sobre la configuración "B" si solo si "A" puede, al menos, optimizar a un requisito no-funcional mejor que "B", sin deteriorar a otro"* [17, 63].

Definición. 2 ***Configuración Óptima de Pareto:*** *es aquella configuración que mejor satisfaga a un requisito no-funcional mientras satisface a los demás de igual o mejor forma.*

Definición. 3 ***Configuración:*** *una configuración está formada por un conjunto de requisitos funcionales que pueden ser implementados en los modelos conceptuales de la aplicación Web.*

Definición. 4 ***Frontera de Pareto:*** *es el espacio de solución constituido por el conjunto de soluciones óptimas de Pareto, es decir, las soluciones que no son dominadas por ninguna otra solución.*

Definición. 5 ***Dominio entre Soluciones:*** *se dice que una solución no es dominada por otra solución cuando, en el espacio de solución, no existe ninguna otra que mejor satisfaga a un requisito no-funcional sin deteriorar a otro.*

El conjunto de soluciones óptimas de Pareto puede ser utilizado por el diseñador para la toma de decisiones, por ejemplo, cuando necesite seleccionar la configuración que mejor balancee la compensación entre los requisitos no-funcionales y cuando necesite considerar la maximización sobre un requisito no-funcional en particular.

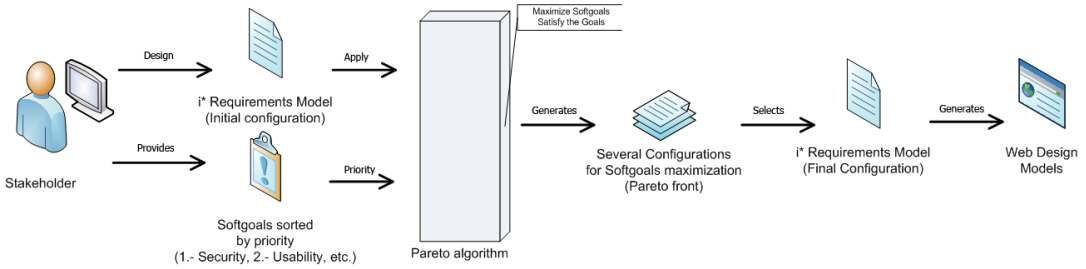


Figura 1.15. Propuesta para la optimización de requisitos no-funcioales en aplicaciones Web.

La Figura 1.15 muestra los pasos definidos en la propuesta descrita en este apartado para la optimización de requisitos no-funcionales. En el primer paso, el *stakeholder*, concretamente en el rol de diseñador, especificará los requisitos de la aplicación Web utilizando el marco de modelado orientado a objetivos *i**, así mismo, establecerá una lista de requisitos no-funcionales a priorizar. El segundo paso consiste en aplicar el algoritmo Optimización de Pareto, como resultado del algoritmo se obtendrá un conjunto de configuraciones. Finalmente, el último paso consiste en la selección de la configuración que mejor satisfaga la lista de requisitos no-funcionales a priorizar. Es importante destacar que los requisitos no-funcionales son considerados *Softgoals* en nuestra propuesta para la especificación de requisitos, como se definió en la Sección 1.3.1, por lo que en el ejemplo se utilizará la palabra *Softgoal* para referirse a los requisitos no-funcionales.

A continuación, se presenta un ejemplo conciso para ejemplificar paso a paso la aplicación de la propuesta para la optimización de *Softgoals* en aplicaciones Web. El caso de estudio es acerca de una aplicación Web para la gestión de conferencias (*Conference Management System*)[6]. El propósito de la aplicación Web es brindar soporte al proceso de envío, evaluación y selección de los artículos para una conferencia. La Figura 1.16, muestra la especificación de requisitos de la aplicación Web (modelo de requisitos *i**). El objetivo de la aplicación Web consiste en "*Process of review of papers be selected*", para satisfacer el objetivo, es necesario la implementación de alguno de los requisitos navegacionales "*Blind review process*" y "*Normal review process*". En este ejemplo el objetivo es logrado a través del requisito navegacional "*Blind review process*". Una explicación detallada del ejemplo se puede consultar en el Capítulo 5 de la tesis doctoral.

En la especificación de requisitos, también se puede observar que algunos requisitos necesitan de otros para poder cumplir con su función, tal es el caso del requisito navegacional "*Review paper*" el cual necesita del requisito de servicio "*Submit review*". Además, algunos requisitos afectan de forma positiva o negativa a algunas *Softgoals*, por ejemplo, el requisito de servicio "*Download paper without author's name*" afecta de forma positiva a la *Softgoal* "*Privacy be maximized*" y de forma negativa a "*Obtain more complete info*". Las contribuciones de los requisitos funcionales a las *Softgoals* pueden verse en la Tabla 1.3.

Una vez especificados los requisitos de la aplicación Web, es necesario elaborar una lista que contenga a los requisitos (implementados o no) que realicen algún tipo de contribución a alguna *Softgoal*. La Tabla 1.4, muestra una lista de requisitos y el tipo de contribución (Tabla 1.3) hacia las *Softgoals*, en donde: S1 se refiere a la *Softgoal "Be fair in review"*, la *Softgoal* representa en el modelo de requisitos que el proceso de revisión en la aplicación para la gestión de conferencias debe de ser imparcial, la *Softgoal* S2 corresponde a *"Review process easier"* y se refiere a que el proceso de revisión debe de ser lo más simple posible de realizar gracias

[6] La especificación completa del caso de estudio se encuentra en `http://users.dsic.upv.es/~west/iwwost01`

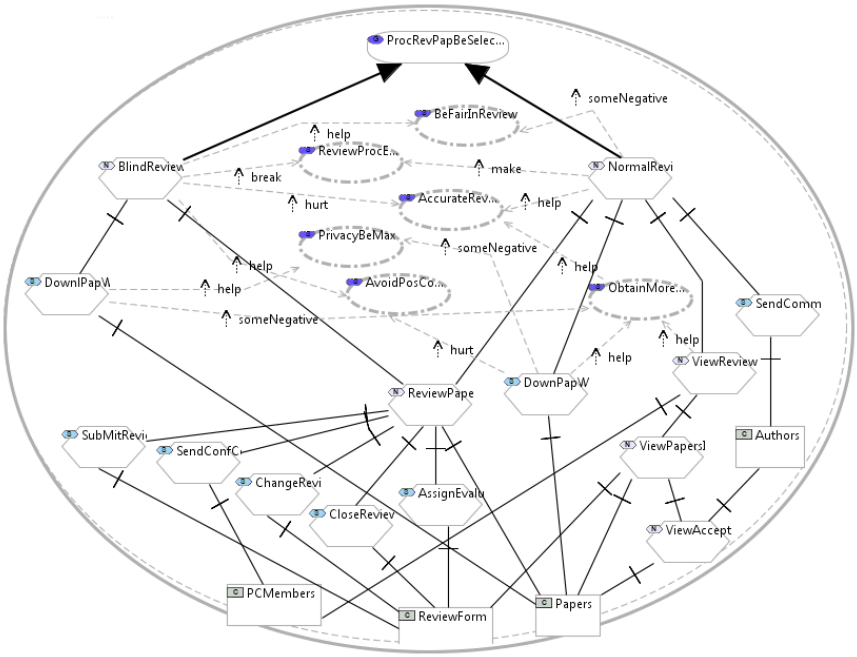


Figura 1.16. Especificación de requisitos de la aplicación Web para la gestión de conferencias.

Tabla 1.3. Tipos de contribuciones.

Contribución	**Tipo de Contribución**
Positiva	Help
Positiva de fuerza desconocida	Some +
Negativa	Hurt
Negativa de fuerza desconocida	Some -
Negativa suficiente para no satisfacer una *Softgoal*	Break
Positiva suficiente para satisfacer una *Softgoal*	Make
Una contribución a una *Softgoal* cuya polaridad es desconocida	Unknown

a que se llevará a cabo por medio de la aplicación Web, la *Softgoal* S3 llamada *"Accurate review process"* es utilizada para indicar que el proceso de revisión debe de ser lo más preciso posible, S4 se refiere a la *Softgoal "Privacy be maximized"* para indicar que en la aplicación Web la seguridad debe de ser maximizada cuando se realicen ciertos requisitos funcionales, la *Softgoal* S5 nombrada *"Avoid possible conflicts of interest"*, permite representar la ética del usuario de la aplicación Web debido a que él deberá ser capaz de evitar conflictos de interés al momento de realizar la revisión del artículo asignado y la *Softgoal* S6, *"Obtain more complete info"* representa que al realizarse un proceso de revisión normal, el usuario de la aplicación obtendrá información más detallada sobre los autores del artículo que le ha sido asignado.

Tabla 1.4. Las contribuciones de los requisitos hacia las *Softgoals*.

Requisitos	***"S1"***	***"S2"***	***"S3"***	***"S4"***	***"S5"***	***"S6"***
"Blind review process"	Help	Break	Hurt	Help	-	-
"Download papers without authors' name"	**-**	**-**	**-**	**-**	**Help**	**Some -**
"Normal review process"	Some -	Make	Help	-	-	-
"Download paper with author's name"	-	-	-	Hurt	Some -	Help
"View review process status"	-	-	-	-	-	Help

A continuación, es necesario almacenar cada configuración posible de los requisitos listados en la Tabla 1.4, en donde: la variable "I" representa el estado "implementado" y la variable "N" el estado "no implementado". Por lo tanto, tenemos 32 posibles configuraciones (Tabla 1.8, columnas 1 y 2).

Además, es necesario asignar un peso a cada tipo de contribución, para esto, deben de asignarse por cada tipo el peso (W): w= +1 si el tipo de contribución es "*Help*", w= -1 si es "*Hurt*", w= +2 para el tipo "*Some +*" , w= -2 para "*Some -*", w= +4 si es del tipo "*Make*" y w= -4 para el tipo "*Break*".

Es importante mencionar que el valor de estos pesos depende de los fundamentos del marco de modelado *i**, por ejemplo, teniendo en cuenta que la contribución del tipo "*Make*" es el valor positivo más fuerte que sirve para al satisfacción de una *Goal* o *Softgoal*, se decidió asignarle el valor numérico 4. Esta forma de proceder continúa en proceso de estudio con la idea de tener una distribución de los valores numéricos lo más confiable posible, por esta razón se están realizando una serie de experimentos actualmente.

$$M = \begin{pmatrix} +1 & -4 & -1 & 0 & +1 & 0 \\ 0 & 0 & 0 & +1 & 0 & -2 \\ -2 & +4 & +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1 & +1 \\ 0 & 0 & 0 & 0 & 0 & +1 \end{pmatrix} \tag{1.1}$$

Tabla 1.5. Lista de *Softgoals*.

Prioridad	Softgoal
1	*"S4.- Privacy be maximized"*
2	*"S2.- Review process easier"*
3	*"S3.- Accurate review process"*
4	*"S1.- Be fair in review"*
5	*"S5.- Avoid possible conflicts os interest"*
6	*"S6.- Obtain more complete info"*

Con los pesos establecidos, el siguiente paso consiste en crear una matriz para calcular los valores númericos asociados cada configuración (Tabla 1.8, columnas 7 a 12), por ejemplo para la configuración X25, si el requisito R1 *"Blind review process"* no es implementado, tenemos que la *Softgoal "Be fair in review"* será afectada (-2). Por último, se calcula si la configuración es Frontera de Pareto (4) (Tabla 1.8, columnas 13), para hacerlo, se deben comparar cada una de las configuraciones indicando si es que existe una que sea mejor que otra (5), por ejemplo, la configuración X3 está en la Frontera de Pareto por que es mejor que las configuraciones X1 y X2 debido a que maximiza al menos a una *Softgoal*, es decir, maximiza a la *Softgoal* 5 *"Avoid possible conflicts of interest"*.

Finalmente, el último paso consiste, aparte de maximizar las *Softgoals*, continuar cumpliendo con el objetivo establecido en el modelo de requisitos (*"Process of review papers be selected"*). Para esto, es necesario crear una lista de *Softgoals* ordenadas por prioridad (Tabla 1.5). En base a la lista, se debe de seleccionar de la Tabla 1.8 las configuraciones que además de estar en la Frontera de Pareto cumplen con el objetivo, y de esta forma escoger la configuración que maximice las *Softgoals*. Para este ejemplo, "X3", "X7" y "X17" cumplen con el objetivo, de las tres configuraciones, la configuración X3 es la mejor opción, porque de acuerdo con la lista (Tabla 1.5), "S4" y "S2" son las *Softgoals* a priorizar, por tanto, la configuración "X17" maximiza "S2", sin embargo afecta negativamente a la *Softgoal* 4 (Tabla 1.8). En cambio, la configuración "X3" maximiza a la *Softgoal* "S4" (1) y no afecta a la *Softgoal* "S2" (0).

Tabla 1.6. The posible requirements to implement or not for the softgoal tradeoff.

Configuración	R1	R2	R3	R4	R5	F(S1)	F(S2)	F(S3)	F(S4)	F(S5)	F(S6)	Pareto front
X1	I	I	I	I	I	-1	0	0	-1	0	0	No
X2	I	I	I	I	N	-1	0	0	-1	0	-1	No
X3	I	I	I	N	I	-1	0	0	1	1	-1	Yes
X4	I	I	I	N	N	-1	0	0	1	1	-2	No
X5	I	I	N	I	I	1	-4	-1	-1	0	0	Yes
X6	I	I	N	I	N	1	-4	-1	-1	0	-1	No
X7	I	I	N	N	I	1	-4	-1	1	1	-1	Yes
X8	I	I	N	N	N	1	-4	-1	1	1	-2	No
X9	I	N	I	I	I	-1	0	0	-2	0	2	Yes
X10	I	N	I	I	N	-1	0	0	-2	0	1	No
X11	I	N	I	N	I	-1	0	0	0	1	1	Yes
X12	I	N	I	N	N	-1	0	0	0	1	0	No
X13	I	N	N	I	I	1	-4	-1	-2	0	2	Yes
X14	I	N	N	I	N	1	-4	-1	-2	0	1	No
X15	I	N	N	N	I	1	-4	-1	0	1	1	Yes
X16	I	N	N	N	N	1	-4	-1	0	1	0	No
X17	N	I	I	I	I	-2	4	1	-1	-1	0	Yes
X18	N	I	I	I	N	-2	4	1	-1	-1	-1	No
X19	N	I	I	N	I	-2	4	1	1	0	-1	Yes
X20	N	I	I	N	N	-2	4	1	1	0	-2	No
X21	N	I	N	I	I	0	0	0	-1	-1	0	No
X22	N	I	N	I	N	0	0	0	-1	-1	-1	No
X23	N	I	N	N	I	0	0	0	1	0	-1	Yes
X24	N	I	N	N	N	0	0	0	1	0	-2	No
X25	N	N	I	I	I	-2	4	1	-2	-1	2	Yes
X26	N	N	I	I	N	-2	4	1	-2	-1	1	No
X27	N	N	I	N	I	-2	4	1	0	0	1	Yes
X28	N	N	I	N	N	-2	4	1	0	0	0	No
X29	N	N	N	I	I	0	0	0	-2	-1	2	Yes
X30	N	N	N	I	N	0	0	0	-2	-1	1	No
X31	N	N	N	N	I	0	0	0	0	0	1	Yes
X32	N	N	N	N	N	0	0	0	0	0	0	No

1.4. Hacia la Gestión de Requisitos en las *Rich Internet Applications*

En la actualidad, las aplicaciones Web evolucionan constantemente, por ejemplo, se evolucionó de las páginas HTML[7] estáticas al siguiente nivel en donde la mayor parte del contenido era generado dinámicamente a través de diversos sistemas y bases de datos. Conforme a esta evolución, se crearon las primeras metodologías para el desarrollo de aplicaciones Web, como OOHDM [60]. Al continuar la evolución natural de las aplicaciones Web, se tomaron en cuenta factores importantes como la estética, el contenido, la funcionalidad y la personalización en la creación de nuevas metodologías de desarrollo (WebML [16], NDT [24], A-OOH [26], UWE [37], OOWS [52]).

Recientemente, las aplicaciones Web han experimentado cambios significativos relacionados con la tecnología de implementación entre los cuales encontramos la distribución de la lógica de negocio entre el cliente y el servidor, la comunicación que se establece entre ambos así como las interfaces de usuario (UIs). Este tipo de aplicaciones Web recibe el nombre de RIAs (*Rich Internet Applications*) y se caracterizan por ofrecer al usuario una serie de características muy similares a las de las aplicaciones de escritorio, por ejemplo, la inclusión de contenido multimedia bajo demanda, la actualización de contenido dinámicamente, es decir, sin que la página Web vuelva a ser generada por completo tras una petición del usuario y la posibilidad

[7] *Hyper Text Markup Language*

de comunicación por medio del audio y video. La Figura 1.17[8] muestra un diagrama de Venn en donde se representa la combinación de tecnologías que han dado origen a las RIAs, por tanto se puede deducir fácilmente que las RIAs son una combinación de la tecnología y/o funcionalidad de las aplicaciones de escritorio, de las aplicaciones Web y de las tecnologías de comunicación (audio, video, Internet).

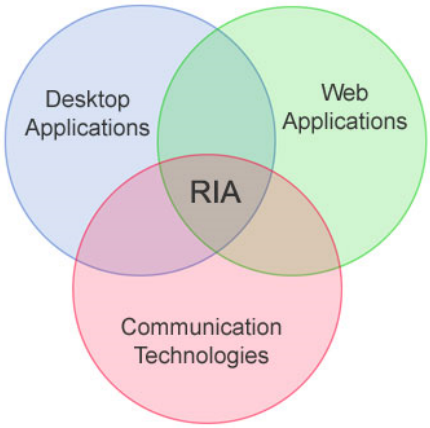


Figura 1.17. Diagrama de Venn que ejemplifica las características de las RIAs.

Sin embargo, la evolución de las aplicaciones Web a RIAs implica una mayor complejidad en su desarrollo debido al nivel de funcionalidad e interactividad que ofrecen al usuario. Por lo tanto, son más difíciles de diseñar e implementar que las aplicaciones Web 1.0. Por esta razón, las metodologías para el desarrollo de aplicaciones Web han sido objeto de mejoras (adaptaciones en algunos casos) y han surgido nuevas aproximaciones específicamente para el desarrollo de RIAs [55]. Dentro del grupo de aproximaciones que han sido extendidas para el soporte de RIAs encontramos a OOHDM [65], OOH con la extención OOH4RIA [46], UWE con las extensiones UWE-Patterns [40], UWE-R [45] y UWE-RUX [43], WebML [13, 14] y OOWS [67]. Por otra parte, dentro del grupo de aproximaciones creadas exclusivamente para el desarrollo de RIAS encontramos a RUX-Model [54], *Internet Aplication Modeling Language* [69] y ADRIA [20], para información más detallada sobre las aproximaciones tradicionales y las de nueva creación el lector puede consultar el Apéndice C de la tesis doctoral. La idea principal de las aproximaciones es brindar soporte a las características particulares de las RIAs, pero la mayoría de las metodologías se han enfocado principalmente en cuestiones de presentación [43, 53] y en capacidades de interacción [46] descuidando la etapa de análisis y especificación de requisitos. Esto es una desventaja muy importante a considerar, debido a que el diseñador tiene que lidiar con los requisitos de la Web 1.0, como lo es la navegación más los requisitos específicos de las RIAs, tales como, la capacidad de respuesta o la reducción del ancho de banda [70].

Además, el diseñador debe de considerar la distribución de los requisitos funcionales, es decir, si son implementados en el cliente o en el servidor y cómo la distribución afecta el cumplimiento de las *Softgoals*. Por ejemplo, si se analiza la distribución de los requisitos funcionales entre el cliente y el servidor, se podrán tomar decisiones de diseño que favorezcan la reducción de la cantidad y frecuencia del tráfico entre el cliente y el servidor, lo que sin duda alguna resulta en un beneficio para la seguridad de la RIA. Por lo tanto, elegir dónde serán implementados los requisitos funcionales (cliente o servidor) es una decisión fundamental (más no trivial) para el correcto desempeño de la RIA.

En esta sección, se presenta una adaptación del algoritmo Óptimo de Pareto, presentado anteriormente, para asistir al diseñador al momento de decidir sobre la distribución de los

[8] Imagen tomada de `http://www.simonwhatley.co.uk/rich-internet-applications-a-background`

requisitos funcionales de la aplicación Web entre el cliente y el servidor. Para esto, se han definido una serie de pasos los cuales pueden verse en la Figura 1.18. El primer paso consiste en la especificación de los requisitos por parte del diseñador y existen dos formas para llevarla a cabo, en la primera, el diseñador puede crear un modelo de requisitos RIA a partir de un modelo de requisitos para aplicaciones Web 1.0, la forma de hacerlo es especificar en el modelo de requisitos los requisitos particulares de las RIAs; la segunda forma, consiste en crear un modelo de requisitos desde cero, es decir, exclusivamente con los requisitos de las RIAs. El segundo paso es aplicar el algoritmo de Óptimo de Pareto para obtener el conjunto de configuraciones Cliente/Servidor. Por último, en el tercer paso el diseñador solo tendrá que seleccionar el modelo de requisitos final con el cuál podrá saber qué requisitos funcionales implementar en el servidor o en el cliente considerando la optimización de las *Softgoals*. Para obtener información más detallada así como las definiciones y conceptos de esta sección consultar el Apéndice C.

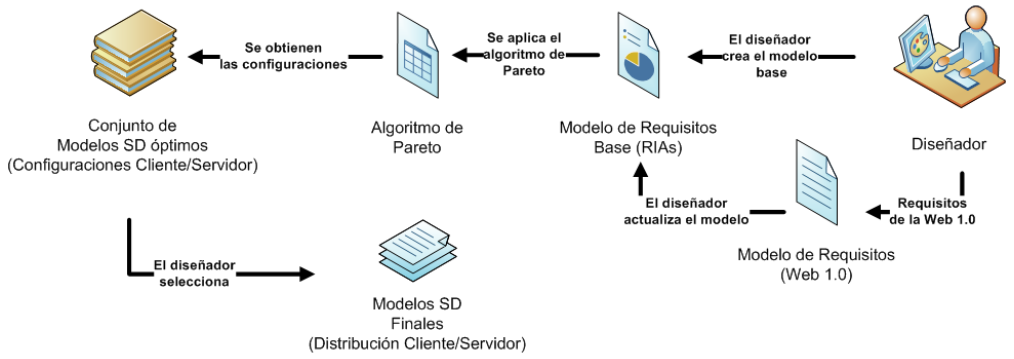


Figura 1.18. Visión general de la propuesta para el análisis de requisitos RIA.

A continuación, se presenta un caso de estudio para ejemplificar la propuesta presentada en este apartado de la tesis doctoral. El caso de estudio es acerca de una compañía de bioinformática, la cual tiene como objetivo ofrecer servicios en línea sobre el análisis del genóma humano. La aplicación Web permitirá al usuario subir la información sobre un gen, analizar la información y obtener reportes personalizados. El primer y único servicio disponible para el usuario será el reporte de enfermedades al que es vunerable el gen, para lograrlo, se comparará la información proporcionada por el usuario con la base de datos de genes de la compañía. La compañía, desea ofrecer una Web basada en RIA con la finalidad de ofrecer una aplicación Web lo suficientemente atractiva e interactiva para los usuarios. Para esto, el equipo de desarrollo de la compañía quiere estudiar y analizar los beneficios que se obtendrán de la distribución de los requisitos funcionales entre el servidor y el cliente, así como el impacto que tendrán en las *Softgoals* para poder considerar asuntos relacionados a la seguridad de la información proporcionada por el usuario así como estudiar la compatibilidad cuando sea mostrada la información a los usuarions de la RIA. Para efectos demostrativos, en el caso de estudio nos enfocaremos en los requisitos necesarios para el reporte de enfermedades.

El primer paso consiste en la especificación del modelo de requisitos base para RIAs por parte del diseñador. En él, el diseñador deberá especificar los objetivos, los requisitos funcionales y las *Softgoals* necesarias para modelar el reporte de enfermedades (Figura 1.19).

Debido a que aún no sabemos qué requisitos serán ubicados de lado del cliente o de lado del servidor, las contribuciones por parte de los requisitos a las *Softgoals* son etiquetadas como "*unknown*" (Tabla 1.3). En el particular caso de la *Softgoal* "Compatibilidad", las contribuciones recibidas por parte de los requisitos quedan establecidas en este paso debido a que no dependen del lugar donde se implementará el requisito (cliente o servidor). Las contribuciones

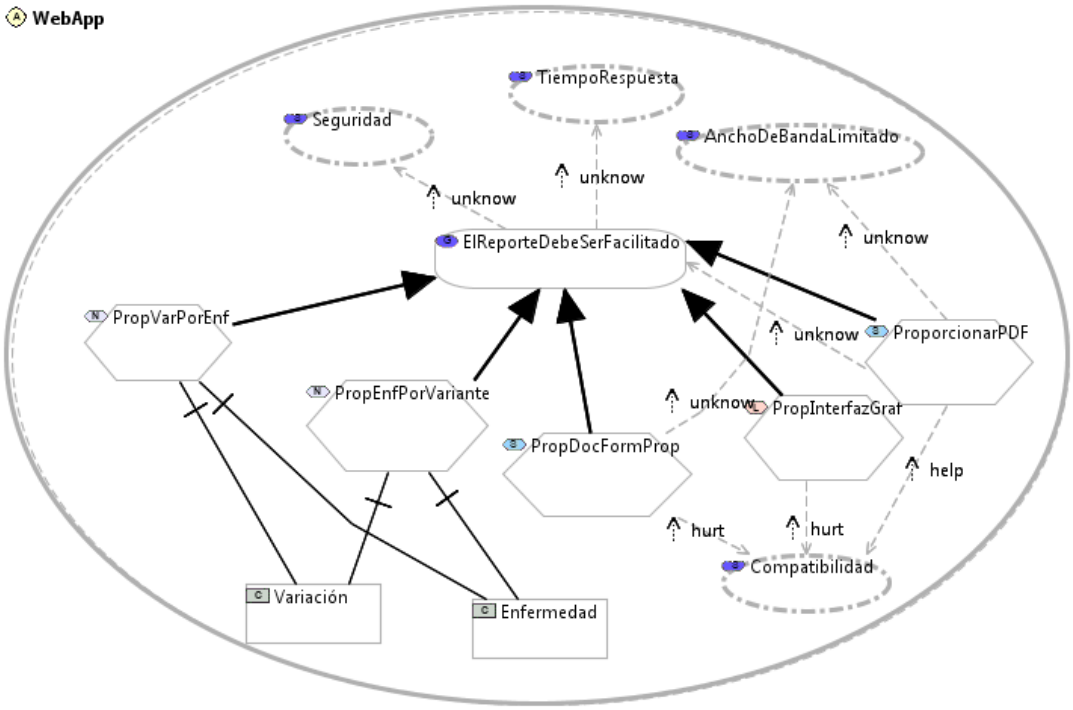


Figura 1.19. Modelo de requisitos RIA base.

originadas en el requisito "Proporcionar interfaz gráfica" también son etiquetadas debido a que el requisito sólo puede ser implementado del lado del cliente.

El segundo paso consiste en aplicar el algoritmo Óptimo de Pareto (Figura 1.18) para obtener el conjunto de configuraciones Cliente/Servidor. Para poder hacerlo, es necesario identificar a los requisitos que realizan contribuciones a las *Softgoals*, no se deberá considerar el requisito "Proporcionar interfaz gráfica" porque, como lo mencionamos antes, solo es posible implementarlo del lado del cliente. La Tabla 1.7 muestra los requisitos y *Softgoals* a utilizar.

Tabla 1.7. *Softgoals* y Requisitos detectados en el modelo de requisitos RIA base.

Softgoals	**Requisitos**
S1.- Seguridad	R1.- Proporcionar variaciones por enfermedad
S2.- Tiempo de respuesta	R2.- Proporcionar enfermedad por variante
S3.- Compatibilidad	R3.- Proporcionar PDF
S4.- Ancho de banda limitado	R4.- Proporcionar documento en formato propietario

Posteriormente, es necesario calcular todas las posibles configuraciones Cliente/Servidor, en donde C representa si el requisito se implementará en el Cliente y S si se implementará en el Servidor, como puede verse en la Tabla 1.8, de la columna 2 a la columna 5. Después se deberán obtener dos matrices, Matriz Cliente (1.2) y Matriz Servidor (1.3), las cuales representan la contribución de cada requisito a cada una de las *Softgoals*, por ejemplo, la fila 3 (requisito "Proporcionar PDF"), columna 4 (*Softgoal* "Tiempo de respuesta") muestra +1 en la Matriz Cliente, por lo que es una contribución del tipo "*Help*" si el requisito es implementado en el Cliente, en cambio, en la Matriz Servidor muestra -1, indicando una contribución del tipo "*Hurt*" si el requisito se implementa en el Servidor. Con las matrices se calculará el resultado de las contribuciones de los requisitos a las *Softgoals*, el cual es mostrado de la columna 6 a la columna 9 de la Tabla 1.8. Por último, se indica si la configuración está o no en la Frontera de Pareto (última columna).

$$M_{ij}^{k} = \begin{pmatrix} -1 & +1 & 0 & 0 \\ -1 & +1 & 0 & 0 \\ -1 & +1 & +1 & -1 \\ -1 & +1 & -1 & -1 \end{pmatrix} \tag{1.2}$$

$$M_{ij}^{k} = \begin{pmatrix} +1 & -1 & 0 & 0 \\ +1 & -1 & 0 & 0 \\ +1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 \end{pmatrix} \tag{1.3}$$

En la Tabla 1.8, las filas de color gris son la Frontera de Pareto. Finalmente, en el tercer paso (ver Figura 1.18), el diseñador puede seleccionar de la Frontera de Pareto la configuración que mejor satisfaga las necesidades del *stakeholder*, es decir, considerando a las *Softgoals* de mayor interés. Por mencionar un ejemplo, tenemos que la configuración X1 (Tabla 1.8) es la mejor opción si la *Softgoal* "Tiempo de respuesta" es la prioridad, en la solución cada requisito será implementado de lado del Cliente y, de acuerdo con el resultado positivo de la columna 10 (Σ), las *Softgoals* restantes serán maximizadas o permanecerán igual, es decir, no serán afectadas negativamente. Por otro lado, la mejor opción respecto a la *Softgoal* "Seguridad" es la configuración X16 (Figura 1.20), de acuerdo con Σ (-2), mejorar la seguridad afectará negativamente en algunas de las *Softgoals*.

Tabla 1.8. La posible distribución de los requisitos entre el servidor y el cliente a través de la compensación de las *softgoals*.

Configuración	**R1**	**R2**	**R3**	**R4**	**F(S1)**	**F(S2)**	**F(S3)**	**F(S4)**	Σ	**Frontera de Pareto**
X1	C	C	C	C	-4	+4	0	+2	+2	Si
X2	C	C	C	S	-2	+2	0	0	0	No debido a X5
X3	C	C	S	C	-2	+2	0	0	0	No debido a X6
X4	C	C	S	S	0	0	0	-2	-2	No debido a X13
X5	C	S	C	C	-2	+2	0	+2	+2	Si
X6	C	S	C	S	0	0	0	0	0	Si
X7	C	S	S	C	0	0	0	0	0	Si (como X6)
X8	C	S	S	S	+2	-2	0	-2	-2	No debido a X14
X9	S	C	C	C	-2	+2	0	+2	+2	Si (como X5)
X10	S	C	C	S	0	0	0	0	0	Si (como X6)
X11	S	C	S	C	0	0	0	0	0	Si (como X6)
X12	S	C	S	S	+2	-2	0	-2	-2	No debido a X14
X13	S	S	C	C	0	0	0	+2	+2	Si
X14	S	S	C	S	+2	-2	0	0	0	Si
X15	S	S	S	C	+2	-2	0	0	0	Si (como X14)
X16	S	S	S	S	+4	-4	0	-2	-2	Si

El resto de configuraciones de la Frontera de Pareto (Tabla 1.8) son configuraciones intermedias que estarán disponibles para que el diseñador pueda seleccionarlas de acuerdo a la compensación de las *Softgoals* que necesite, por ejemplo, las configuraciones X6 y X14, ambas tienen $\Sigma = 0$, por lo tanto, todas las *Softgoals* están balanceadas. En la configuración X14, los requisitos R1, R2 y R4 son ubicados en el servidor y R2 en el cliente, la solución quizá proporcione una buena compensación en caso de que la seguridad sea una prioridad por parte del *stakeholder*, pero no se mejorarán el resto de las *Softgoals*, incluso, la *Softgoal* "Tiempo de respuesta" será afectada negativamente.

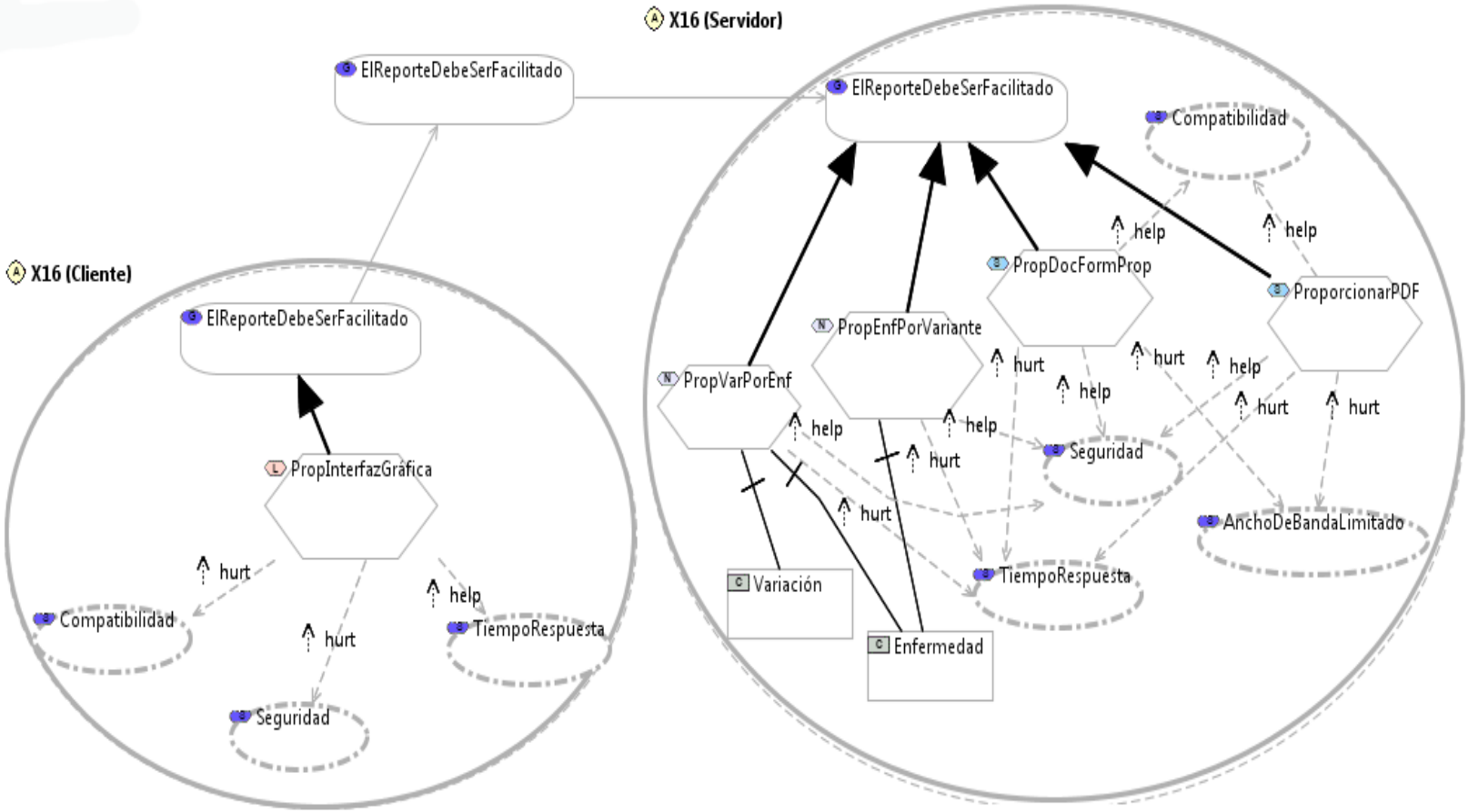


Figura 1.20. Configuración X16: distribución de los requisitos entre el cliente y el servidor.

1.5. Implementación

Esta sección aborda la descripción de la implementación de cada uno de los apartados descritos en la sección 1.2 de la tesis doctoral, por lo tanto, la implementación sirve como prueba de concepto de los objetivos planteados en la investigación realizada en la tesis. A continuación, se puntualiza cómo es que los objetivos han sido sustentados a través de cada uno de los apartados establecidos en esta sección.

- ***Sección 1.5.1:*** "Editor Gráfico para la Especificación de Requisitos Web (*WebREd*)". En este apartado se explica la implementación de un editor gráfico para el análisis y especificación de requisitos por medio del marco de modelado orientado a objetivos *i**. Por lo tanto, se complementan los objetivos específicos uno y dos (Sección 1.2). Con el editor gráfico, se establecen mecanismos para la comprensión de los objetivos de negocio, y por medio del editor, los *stakeholders* podrán razonar sobre ellos. Cabe destacar que los mecanismos están integrados dentro de una etapa de análisis de requisitos orientada a objetivos (Sección 1.3.1 y Capítulo 3), por tanto, brindan soporte para representar las expectativas reales de los *stakeholders* de la aplicación Web.
- ***Sección 1.5.2:*** "Transformaciones Modelo a Modelo". La sección permite dar soporte al objetivo tres de la tesis doctoral, gracias a que por medio de las transformaciones modelo a modelo es posible la generación de las estructuras de los modelos conceptuales de la aplicación Web, que posteriormente, deberán ser completados por el diseñador. Por lo tanto, la sección 1.5.2 ofrece un alto grado de automatización en el desarrollo de aplicaciones Web por medio de un conjunto de transformaciones y con esto brinda soporte a la generación de código. Para más información consultar los artículos listados en la sección 1.1.3 así como el Capítulo 3 de la tesis doctoral.
- ***Sección 1.5.3:*** "Trazabilidad de Requisitos". La sección proporciona el soporte necesario para sustentar la aplicabilidad del objetivo específico cuatro de la sección 1.2 de la tesis. Por medio de las transformaciones modelo a modelo es posible ofrecer al diseñador soporte para trazabilidad de requisitos. Para más información consultar los artículos listados en la sección 1.1.3 y el Capítulo 4 de la tesis doctoral.
- ***Sección 1.5.4:*** "Optimización de Pareto". La implementación que complementa la tesis doctoral soportar el objetivo cinco de la sección 1.2. Básicamente, el apartado 1.5.4 asiste al diseñador de la aplicación Web al momento de la selección de los requisitos funcionales a implementar a través de alternativas de diseño que consideren el balance y maximización de los requisitos no-funcionales. Para más información consultar el Capítulo 5 y el Apéndice C de la tesis.

Por otra parte, la propuesta presentada en la tesis doctoral ha seguido un proceso de desarrollo basado en MDA. Por lo tanto, para llevar a cabo la implementación ha sido necesaria la combinación de un conjunto de tecnologías MDD entre las que destacan (i) la plataforma de desarrollo *Eclipse* [21], (ii) la tecnología EMF (*Eclipse Modeling Framework*) [22] y GMF (*Graphical Modeling Framework*) [32] y (iii) el lenguaje para transformaciones entre modelos ATL (*Atlas Transformation Language*) [8]. Cada una de las tecnologías utilizadas son descritas a continuación.

- ***Eclipse***. Es un entorno de desarrollo integrado (*Integrated Development Enviroment*, IDE) de código abierto multiplataforma desarrollado originalmente por IBM[9]. *Eclipse* es ahora desarrollado por la Fundación *Eclipse* [21], una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto. La característica principal del IDE es que es una plataforma de programación utilizada para crear entornos integrados de desarrollo que puede ser extendida por medio de *plugins* con el fin de añadir más características y nuevas funcionalidades.
- ***Eclipse Modeling Framework (EMF)***. Es un marco de trabajo para modelado que ofrece facilidad de generación de código para construir herramientas y otras aplicaciones

[9] `http://www-01.ibm.com/software/rational/eclipse/`

basadas en un modelo de datos estructurado. Desde una especificación del modelo descrita en XMI (XML de Intercambio de Metadatos), *EMF* suministra herramientas y soporte en tiempo de ejecución para producir un conjunto de clases Java para el modelo, un conjunto de clases del tipo *Adapter* que permiten visualización y edición basándose en comandos del modelo, y un editor básico. Los modelos pueden ser especificados usando notación Java, documentos XML, o herramientas de modelado como *Rational Rose*[10], y después ser importados a *EMF*. Lo más importante de todo, *EMF* suministra las bases para la interoperabilidad con otras herramientas y aplicaciones basadas en *EMF*. Dentro de *EMF* encontramos un metamodelo para describir modelos llamado Ecore, este metamodelo incluye soporte en tiempo de ejecución para los modelos además de notificación de cambios.

- ***Graphical Modeling Framework (GMF)***. Es un marco de trabajo que permite el desarrollo de editores gráficos. GMF combina EMF y GEF (*Graphical Editing Framework*) para el desarrollo de editores gráficos *ad-hoc*. De tal forma que, los elementos creados con el editor gráfico, son una representación visual de cada concepto especificado en el metamodelo.
- ***Atlas Transformation Language (ATL)***. Es un lenguaje de transformación de modelos basado en los estándares del OMG [51] y OCL 2.0 [50]. ATL es un lenguaje declarativo e imperativo (híbrido) que permite la transformación entre modelos (M2M). Las construcciones declarativas son la opción preferida para escribir las transformaciones debido a que permiten expresar correspondencias, entre los elementos del modelo fuente y del modelo destino a partir de una serie de composiciones de reglas. Las construcciones imperativas proporcionan constructores para facilitar la especificación de correspondencias que de forma declarativa serían más complejas de implementar.
- ***Xpand***. Es un lenguaje especializado en la generación de código a partir de modelos EMF [71]. Xpand ofrece la posibilidad de escribir completas librerias de funciones para la lectura de modelos EMF y permite la utilización de las librerias de funciones desde código Java [35] o expresiones Xtend.
- ***Java***. Es un lenguaje de programación orientado a objetos, diseñado principalmente para tener el mínimo de dependencia del sistema operativo para su utilización. Actualmente, Java es uno de los lenguajes de programación mías populares [35].

Finalmente, en el apartado 1.5.5 de esta sección se presenta un caso de estudio para ejemplificar la aplicabilidad de la implementación de la propuesta descrita en la tesis doctoral. En el caso de estudio, se detalla la especificación de requisitos Web por medio del marco de modelado *i**, el soporte para trazabilidad de requisitos y la selección de las distintas alternativas de implementación de los requisitos funcionales mediante la optimización de las *Softgoals*.

1.5.1. Editor Gráfico para la Especificación de Requisitos Web (*WebREd*)

En este apartado son implementados los conceptos (sintaxis abstracta) del marco de modelado *i** y la clasificación de requisitos presentada en el Capítulo 1.3.1 de la tesis para proveer al diseñador con un editor gráfico para la especificación de requisitos Web. Como se mencionó anteriormente, la propuesta presentada en la tesis doctoral se encuentra alineada con MDA, por lo tanto, el editor WebREd corresponde al modelo independiente de computación (CIM). Concretamente, el metamodelo de requisitos implementado en la tesis doctoral ha sido extendido para incorporar los tipos de requisitos de *i** y la clasificación presentada en la sección 1.3.1. La Figura 1.3 muestra una captura de pantalla donde puede apreciarse la implementación del metamodelo en *Eclipse*. La sintáxis concreta del metamodelo para requisitos Web se ha implementado para desarrollar un editor gráfico (Figura 1.21) por medio de la tecnología GMF.

El editor (Figura 1.21) WebREd (*Web Requirements Editor*) brinda al diseñador de la aplicación Web una interfaz gráfica para especificar en un diagrama los requisitos funcionales y las *Softgoals* de la aplicación Web. WebREd permite la creación, modificación y actualización de la especificación de requisitos Web (diagramas). Además, cada una de las propiedades de

[10] http://www-01.ibm.com/software/awdtools/developer/rose/

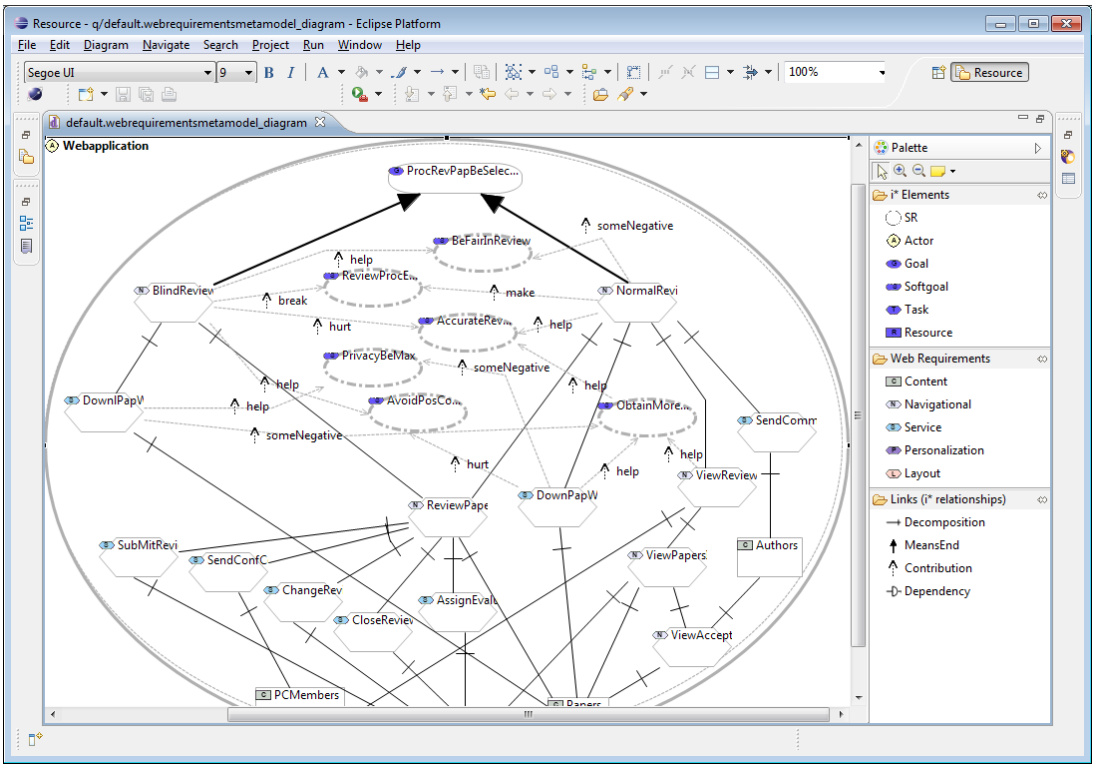


Figura 1.21. Editor WebREd para la especificación de requisitos Web con *i**.

los elementos del marco de modelado *i** pueden ser modificadas seleccionando cada elemento en la vista de propiedades (Figura 1.22).

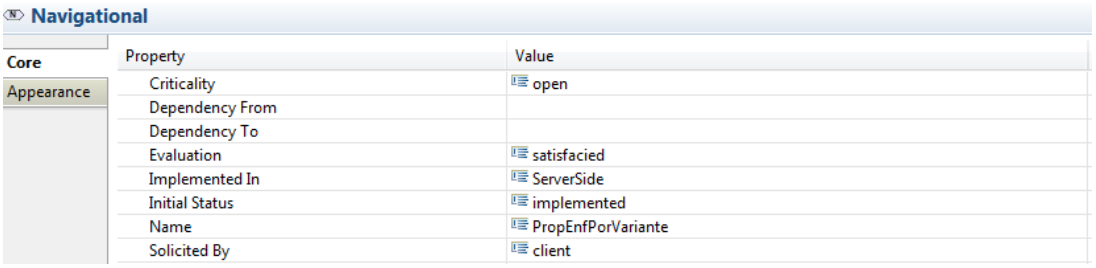


Figura 1.22. Vista de propiedades del editor WebREd.

Es importante destacar que además de poder especificar modelos de requisitos Web (*Content*, *Navigational*, *Personalization*, *Service* y *Layout*), WebREd permite la creación de diagramas (modelos orientados a objetivos) comunes de *i** gracias a que coloca a disposición del diseñador los elementos clásicos (*Goal*, *Task*, *Resource* y *Softgoal*) del marco de modelado *i** como puede verse en la Figura 1.23, en donde se muestran los elementos que permiten la creación de cada elemento *i** realizando un *drag and drop* de cada elemento sobre el diagrama.

1.5.2. Transformaciones Modelo a Modelo

Con el fin de aprovechar cada una de las ventajas que ofrece la ingeniería dirigida por modelos y con esto automatizar el paso del CIM al PIM, se han desarrollado una serie de

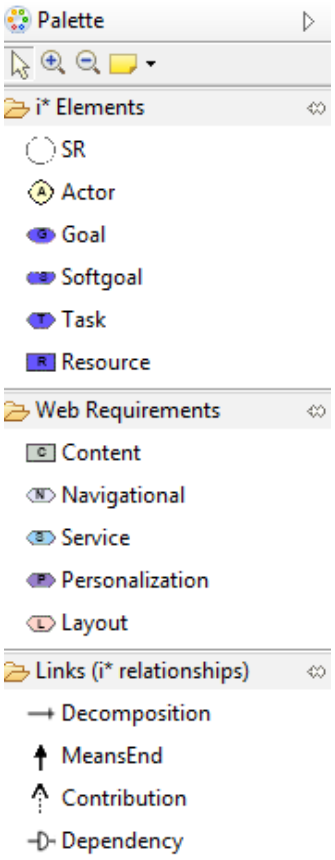


Figura 1.23. Elementos para el diseño del modelo de requisitos Web con *i**.

reglas de transformación definidas formalmente en el lenguaje QVT (sección 1.3.1). Con esta idea, las transformaciones deben de ser vistas desde una perspectiva de modelado como modelos de transformaciones [11]. En la investigación asociada a la tesis doctoral, QVT se ha utilizado como metamodelo para formalizar las transformaciones modelo a modelo abstrayéndolas como modelos, con el fin de mejorar la comprensión del proceso de transformación. Sin embargo, una vez que las transformaciones han sido modeladas, necesitan implementarse. Como se ha mencionado anteriormente, se ha utilizado el lenguaje ATL para implementar cada una de las reglas definidas en la sección 1.3.1 y de esta forma ejecutarlas con el fin de realizar el proceso de normalización de forma automática. Cabe destacar que utilizar ATL para implementar transformaciones QVT es factible debido a la alineación que existe entre la arquitectura de ambos de lenguajes [36].

La Figura 1.24 muestra el código ATL referente a la implementación de la regla *Content2DomainClass* (descrita en detalle en la sección 1.3.1). La regla tiene como objetivo crear las clases del modelo de dominio de A-OOH.

1.5.3. Trazabilidad de Requisitos

El soporte para la trazabilidad de los requisitos a través de los modelos conceptuales de la aplicación Web se efectúa por medio de una extensión [9] del metamodelo de *weaving*, explicado previamente en la sección 1.3.2 de la tesis doctoral. La extensión para trazabilidad del metamodelo de *weaving* permite generar por medio de transformaciones un modelo de trazabilidad. El modelo de trazabilidad puede ser visualizado y modificado mediante la herramienta AMW

```
--For each Content element in the i* Requirements Model, a UML Class will be created in the Domain Model.
rule Content2Class
{
    from
        vContent : MMiStar!Content
    to
        vClass : UML!Class (name<-vContent.name),
 __traceLink : traceability!TraceLink (
      name <- 'Content2DomainClass',
      sourceElements <- Sequence {__LinkEnd_vContent},
      targetElements <- Sequence {__LinkEnd_vClass},
      model <- thisModule.__wmodel, ruleName <-'Content2DomainClass', description<-'From each Content element
      in RequirementsModel, this rule creates a UML Class in DomainModel'),
    __LinkEnd_vContent : traceability!TraceLinkEnd (element <- __elementRef_vContent ),
    __elementRef_vContent : traceability!ElementRef (ref <- vContent.__xmiID__, modelRef <- thisModule.__model_IN),
    __LinkEnd_vClass : traceability!TraceLinkEnd (element <- __elementRef_vClass ),
    __elementRef_vClass : traceability!ElementRef (ref <- vClass.__xmiID__, modelRef <- thisModule.__model_OUT)

}
```

Figura 1.24. Implementación de la transformación *Content2DomainClass* en el lenguaje ATL.

(*Atlas Model Weaver*) [19]. De esta forma, es posible visualizar las correspondencias entre cada uno de los requisitos funcionales expresados en el modelo de requisitos *i** con cada uno de los elementos de los modelos conceptuales de la aplicación Web [2].

1.5.4. Optimización de Pareto

El algoritmo Optimización de Pareto ha sido implementado por medio del lenguaje de programación Java. Adicionalmente, ha sido necesario el uso de las clases base (*core*) de EMF y el lenguaje Xpand con el fin de ser capaz de leer el modelo de requisitos *i**. El algoritmo se ha implementado como un *plugin* de Eclipse para su fácil incorporación con el editor WebREd.

La Optimización de Pareto es calculada en tres pasos, el primero de ellos consiste en establecer la lista de prioridades para las *Softgoals*, es decir, el diseñador ordenará las *Softgoals* indicando cual es la de mayor importancia de acorde a lo establecido por los *stakeholders*. En el segundo paso, son extraídos los requisitos funcionales así como las contribuciones que estos realizan a las *Softgoals* y en el tercer paso se calcula la Optimización de Pareto. La Figura 1.25 muestra la ventana principal de la implementación del algoritmo. La ventana esta dividida en 3 pestañas (*Configuration*, *Requirements Contributions*, *The Pareto Front*), cada una de ellas corresponde a los 3 pasos necesarios para la ejecución del algoritmo.

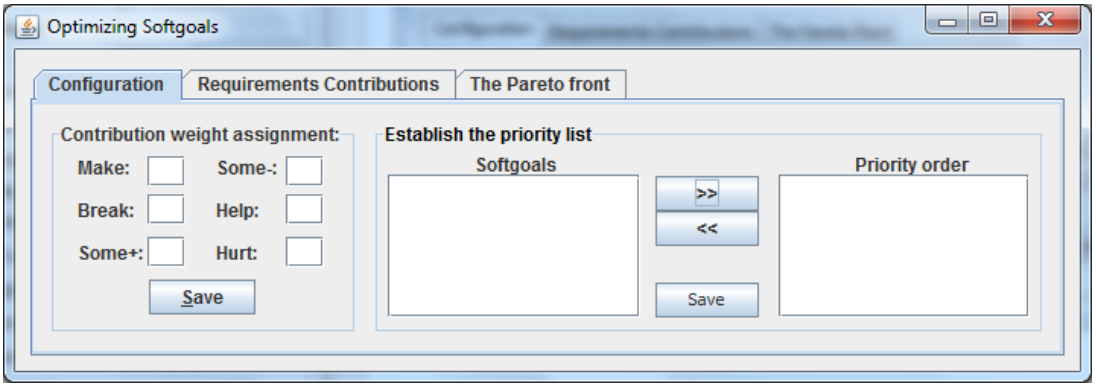


Figura 1.25. Ventana principal del *plugin* Optimización de Pareto.

1.5.5. Caso de estudio

En este apartado se presenta un caso de estudio para demostrar la aplicabilidad de nuestra propuesta. El caso de estudio es el utilizado la sección 1.3.2 y en el Capítulo 5 acerca de una aplicación Web para la gestión de conferencias (*Conference Management System*). Recordemos que el propósito de la aplicación Web es brindar soporte al proceso de envío, evaluación y selección de los artículos para una conferencia.

El primer paso consiste en la especificación de los requisitos para la aplicación Web "*ContentManagementSystem*". La Figura 1.26, muestra la especificación de requisitos de la aplicación Web (modelo de requisitos). El objetivo de la aplicación Web consiste en "*Process of review of papers be selected*", para satisfacer el objetivo, es necesario la implementación de alguno de los requisitos navegacionales "*Blind review process*" y "*Normal review process*". En el ejemplo el objetivo es logrado a través del requisito navegacional "*Blind review process*".

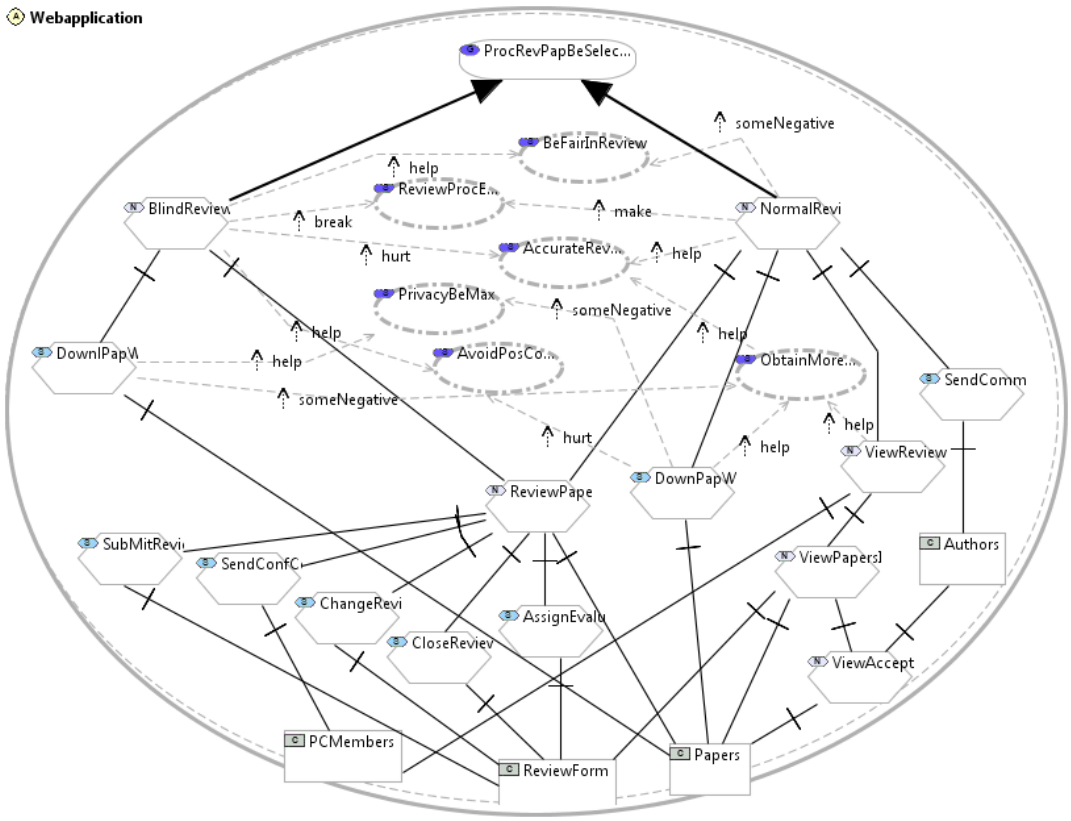


Figura 1.26. Especificación de requisitos Web con *i** para el caso de estudio.

El siguiente paso consiste en ejecutar las reglas de transformación para derivar automáticamente la estructura del modelo de dominio de A-OOH. En la Figura 1.27 se muestra la regla de transformación *Service2ClassOperation* (definida en la sección 1.3.1) implementada en ATL.

En este caso de estudio, el modelo de dominio de A-OOH se muestra en la Figura 1.28. Como se puede observar, el modelo está constituido por un *UML-Package*, dentro del cual se encuentran los elementos comunes de un diagrama de clases UML tales como las clases, las operaciones dentro de las clases y las asociaciones.

El próximo paso es la obtención de la estructura del modelo de navegación de A-OOH. Para esto se han implementado en ATL (Figura 1.29) las reglas de transformación presentadas en la sección 1.3.1. El proceso de derivación de un modelo de navegación permite obtener un modelo

```
--For each Service element associated with a Content element in the i* Requirements Model,
--a UML Operation within a Class will be created in the Domain Model.
rule Service2ClassOperation
{
    from
        vDecomposition : MMiStar!Decomposition(vDecomposition.isParentService())
    to
        vOperation : UML!Operation (name <-vDecomposition.Element.name, class<-vDecomposition.SubElement),
         __traceLink : traceability!TraceLink (
          name <- 'Service2ClassOperation',
          sourceElements <- Sequence {__LinkEnd_vDecomposition},
          targetElements <- Sequence {__LinkEnd_vOperation},
          model <- thisModule.__wmodel, ruleName <-'Service2ClassOperation',
          description<-'From each Service element in RequirementsModel, this rule creates a UML Operation in DomainModel'),
        __LinkEnd_vDecomposition : traceability!TraceLinkEnd (element <- __elementRef_vDecomposition),
        __elementRef_vDecomposition : traceability!ElementRef (ref <- vDecomposition.__xmiID__,
            modelRef <- thisModule.__model_IN),
        __LinkEnd_vOperation : traceability!TraceLinkEnd (element <- __elementRef_vOperation),
        __elementRef_vOperation : traceability!ElementRef (ref <- vOperation.__xmiID__, modelRef <- thisModule.__model_OUT)

}

rule CreateAssociationWithProperties (vPropSource : MMiStar!Decomposition, vPropDest : MMiStar!Decomposition)
{
    to
        vAssociation :  UML!Association (name <- vPropSource.name + '--->' + vPropDest.name,
        ownedEnd <-  Set {vPropSrc, vPropDst}),
        vPropSrc : UML!Property (name <- 'src', type <- vPropSource, association <-vAssociation),
        vPropDst   : UML!Property (name <- 'dst', type <-vPropDest, association <-vAssociation)
    do
    {
        thisModule.vModeloSalida.packagedElement<-vAssociation;
    }
}
```

Figura 1.27. Reglas ATL para obtener las operaciones de las clases del modelo de dominio A-OOH.

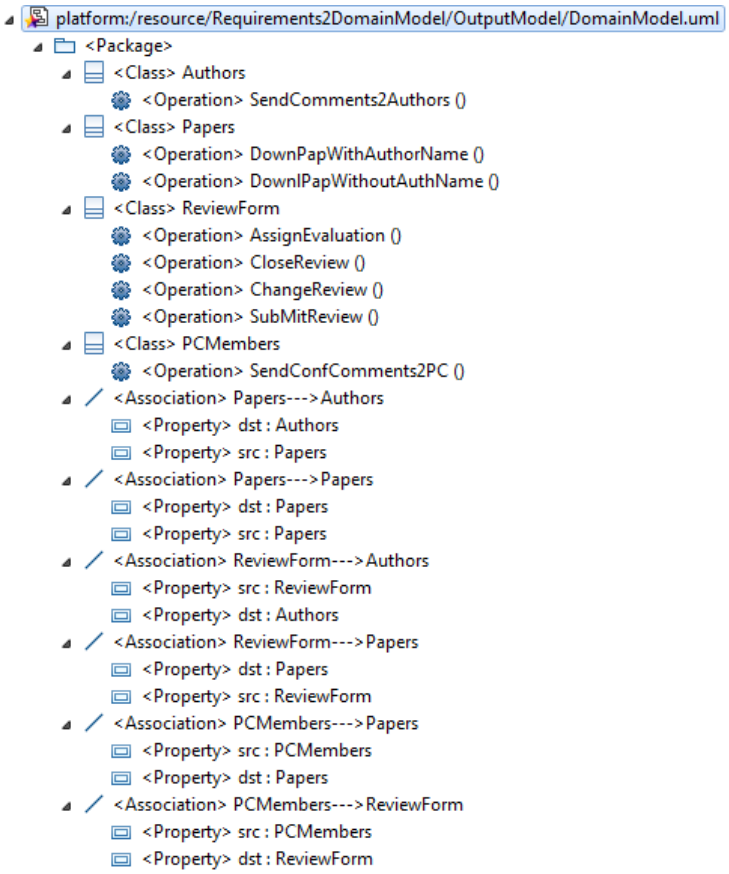


Figura 1.28. Modelo de dominio A-OOH.

conceptual donde se especifiquen las rutas que el usuario de la aplicación Web podrá utilizar para desplazarse por el contenido. En la Figura 1.30 se muestra el modelo de navegación obtenido automáticamente a partir del modelo de requisitos. Como se describió en la sección 1.3.1, el modelo de navegación esta constituido por clases, enlaces y nodos navegacionales.

```
-- From i* SR model to create the Navigational Model
rule StrategicRelationshipModel2NavigationalModel{
    from
        vModeloEntrada : WebRequirementsMetamodel!SR
    to
        vModeloSalida : NavigationalMetamodel!NavigationalModel (
            description <- 'Modelo de Navegación',--vModeloEntrada.Description,
            hasNodes<-vModeloEntrada.iElement->select(e|e.oclIsTypeOf(WebRequirementsMetamodel!Content))),
        vHomeNavClass : NavigationalMetamodel!Navigational_C_Collection(isInitial<-true, name<-'Home(Menu)', personalizationRules<-'None')
    do
    {
        vModeloSalida.hasNodes<- vHomeNavClass;
    }
}

rule NavPers2NavClass
{
    from
        vContent : WebRequirementsMetamodel!Content
    to
        vClass : NavigationalMetamodel!NavigationalClass (name<-vContent.name, isInitial<-false, personalizationRules<-'None'),
        vTransversalLink : NavigationalMetamodel!TransversalLink( name <-'Home(Menu)' + '->' + vContent.name,
            origin<-NavigationalMetamodel!Navigational_C_Collection->allInstances()->
                                    select(e|e.oclIsTypeOf(NavigationalMetamodel!Navigational_C_Collection)),
            target<-vClass,
            Owns2NavModel<-NavigationalMetamodel!NavigationalModel->allInstances()->
                                    select(e|e.oclIsTypeOf(NavigationalMetamodel!NavigationalModel)),
            personalizationRules<-'None', filterOrigin<-'None', filterTarget<-'None', indexedBy<-'Undefined')
    }
}
```

Figura 1.29. Extracto de la regla ATL para obtener el modelo de navegación.

Al mismo tiempo que ocurre la generación de los modelos de dominio y navegación de A-OOH se obtiene el modelo de trazabilidad (Sección 1.3.2). Esto es debido a que las reglas para crear los enlaces para trazabilidad están introducidas en las reglas ATL que derivan los modelos conceptuales. La Figura 1.31 muestra la parte de la transformación ATL donde se genera el modelo de trazabilidad dentro de la regla de transformación que derivan el modelo de dominio.

El próximo paso consiste en visualizar el modelo de trazabilidad obtenido por medio de la herramienta AMW *Atlas Model Weaver* [19]. AMW es una herramienta que permite visualizar los enlaces contenidos en los modelos de *weaving*. En la Figura 1.32 se muestra el soporte para trazabilidad de A-OOH, en el lado izquiero de la imagen se visualiza el modelo de requisitos, al centro el modelo de trazabilidad y en el lado derecho se puede observar el modelo de dominio. Cuando el diseñador selecciona algún elemento de cualquier modelo (requisitos, trazabilidad o dominio) la herramienta resalta los respectivos elementos con los que corresponde el elemento seleccionado. Por último, en la parte inferior de la figura se pueden ver las propiedades de los elementos del modelo de trazabilidad.

Como se explicó en la sección 1.3.1, una vez que han sido obtenidos las estructuras de los modelos de dominio y navegación el diseñador solo tendrá que refinarlos, por ejemplo, en el modelo de dominio el usuario tendrá que añadir de forma manual la cardinalidad de las asociaciones entre las clases. Más información se detalla en el Capítulo 3 de la tesis.

Una vez que se tiene un prototipo inicial de los modelos conceptuales de la aplicación, vamos a suponer que el diseñador necesita optimizar la seguridad de la aplicación Web debido a una solicitud expresa de los *stakeholders*. Para esto, necesita la ejecución del *plugin* Optimización de Pareto y con ello obtener como resultado un conjunto de configuraciones que le permitan saber que requisitos funcionales implementar de acuerdo a las *Softgoals* que se necesiten optimizar.

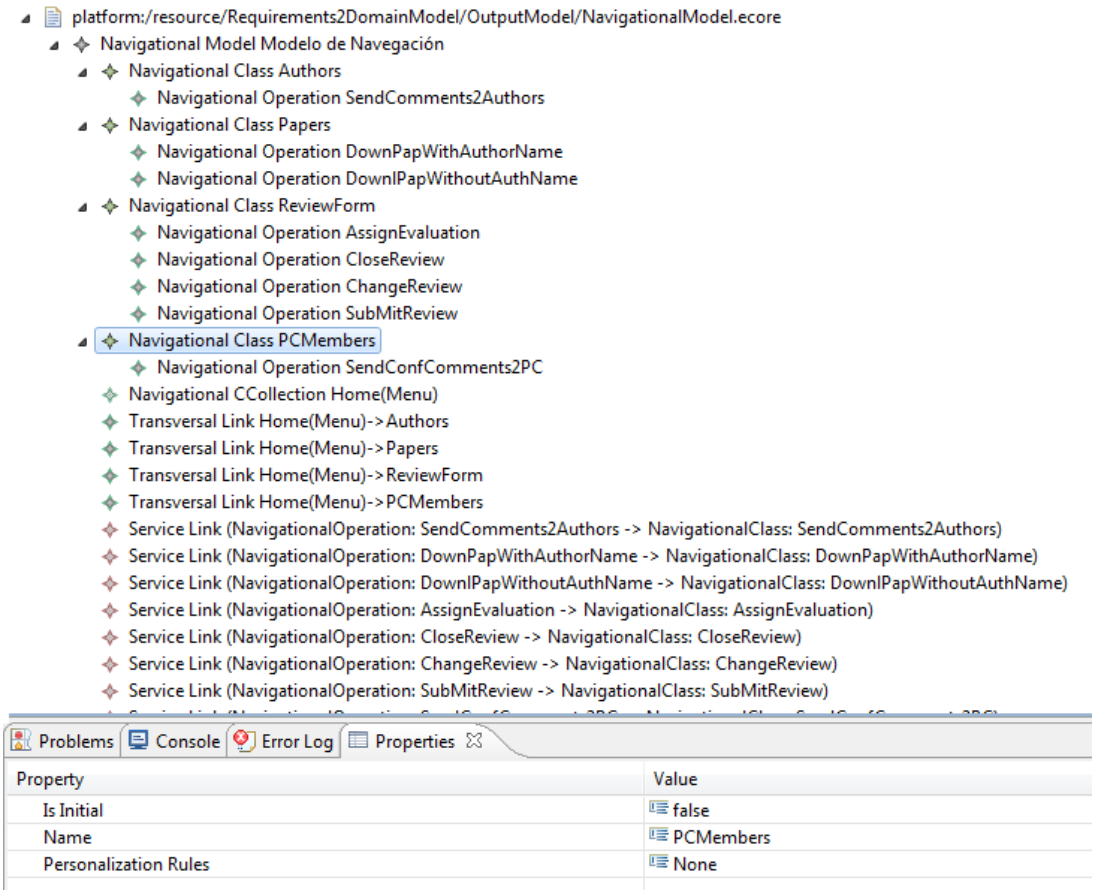


Figura 1.30. Modelo de navegación del caso de estudio.

```
--http://www.eclipse.org.uml2/2.0.0./UML
module RequisitosDominioTrazabilidad;
create DomainModel : UML, TraceModel : traceability  from RequirementsModel : MMiStar;

uses "Mode2001.library";

helper def: __wmodel : traceability!TraceModel = OclUndefined;
helper def: __model_IN : traceability!TraceModelRef = OclUndefined;
helper def: __model_OUT : traceability!TraceModelRef =OclUndefined;

entrypoint rule InitTrace() {
  to
    wmodel : traceability!TraceModel ( name <-'AOOH(Traceability)',
        description <-'This is the Traceability Model, created to support traceability from Requirements Model to Domain Model in A-OOH approach',
      wovenModels <- Sequence {model_IN, model_OUT}),
    model_IN : traceability!TraceModelRef (name <- 'RequirementsModel',
        description<-'This is a reference to the A-OOH Requirements Model specified using the iStar framework'),
    model_OUT : traceability!TraceModelRef (name <- 'DomainModel',
        description<-'This is a reference to the A-OOH Domain Model generated from the transformation')
  do {
    thisModule.__wmodel <- wmodel;
    thisModule.__model_IN <- model_IN;
    thisModule.__model_OUT <- model_OUT;
  }
}
```

Figura 1.31. Extracto de la regla ATL para obtener el modelo de dominio en donde se crea el modelo de trazabilidad.

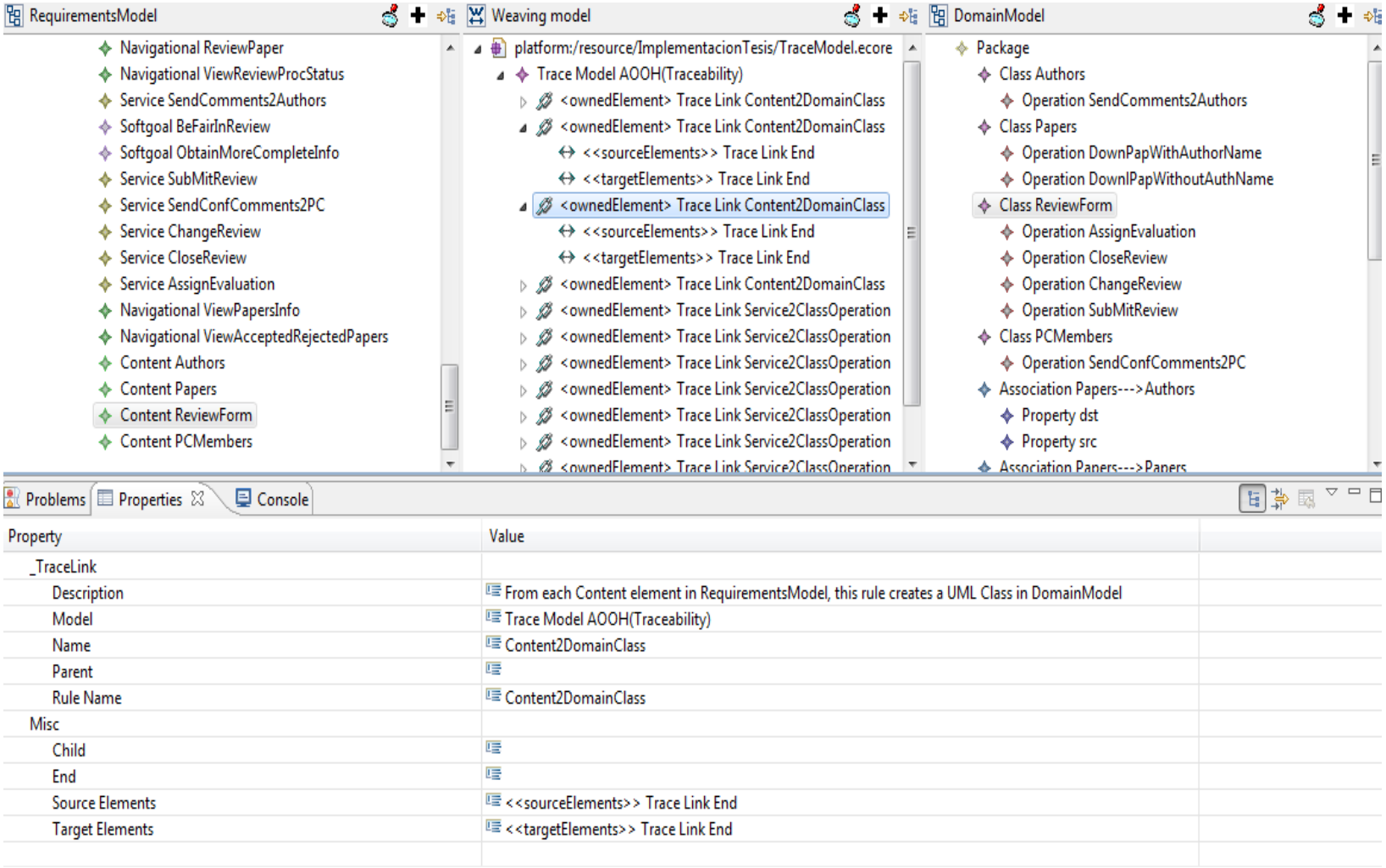


Figura 1.32. Visualización del modelo de trazabilidad en la herramienta AMW.

El *plugin* para la Optimización de Pareto utiliza como modelo de entrada el modelo de requisitos *i**, así, por medio de las clases EMF y el lenguaje Xpand el modelo de requisitos es leído para extraer todos los requisitos, *Softgoals* y los enlaces de contribución (*contribution links*) contenidos en él. La Figura 1.33 muestra la ventana principal del *plugin*, la cual contiene un *tab pane* dividido en 3 pestañas (*tabs*). La primera pestaña se llama *Configuration*, en el costado izquierdo el diseñador puede cambiar los valores establecidos por *default* que representan la fuerza de las contribuciones realizadas por parte de los requisitos funcionales a las *Softgoals*, los valores son utilizados en el cálculo de la matriz de pesos (Sección 1.3.2). En el costado derecho de la misma figura se muestra la sección en donde el diseñador establece la lista de prioridades de *Softgoals*. En este caso de estudio, el diseñador necesita maximizar la *Softgoal Privacy Be Maximized* que es la relacionada con la seguridad de la aplicación Web.

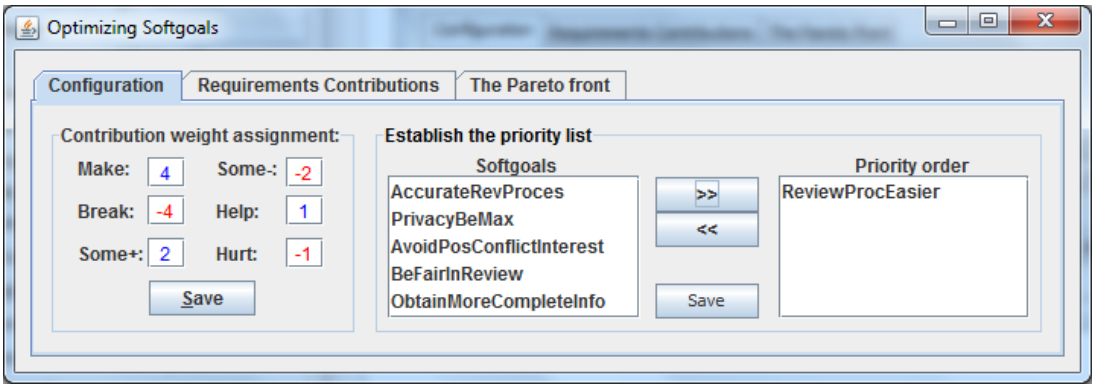


Figura 1.33. Ventana principal del *plugin* Optimización de Pareto (pestaña de configuración).

En la Figura 1.34 se muestra la pestaña *Requirements Contributions* en donde se puede ver en un listado el resultado del *plugin* al extraer del modelo de requisitos *i** (Figura 1.26) aquellos requisitos funcionales que realicen algún tipo de contribución a las *Softgoals*. En la primera columna se encuentran listados los requisitos funcionales extraídos, cabe destacar que los requisitos funcionales listados son solo aquellos que realizan contribuciones a las *Softgoals* independientemente de su estado, es decir, si se encuentran implementados o no en los modelos conceptuales de la aplicación Web. Las seis columnas restantes corresponden con cada una de las *Softgoals* extraídas del modelo de requisitos. Finalmente, en la parte inferior de la pestaña *Requirements Contributions* se encuentra el botón para calcular el resultado del algoritmo Optimización de Pareto, es decir la Frontera de Pareto.

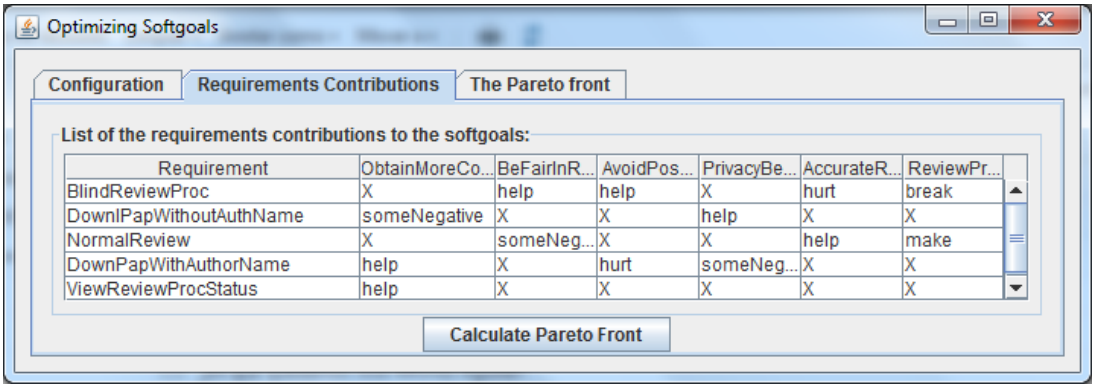
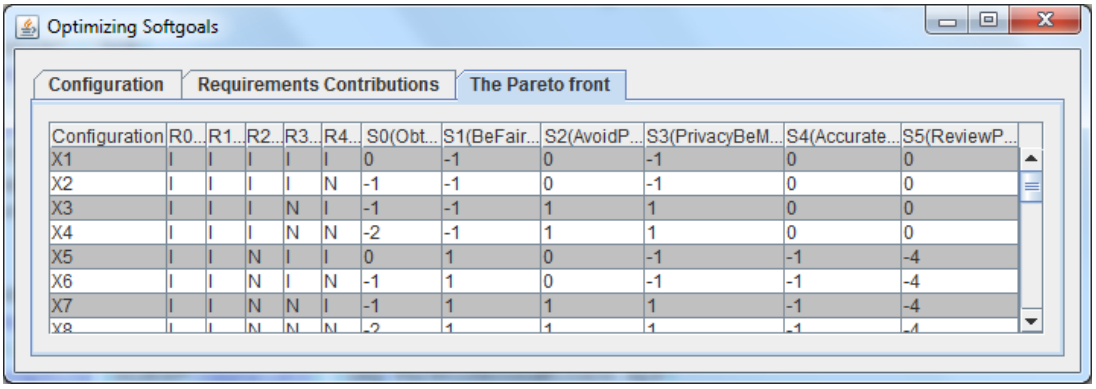


Figura 1.34. *Plugin* Optimización de Pareto. Las contribuciones de los requisitos a las *Softgoals*.

El resultado de la ejecución del algoritmo se puede ver en la Figura 1.35. La pestaña *The Pareto Front* muestra una tabla similar a la descrita en la sección 1.3.2 en donde son mostrados los resultados de la ejecución del algoritmo. Las configuraciones posibles se enlistan desde la primera columna llamada *Configurations* hasta la columna seis, la variable "I" representa el estado implementado para un requisito funcional determinado y la variable "N" significa que ese requisito funcional no debe de ser implementado en los modelos conceptuales de la aplicación Web. Las configuraciones que están en la Frontera de Pareto se resaltan en color gris para que sean fáciles de identificar por parte del diseñador.



Configuration	R0...	R1...	R2...	R3...	R4...	S0(Obt...	S1(BeFair...	S2(AvoidP...	S3(PrivacyBeM...	S4(Accurate...	S5(ReviewP...
X1	I	I	I	I	I	0	-1	0	-1	0	0
X2	I	I	I	I	N	-1	-1	0	-1	0	0
X3	I	I	I	N	I	-1	-1	1	1	0	0
X4	I	I	I	N	N	-2	-1	1	1	0	0
X5	I	I	N	I	I	0	1	0	-1	-1	-4
X6	I	I	N	I	N	-1	1	0	-1	-1	-4
X7	I	I	N	N	I	-1	1	1	1	-1	-4
X8	I	I	N	N	N	-2	1	1	1	-1	-4

Figura 1.35. *Plugin* Optimización de Pareto. La Frontera de Pareto.

El diseñador asistido por la tabla de la Frontera de Pareto (Figura 1.35) podrá ser capaz de seleccionar la configuración final de acuerdo con la lista de prioridades de las *Softgoals*. En este caso de estudio, para maximizar la seguridad de la aplicación Web, el diseñador podrá elegir entre las configuraciones "X3" y "X7".

Finalmente,, la opción a implementar es la configuración "X3" debido a que la configuración "X7" maximiza la *Softgoal* relacionada con la seguridad (+1) pero afecta demasiado (-4) a la *Softgoal Review Process Easier* la cual esta vinculada con la usabilidad de la aplicación. La configuración "X8" no es considerada por que no está en la Frontera de Pareto.

1.6. Conclusiones

A continuación, se sintetiza el trabajo descrito en la tesis doctoral. En primer lugar, en la Sección 1.6.1 se listan y comentan las principales aportaciones de la investigación descrita en la tesis. Posteriormente, en la Sección 1.6.2 se mencionan las limitaciones dentro del ámbito de la investigación realizada y se presenta el trabajo que se realiza actualmente así como las posibles contribuciones futuras.

1.6.1. Aportaciones

Como se puntualizó anteriormente, la Web cambia constantemente debido a la evolución constante en las tecnologías de implementación. Por consiguiente, las metodologías ingenieriles para el desarrollo de este tipo de aplicaciones necesitan adaptarse a los cambios con el fin de ofrecer al diseñador una aproximación sistemática e integral para el desarrollo de aplicaciones Web.

En este sentido, en la tesis doctoral se ha presentado una propuesta dirigida por modelos en ingeniería Web, la cual permite la especificación de los requisitos funcionales y no-funcionales de la aplicación Web de manera integral, sistemática y bien estructurada (Capítulo 3).

En la propuesta, los requisitos son especificados en un modelo por medio del marco de modelado orientado a objetivos *i** y a razón de que el método Web A-OOH se basa en MDA, el modelo, con los requisitos especificados, corresponde al nivel CIM de esta arquitectura. Gracias a eso, en el Capítulo 3 se demostró que es posible la derivación automática de la estructura de los modelos conceptuales de dominio y navegación del método Web A-OOH a partir de la especificación de requisitos, con lo cual, es posible asegurar que los requisitos de la aplicación Web han sido reflejados en los modelos conceptuales a nivel PIM. Por lo tanto, la propuesta permite a los diseñadores disminuir el nivel de complejidad en el desarrollo de aplicaciones Web, ahorrando tiempo y esfuerzo, por lo que el costo del proyecto disminuye.

Por otra parte, debido a la naturaleza cambiante de la Web y a su audiencia heterogénea, mantener una etapa de análisis y especificación de requisitos resulta cada vez más complicado. A razón de esto, en la tesis se describe el soporte ofrecido al diseñador de la aplicación Web por medio de una gestión integral de requisitos. Concretamente, se asiste al diseñador en aspectos como la trazabilidad de los requisitos de CIM a PIM (Sección 1.1.3), el análisis de impacto en los requisitos (Capítulo 4) y en la implementación de requisitos funcionales en base a distintas alternativas de diseño considerando la maximización de los requisitos no-funcionales (Capítulo 5).

Además, se ha realizado una implementación en la plataforma *Eclipse* para apoyar cada parte de la propuesta. Como prueba de concepto, se ha desarrollado un editor gráfico para la especificación de los requisitos de la aplicación Web (Sección 1.5.1), así mismo, se implementaron un conjunto de transformaciones modelo a modelo para proveer un alto grado de automatización en la generación de los modelos conceptuales de la aplicación Web (Sección 1.5.2).

Finalmente, gracias a la capacidad y mecanismos de extensión de la plataforma *Eclipse*, se desarrolló un *plugin* para asistir al diseñador Web respecto a la toma de decisiones de diseño en base a los requisitos no-funcionales expresados en la especificación de los requisitos (Sección 1.5.4). Por medio del *plugin*, el diseñador de la aplicación Web podrá analizar distintas alternativas de diseño, es decir, podrá analizar y seleccionar los requisitos funcionales a implementar en los modelos conceptuales de la aplicación Web considerando la prioridad establecida sobre los requisitos no-funcionales.

1.6.2. Limitaciones y Trabajo Futuro

En el transcurso de la investigación no fue posible cubrir todos los aspectos del proceso de ingeniería de *software* (análisis, especificación, validación, gestión, etc.). Por tal motivo, la investigación fue delimitada en cuestiones como:

- La herramienta WebREd permite la especificación de los requisitos de la aplicación Web en un ambiente gráfico, pero necesita un proceso de re-ingeniería para mejorar la interfaz gráfica de usuario.
- En lo que respecta al proceso de validación de los requisitos, solo ha sido considerada la trazabilidad de los requisitos, dejando de lado a los casos de prueba.
- En lo que se refiere a la gestión de requisitos, la propuesta ofrece soporte para el análisis de impacto, pero se carece de un control de versiones, propagación de cambios y herramientas para la gestión de la configuración.
- La propuesta no está integrada en una sola herramienta, es decir, la constituyen un conjunto de *plugins* de la plataforma Eclipse, por lo tanto resulta difícil su aplicación en casos reales, con lo cual resalta la necesidad de desarrollar una herramienta CASE que integre cada uno de los *plugins* presentados en la Sección 1.5.

A razón de los puntos anteriores, a continuación se listan las áreas de interés para el trabajo futuro de la investigación realizada en la tesis doctoral.

- El marco de modelado orientado a objetivos i^* es poco escalable, por lo que se estudiarán mecanismos que ofrezcan solución a este problema de tal forma que sea posible manejar el crecimiento de los requisitos de manera fluida, es decir, que cuando la especificación de los requisitos crezca el modelo pueda ser comprendido.
- Definir e implementar las transformaciones modelo a modelo para la generación automática de los modelos conceptuales de usuario, personalización y presentación del método Web A-OOH.
- Realizar distintos experimentos para comprobar el funcionamiento del algoritmo de Optimización de Pareto en casos reales.
- Integrar la propuesta presentada en la tesis en un prototipo de herramienta CASE.
- Extender la propuesta para la especificación de requisitos Web hacia las RIAs con la idea de obtener los modelos conceptuales a nivel PIM a partir de la especificación de los requisitos.
- Analizar la distribución de los requisitos funcionales de las RIAs (cliente o servidor) para proveer un algoritmo que permita asistir al diseñador Web en el proceso de distribución de los requisitos funcionales considerando a los requisitos no-funcionales desde la etapa de análisis y especificación de requisitos.

Finalmente, cabe destacar que la tesis doctoral representa la primera propuesta dirigida por modelos en ingeniería Web que aplica un marco de trabajo orientado a objetivos para la etapa de análisis y especificación de requisitos en un contexto MDA y a la vez permite la generación de los modelos conceptuales de la aplicación Web en un proceso con un alto grado de automatización. Además, gracias a la integración con el método Web A-OOH se ha logrado configurar un método de ingeniería Web dirigido por modelos que consume el proceso de desarrollo de la aplicación Web basado en MDA gracias a que el proceso inicia en el modelo independiente de computación (CIM), del cual son generados los modelos independientes de la plataforma (PIM) y culminando con los modelos específicos de la plataforma.

Bibliografía

1. J. A. Aguilar, I. Garrigós, S. Casteleyn, and J.-N. Mazón. Optimizing Non-Functioanl Requirements in Web Applications. In *Proceedings of the 8th International Workshop on Web Information Systems Modeling (WISM) (In press) held in conjunction with ER*, 2011.
2. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Modelos de weaving para trazabilidad de requisitos Web en A-OOH. In *DSDM: Actas del VII Taller sobre Desarrollo de Software Dirigido por Modelos, JISBD, Congreso Español de Informática (CEDI)*, pages 146–155, Valencia, España, 2010. SISTEDES.
3. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Impact Analysis of Goal-Oriented Requirements in Web Engineering. In B. Murgante, O. Gervasi, A. Iglesias, D. Taniar, and B. Apduhan, editors, *Proceedings of the 12th International Conference on Computational Science and Its Applications (ICCSA)*, volume 6786 of *Lecture Notes in Computer Science*, pages 421–436. Springer Berlin / Heidelberg, 2011.
4. J. A. Aguilar, I. Garrigós, J.-N. Mazón, and J. Trujillo. An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering. *Journal of Universal Computer Science (J. UCS)*, 16(17):2475–2494, 2010.
5. J. A. Aguilar, I. Garrigós, J. N. Mazón, and J. Trujillo. Web Engineering approaches for requirement analysis- A Systematic Literature Review. In *Proceedings of the 6th Web Information Systems and Technologies (WEBIST)*, volume 2, pages 187–190, Valencia, Spain, 2010. SciTePress Digital Library.
6. S. Anwer and N. Ikram. Goal-Oriented Requirement Engineering: A Critical Study of Techniques. In *Proceeding of the 13th Asia Pacific Software Engineering Conference (APSEC)*, pages 121 –130, December 2006.
7. R. Arnold and S. Bohner. Impact analysis-Towards a framework for comparison. In *Proceeding of the Software Maintenance (CSM)*, pages 292 –301, sep 1993.
8. ATLAS Transformation Language. `http://www.eclipse.org/m2m/atl/`.
9. M. Barbero, M. Del Fabro, and J. Bézivin. Traceability and provenance issues in global model management. In *Proceeding of the 3rd ECMDA-Traceability Workshop*, 2007.
10. J. Bézivin. On the unification power of models. *Software and Systems Modeling*, 4(2):171–188, 2005.
11. J. Bézivin, F. Büttner, M. Gogolla, F. Jouault, I. Kurtev, and A. Lindow. Model transformations? transformation models! *Model Driven Engineering Languages and Systems*, pages 440–453, 2006.
12. D. Bolchini and J. Mylopoulos. From Task-Oriented to Goal-Oriented Web Requirements Analysis. In *Proceedings of the 4th International Conference on Web Information Systems Engineering (WISE)*, page 166, Washington, DC, USA, 2003. IEEE Computer Society.
13. A. Bozzon, S. Comai, and P. Fraternali. Current Research on the Design of Web 2.0 Applications Based on Model-Driven Approaches. In *Proceedings of the 8th International Conference on Web Engineering (ICWE)*, pages 25–31, 2008.
14. A. Bozzon, S. Comai, P. Fraternali, and G. T. Carughi. Conceptual modeling and code generation for Rich Internet Applications. In *Proceedings of the 6th International Conference on Web Engineering (ICWE)*, page 353, New York, New York, USA, 2006. ACM Press.
15. S. Casteleyn, W. V. Woensel, and G.-J. Houben. A semantics-based aspect-oriented approach to adaptation in Web engineering. In *Proceedings of the 18th Conference on Hypertext and Hypermedia (HT)*, pages 189–198, 2007.
16. S. Ceri, P. Fraternali, and A. Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites. *Journal of Computer and Telecommunications Networking*, 33(1-6):137–157, 2000.
17. Y. Collette and P. Siarry. *Multiobjective optimization: principles and case studies*. Springer Verlag, 2003.
18. O. M. F. De Troyer and C. J. Leune. WSDM: a user centered design method for Web sites. *Comput. Netw. ISDN Syst.*, 30(1-7):85–94, 1998.
19. M. Del Fabro, J. Bézivin, and P. Valduriez. Weaving Models with the Eclipse AMW plugin. In *Proceedings of the Eclipse Modeling Symposium, Eclipse Summit Europe*, volume 2006, 2006.
20. P. Dolog and J. Stage. Designing interaction spaces for Rich Internet Applications with UML. *Journal of Web Engineering*, pages 358–363, 2007.
21. Eclipse. `http://www.eclipse.org/`, 2010.
22. Eclipse Modeling Framework. `http://www.eclipse.org/modeling/emf/`, 2010.
23. M. Escalona, M. Mejías, and J. Torres. Developing systems with NDT & NDT-Tool. In *Proceedings of the 13th International Conference on Information Systems Development: methods and tools, theory and practice*, pages 149–59, 2004.

24. M. J. Escalona and G. Aragón. NDT. A Model-Driven Approach for Web Requirements. *IEEE Transactions on Software Engineering*, 34(3):377–390, 2008.
25. M. J. Escalona and N. Koch. Requirements Engineering for Web Applications - A Comparative Study. *J. Web Eng.*, 2(3):193–212, 2004.
26. I. Garrigós. *A-OOH: Extending Web Application Design with Dynamic Personalization.* PhD thesis, University of Alicante, Spain, 2008.
27. I. Garrigós, J.-N. Mazón, and J. Trujillo. A Requirement Analysis Approach for Using i* in Web Engineering. In *Proceedings of the 9th International Conference on Web Engineering (ICWE)*, pages 151–165, 2009.
28. P. Godfrey, R. Shipley, and J. Gryz. Algorithms and analyses for maximal vector computation. *The VLDB Journal*, 16:5–28, January 2007.
29. R. Goeritzer. Using impact analysis in industry. In *Proceeding of the 33rd International Conference on Software Engineering (ICSE)*, pages 1155–1157. ACM, 2011.
30. J. Gómez, C. Cachero, and O. Pastor. Extending a Conceptual Modelling Approach to Web Application Design. In *Proceedings of the 12th International Conference on Advanced Information Systems Engineering (CAiSE)*, pages 79–93, London, UK, 2000. Springer-Verlag.
31. O. Gotel and A. Finkelstein. An analysis of the requirements traceability problem. In *Proceedings of the 1st International Conference on Requirements Engineering (RE)*, pages 94–101, 1994.
32. Graphical Modeling Framework. `http://www.eclipse.org/modeling/gmp//`, 2010.
33. i* wiki. `http://istar.rwth-aachen.de`.
34. S. E. Institute. *CMMI for Development®: Guidelines for Process Integration and Product Improvement.* Addison-Wesley Professional, 3rd edition, 2011.
35. Java Language. `http://www.java.com`, 2011.
36. F. Jouault and I. Kurtev. On the Architectural Alignment of ATL and QVT. In *Proceedings of the ACM Symposium on Applied Computing*, pages 1188–1195. ACM, 2006.
37. N. Koch. The Expressive Power of UML-based Web Engineering. In *Proceedings of the 1st International Workshop on Web-oriented Software Technology (IWWOST)*, pages 40–41, 2002.
38. N. Koch, A. Knapp, G. Zhang, and H. Baumeister. UML-Based Web Engineering. In *Web Engineering: Modelling and Implementing Web Applications*, Human-Computer Interaction Series, pages 157–191. Springer London, 2008.
39. N. Koch, S. Meliá, N. Moreno, V. Pelechano, F. Sánchez, and J. Vara. Model-driven Web Engineering. *The European Journal for the Informatics Professional*, pages 40–45, 2008.
40. N. Koch, M. Pigerl, G. Zhang, and T. Morozova. Patterns for the Model-Based Development of RIAs. In *Proceedings of the 9th International Conference on Web Engineering (ICWE)*, ICWE, pages 283–291, Berlin, Heidelberg, 2009. Springer-Verlag.
41. N. Koch, G. Zhang, and M. J. E. Cuaresma. Model transformations from requirements to Web system design. In *Proceedings of the International Conference of Web Engineering (ICWE)*, pages 281–288, 2006.
42. H. T. Kung, F. Luccio, and F. P. Preparata. On Finding the Maxima of a Set of Vectors. *J. ACM*, 22:469–476, October 1975.
43. M. Linaje, J. C. Preciado, and F. Sanchez-Figueroa. Engineering Rich Internet Application User Interfaces over Legacy Web Models. *Journal of IEEE Internet Computing*, 11:53–59, 2007.
44. M. Lindvall and K. Sandahl. How well do experienced software developers predict software change? *Journal. Systems and Software*, 43:19–27, October 1998.
45. L. Machado, O. Filho, and J. a. Ribeiro. UWE-R: an extension to a Web engineering methodology for Rich Internet Applications. *Journal of Trans. Info. Sci. and App.*, 6:601–610, April 2009.
46. S. Meliá, J. Gómez, S. Pérez, and O. Díaz. A model-driven development for GWT-based Rich Internet Applications with OOH4RIA. In *Proceeding of the 8th International Conference on Web Engineering (ICWE)*, pages 13–23. IEEE, 2008.
47. Model Driven Architecture. `http://www.omg.org/mda/`.
48. N. Moreno, J. Romero, and A. Vallecillo. An overview of Model-Driven Web Engineering and the MDA. *Web Engineering: Modelling and Implementing Web Applications*, pages 353–382, 2008.
49. B. Nuseibeh and S. M. Easterbrook. Requirements engineering: a roadmap. In *Proceedings of the Internationa Conference on Software Engineering (ICSE)*, pages 35–46, 2000.
50. Object Constraint Language, Version 2.0. `http://www.omg.org/spec/OCL/2.0/`.
51. Object Management Group. `http://www.omg.org`.
52. O. Pastor, S. M. Abrahão, and J. Fons. An Object-Oriented Approach to Automate Web Applications Development. In *Proceedings of the 2nd International Conference on Electronic Commerce and Web Technologies (EC-Web)*, pages 16–28, London, UK, 2001. Springer-Verlag.

53. J. Preciado, M. Linaje, S. Comai, and F. Sanchez-Figueroa. Designing Rich Internet Applications with Web Engineering Methodologies. In *Proceedings of the 9th IEEE International Workshop on Web Site Evolution (WSE)*, pages 23–30. IEEE, Oct. 2007.
54. J. Preciado, M. Linaje, and F. Sanchez-Figueroa. Adapting Web 1.0 User Interfaces to Web 2.0 Multidevice User Interfaces using RUX-Method. *Journal of Universal Computer Science (J.UCS)*, 14(13):2239–2254, jul 2008.
55. J. C. Preciado, M. Linaje, F. Sanchez, and S. Comai. Necessity of methodologies to model Rich Internet Applications. In *Proceedings of the 7th IEEE International Symposium on Web Site Evolution*, pages 7–13, Washington, DC, USA, 2005. IEEE Computer Society.
56. B. Qian, L. Wang, D. xian Huang, W. liang Wang, and X. Wang. An effective hybrid DE-based algorithm for multi-objective flow shop scheduling with limited buffers. *Computers and Operations Research*, 36(1):209 – 233, 2009.
57. R. Quintero, V. Pelechano, O. Pastor, and J. Fons. Aplicación de MDA al Desarrollo de Aplicaciones Web en OOWS. *Jornadas de Ingeniería de Software y Base de Datos (JISBD), VIII*, pages 84–668, 2003.
58. QVT Language. `http://www.omg.org/cgi-bin/doc?ptc/2005-11-01`.
59. D. Schmidt. Guest editor's introduction: Model-driven engineering. *Computer*, 39(2):25–31, 2006.
60. D. Schwabe and G. Rossi. The Object-Oriented Hypermedia Design Model. *Communications of the ACM*, 38(8):45–46, 1995.
61. Software Engineering Institute. `http://www.sei.cmu.edu/`, 2010.
62. I. Sommerville. *Software Engineering*. Addison-Wesley, 6th edition, 2001.
63. F. Szidarovszky, M. Gershon, and L. Duckstein. *Techniques for multiobjective decision making in systems management*. Elsevier, 1986.
64. Unified Modeling Language. `http://www.uml.org`.
65. M. Urbieta, G. Rossi, J. Ginzburg, and D. Schwabe. Designing the Interface of Rich Internet Applications. In V. A. F. Almeida and R. A. Baeza-Yates, editors, *5th Latin American Web Congress (LA-Web)*, pages 144–153. IEEE Computer Society, 2007.
66. P. Valderas and V. Pelechano. A Survey of Requirements Specification in Model-Driven Development of Web Applications. *ACM Trans. Web*, 5:10:1–10:51, May 2011.
67. F. Valverde. *OOWS 2.0: Un Método de Ingeniería Web Dirigido por Modelos para la Producción de Aplicaciones Web 2.0*. PhD thesis, Universidad Politécnica de Valencia, 2010.
68. A. Van Lamsweerde. Goal-Oriented Requirements Engineering: A Guided Tour. In *Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE)*, RE '01, pages 249–, Washington, DC, USA, 2001. IEEE Computer Society.
69. J. Wright. A Modelling Language for Interactive Web Applications. In *Proceedings of the 24th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 689–692. IEEE, 2010.
70. J. M. Wright and J. B. Dietrich. Requirements for rich internet application design methodologies. In *Proceedings of the 9th international conference on Web Information Systems Engineering*, WISE '08, pages 106–119, Berlin, Heidelberg, 2008. Springer-Verlag.
71. Xpand Language. `http://wiki.eclipse.org/Xpand`, 2011.
72. S. Yoo and M. Harman. Using hybrid algorithm for Pareto efficient multi-objective test suite minimisation. *Journal of Systems and Software*, 83(4):689–701, 2010.
73. E. Yu. *Modelling Strategic Relationships for Process Reenginering*. PhD thesis, University of Toronto, Canada, 1995.
74. E. Yu. Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering. In *Journal of Requirements Engineering*, pages 226–235, 1997.

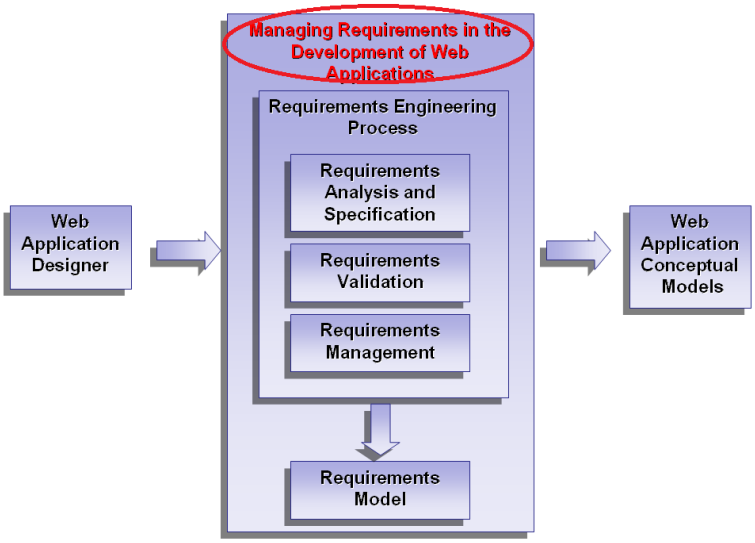
Parte II

Tesis Doctoral como Compendio de Artículos

2

Web Engineering Approaches for Requirements Analysis - A Systematic Literature Review

El contenido del capítulo corresponde con el artículo: ***J.A. Aguilar, I. Garrigós, J.-N. Mazón, J. Trujillo. Web Engineering Approaches for Requirements Analysis - A Systematic Literature Review. 6th Web Information Systems and Technologies (WEBIST 2010), Vol. 2, pp. 187-190, 2010.***



El capítulo presenta una revisión sistemática de la literatura con el fin de obtener el estado de la cuestión en lo referente a métodos para la especificación, análisis y modelado de requisitos en ingeniería Web así como las herramientas de soporte ofrecidas por cada uno de los métodos considerados. Los resultados obtenidos muestran, entre otras cosas, que gran parte de las metodologías Web no ofrecen un soporte integral en la etapa de análisis y especificación de requisitos.

WEB ENGINEERING APPROACHES FOR REQUIREMENT ANALYSIS

A Systematic Literature Review

Jose Alfonso Aguilar, Irene Garrigós, Jose-Norberto Mazón and Juan Trujillo
Lucentia Research Group, University of Alicante, Spain
jaac12@alu.ua.es, igarrigos@dlsi.ua.es, jnmazon@dlsi.ua.es, jtrujillo@dlsi.ua.es

Keywords: Requirement Analysis, Web Engineering, Systematic Review.

Abstract: Web engineering software development is facing continuous changes in technology implementation. This involves analysts, developers and designers to provide extra effort in the design and maintenance of Web applications in order to adapt them to changes in requirements and implementation technologies. In this paper, a systematic review is presented in order to obtain, in a formal way, the current state-of-the-art about approaches for modeling, analysis and specification of Web engineering requirements, supported with a formal and well defined strategy.

1 INTRODUCTION

One of the most important factors of success in the software development is the elicitation, management, and analysis of requirements. This is especially true in Web engineering due to the heterogeneous audience of the Web, which may lead to websites difficult to comprehend by visitors and complex to maintain by designers (Garrigós, Mazón et al. 2009). Importantly, Web applications have certain characteristics that make them different from traditional software or information systems such as the amount of information they offer (content), the access to the different scenarios where they offer this information (navigational) and how providing this information to the user or the groups of users (functionality) of the website. These unique characteristics of Web applications enforced new Web engineering methodologies to cope with those new requirements and Web developers need to adopt it.

The goal of this work is to make a comprehensive review and synthesis of the current state-of-the-art in the literature related to the specification of Web engineering requirements with the purpose of showing gaps in current research and shed light on potential future research lines. In this paper, we expect to study methods and techniques that propose the specification of Web requirements and bring some kind of support to generate several models of the website from the requirements.
The main objective of this work has been achieved through a systematic review using the approach of Kitchenham (Kitchenham 2004) and taking into account Mark Staples experiences (Staples and Niazi 2007).

2 BACKGROUND

Web Engineering (WE) is a branch of Software Engineering (SE) that defines techniques, processes and specific models for the Web environment. WE has a sub-discipline that refers to the phase in application development in which requirements of different stakeholders are gathered and processed, resulting in a requirements specification, this phase is called Requirements Engineering (RE) (Almeida, van Eck et al. 2006).

In this sense, one problem associated with RE is that requirements should be fully completed. To this aim, it is necessary that users and stakeholders can see that it has completed the transformation of their requirements in the final work product and be able to distinguish which one belongs to a certain requirement, this will help determining which requirements will be impacted due to the modification of a work product or vice versa. Furthermore, requirements engineering needs to ensure Requirements Traceability (RT), RT refers to the ability to describe and follow the life of a

requirement, in both a forwards and backwards direction (Gotel and Finkelstein 1994). Forward traceability concerns to follow the requirement to its final implementation. Backward traceability refers to follow the work product to its source requirement (i.e. the associated requirement that originate it).

Recently, the Model-Driven Development (MDD) has become an alternative to solve the problems associated with both SE and WE. MDD was proposed by the OMG (*Object Management Group*). The key idea is that if the software development is guided by models that represent the final software to develop, some benefits will be obtained in some aspects like functionality, interoperability and maintenance.

3 METHODOLOGY

The interest of developing a systematic review starts with the need summarizing existing information about the formal processes for requirements analysis (RA) and specification of Web engineering through modeling and their traceability. This is done with the purpose of establishing more general conclusions revealing the advantages and disadvantages of each approach and to establishing a starting point for future research. To do this, we consider the following research questions:

1. Which Web engineering approaches do exist that cover the requirements analysis in Web engineering and which support requirements traceability?
2. Which are the techniques proposed in Web engineering approaches for requirements analysis and traceability?
3. Which tools have been applied in Web engineering approaches to requirements analysis and which with integrated support for traceability?
4. Which shortcomings have been detected until now in Web engineering approaches in relation to requirements analysis and traceability?

Once we have the research questions, the next step was selecting the digital resources for the search. The resources available for conducting the investigation were: eJournals (ACM, IEEE, Science Direct), Digital library of scientific literature (DBLP Computer Science Bibliography), World Wide Web (Google Scholar). After having selected the search sources, the strings to use for the search were defined. Finally, 13 relevant papers were found in total resulting from the execution of the search strings and search protocol.

Once we have the primary studies, the next step is to extract and summarize the information answering the research questions. As a result, the following Web approaches are presented.

UWE (UML-based Web Engineering. It proposes interviews, questionnaires and checklists as appropriated techniques for the requirements capture (Escalona and Koch 2004). Regarding to the requirements specification, this phase is carried out by means of UML Profiles (Escalona and Koch 2007). This approach does not provide traceability support. With regard to the implementation, UWE has a *plugin* called *MagicUWE* to be used with the CASE tool *MagicDraw* (Busch and Koch 2009). It is worth to mention that in (Kroiss, Koch et al. 2009) the author proposes a set of plugins for *Eclipse* called *UWE4JSF* for the automatic generation of Web applications in JSF (*JavaServer Faces*) derived from UWE models.

Navigational Development Techniques **(NDT).** In requirements analysis phase NDT applies use case diagrams and a set of templates for requirements with a textual description (Escalona and Aragon 2008). Traceability is supported through a set of traceability matrices that keeps the relationship between a requirement and their respective artifact that satisfies it (Escalona, Mejías et al. 2004). Requirements analysis phase and traceability is supported by the NDT-Suite tool (*NDT-Profile*, *NDT-Driver-Quality NDT*) using profiles to work with the CASE tool *Enterprise Architect*. In (Escalona and Koch 2007), the authors of NDT and UWE have developed an UML profile for Web requirements called *WebRE*. In combination of both approaches, the conceptual models can be derived from requirements specification; the main drawback of this approach is the lack of tool support for model-to-model transformations.

Web Modeling Language (WebML). This is a method for designing websites that allow high-level modeling (Ceri, Fraternali et al. 2000). The requirements analysis phase is not described in detail, but in (WebML 2009) the author proposes the use of UML (*Unified Modeling Language*) using use case and activity diagrams for its specification. This approach does not have traceability support.

WSDM: Web Site Design Method. The initial two phases of this approach (Mission Statement and User Modeling) are responsible for managing requirements through techniques such as concept maps (of roles and activities) and the data dictionary for the definition of functional and security requirements (De Troyer and Leune 1998). The requirements in this approach are specified in a textual form. The lack of transformations between models and support for traceability as well as the lack of a prototype tool which supports the

Table 1: Summary of methodological approaches.

Approach	Techniques	Traceability Support	Tools		
			Support	Requirements A.	Traceability
NDT	Use cases, textual templates	Yes	NDT-Tool	NDT-Tool	NDT-Suite
WebML	Use cases,activity diagrams	No	WebRatio	No	No
WSDM	Concept maps, data dictionary	No	No	No	No
UWE	Use cases, UML profiles	No	ArgoUWE	MagicUWE ArgoUWE	No
OOWS	Use cases, Task diagrams, FRT	Yes	OlivaNova	OlivaNova OOWS-Suite	AGG TaskTracer
A-OOH	Use cases, i*,UML profiles	No	VisualWade	Eclipse Plugin	No

requirements specification demonstrates the limitations of this approach.

OOWS: An Object-Oriented Approach for Web Solutions Modeling. This approach provides a gathering requirements phase through a series of strategies that implement FRT (*Function Refinement Tree*), use cases and a set of diagrams (tasks, task specification and data description) for navigation requirements. This approach has traceability support (Valderas and Pelechano 2009), this is done through a set of transformations rules defined by graph theory. In regard to tool support, this approach has an environment called *OOWS-Suite* (Valverde, Valderas et al. 2007), which is integrated with the *OlivaNova* tool to provide support for requirements gathering phase. For the traceability tool support, this approach uses two tools, the first one is the open source tool called AGG (*Attributed Graph Grammar System*) and the second is *TaskTracer* developed by the authors to generate traceability reports.

***A-OOH* (*Adaptive Object Oriented Hypermedia*).** It is an extension of the OOH modeling method (*Object Oriented Hypermedia*) addressed to the user. The techniques used in this approach for the requirements specification are the i* framework and UML-Profiles. The requirements model is specified by the designer using i* models, specifically the SR (Strategic Relationship) model and the SD (Strategic Dependency) model. Next, the conceptual models are generated by means of QVT transformations, thus are considering the goals and needs of stakeholders that will meet the expectations of users thereby reducing errors that may appear on the final implementation. The traceability between the models generated by this approach could be derived from the transformations rules. The requirements specification phase in A-OOH has been implemented using the *Eclipse* development platform (Garrigós, Mazón et al. 2009). Next, a brief analysis about the approaches described above is presented, to facilitate this analysis Table 1 is shown.

Table 1 shows a tendency towards the application of UML profiles as a technique used by the approaches studied in this systematic review and the persistence of other one, the use cases. Moreover, although traceability is a very important success factor for software engineering, a common problem in Model-Driven Web development processes is that most of the approaches lack support for traceability (except NDT and OOWS).

Except WSDM, each approach has a tool to support it. In the requirements phase, only NDT, UWE, OOWS and A-OOH have a tool support. In terms of traceability, the two approaches that have implemented traceability have a tool support. It is worth noting that OOWS combines two tools to achieve this support.

In this context, the approaches explored in this systematic review do not consider the real user expectations of the website as well as stakeholders from an early stage of requirements analysis. A-OOH is the exception, as it considers these expectations through the i * framework where requirements are modeled based in the user goals and objectives, thus avoiding the requirement specification in textual form (with the effort involved to do it this way) and the based on tasks (depending in most cases the analyst's experience).

Is worth mentioning the support offered by the approaches WSDM, NDT, UWE and WebML through its website, they offer examples, published papers and their respective tools for everyone who visits their website, except WSDM which only offers the download of published papers due to that is a tool that has proprietary licence. In the particular case of UWE and WebML is necessary to mention that in their website they have guided step by step

examples to study and practice the developing of a Web application using their respective support tool. This confirms that these two approaches are mostly used in academic (university) projects.

Finally, MDD is applied successfully by several Web engineering methods. However, any approach has a fully guided methodology for the analysis of Web engineering requirements. To sum up, current situation of Web engineering approaches presents the following drawbacks: i) lack of traceability support, ii) RA phase without real user goals and needs; and iii) soundless tool support.

To solve the drawbacks mentioned above, we propose the development of a solution at conceptual modeling level using the approach A-OOH. With regard to the first drawback (lack of support for traceability), the solution will be implemented using Weaving Models in model-to-model transformations for traceability support. For the second one (RA phase without real user goals and needs), will be used the i* framework models for the requirements specification, the requirements model is specified by the designer using i* models, specifically the SR (Strategic Relationship) model and the SD (Strategic Dependency) model. Finally, these drawbacks will be supported by an open source tool.

4 CONCLUSIONS

This work presents the results obtained after carrying out a systematic review of literature whose aim was to make a comprehensive review and synthesis of the current state of the art in the literature related to modeling of Web engineering requirements. To do this, a total of 2813 papers published in literature and extracted from the most relevant scientific sources considered, of which 43 were eventually analyzed in depth, in accordance with the systematic review process adopted.

The results of this systematic review shows that Web engineering methods were not designed to address the design through the analysis of Web engineering requirements. Therefore our future work will be to prove the feasibility of applying the technologies previously-mentioned to propose a design approach guided by the analysis and traceability of requirements in Web engineering and supported by an open source tool.

REFERENCES

Almeida, J. P., van Eck, P. A. T. & Iacob, M. E. 2006. Requirements traceability and transformation conformance in model-driven development. *10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06).* Hong Kong: IEEE Computer Society Press.

Busch, M. & Koch, N. 2009. MagicUWE: A CASE Tool Plugin for Modeling Web Applications. *9th International Conference on Web Engineering.* San Sebastian, Spain: Springer.

Ceri, S., Fraternali, P. & Bongio, A. 2000. Web Modeling Language (WebML): a modeling language for designing Web sites. *Computer Networks,* 33**,** 137-157.

De Troyer, O. M. F. & Leune, C. J. 1998. WSDM: a user centered design method for Web sites. *Computer Networks and ISDN Systems,* 30**,** 85-94.

Escalona, M. & Koch, N. 2004. Requirements engineering for Web Applications: a comparative study. *Journal of Web Engineering,* 2**,** 193-212.

Escalona, M., Mejías, M. & Torres, J. Year. Developing systems with NDT & NDT-Tool. *In:* 13th Information System Development, 2004 Lithuania. 149–59.

Escalona, M. J. & Aragon, G. 2008. NDT. A Model-Driven Approach for Web Requirements. *Software Engineering, IEEE Transactions on,* 34**,** 377-390.

Escalona, M. J. & Koch, N. 2007. Metamodeling the Requirements of Web Systems. Lecture Notes in Business Information Processing. Vol. 1. 2007. Pag. 267-282

Garrigós, I., Mazón, J.-N. & Trujillo, J. 2009. A Requirement Analysis Approach for Using i* in Web Engineering. *9th International Conference on Web Engineering.* San Sebastian, Spain: Springer-Verlag.

Gotel, O. C. Z. & Finkelstein, C. W. Year. An analysis of the requirements traceability problem. *In:* Requirements Engineering, 1994., Proceedings of the First International Conference on, 1994. 94-101.

Kitchenham, B. 2004. Procedures for performing systematic reviews. *Keele University and National ICT Australia Ltd***,** 1-28.

Kroiss, C., Koch, N. & Knapp, A. 2009. UWE4JSF: A Model-Driven Generation Approach for Web Applications. *Proceedings of the 9th International Conference on Web Engineering.* San Sebastian, Spain: Springer-Verlag.

Staples, M. & Niazi, M. 2007. Experiences using systematic review guidelines. *J. Syst. Softw.,* 80**,** 1425-1437.

Valderas, P. & Pelechano, V. 2009. Introducing requirements traceability support in model-driven development of web applications. *Information and Software Technologies,* 51**,** 749-768.

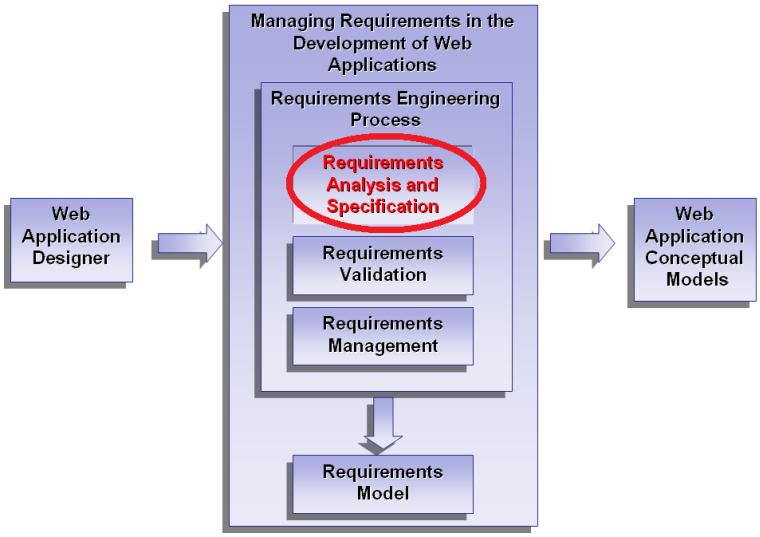
Valverde, F., Valderas, P. & Fons, J. Year. OOWS Suite: Un Entorno de desarrollo para Aplicaciones Web basado en MDA. *In:* 11th International Database Engineering & Applications Symposium, 2007 Banff, Canada.

WebML. 2009. *WebML: The Web Modeling Language*. Retrieved: July 2009, from http://www.webml.org.

3

An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering

El contenido del capítulo corresponde con el artículo: ***J.A. Aguilar, I. Garrigós, J.-N. Mazón, J. Trujillo. An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering. Journal of Universal Computer Science (J.UCS), 16(17): 2475-2494 (2010).***



Se presenta el Capítulo 3, a razón de que los resultados obtenidos en el Capítulo 2 muestran, entre otras cosas, que gran parte de las metodologías Web no ofrecen un soporte integral en la etapa de análisis y especificación de requisitos.

El Capítulo 3 describe una metodología para la gestión de requisitos en ingeniería Web. La metodología está basada en el marco de modelado orientado a objetivos *i** y en MDA (*Model-Driven Architecture*). La propuesta le permite al diseñador de la aplicación Web derivar la estructura de los modelos conceptuales que conforman la aplicación a partir de la especificación de requisitos.

Journal of Universal Computer Science, vol. 16, no. 17 (2010), 2475-2494
submitted: 15/2/10, accepted: 30/8/10, appeared: 1/9/10 © *J.UCS*

An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering

José Alfonso Aguilar, Irene Garrigós, Jose-Norberto Mazón, Juan Trujillo
(Lucentia Research Group, Department of Software and Computing Systems - DLSI, University of Alicante, Alicante, Spain
{ja.aguilar,igarrigos,jnmazon,jtrujillo}@dlsi.ua.es)

Abstract: Web designers usually ignore how to model real user expectations and goals, mainly due to the large and heterogeneous audience of the Web. This fact leads to websites which are difficult to comprehend by visitors and complex to maintain by designers. In order to ameliorate this scenario, an approach for using the i^* modeling framework in Web engineering has been developed in this paper. Furthermore, due to the fact that most of the existing Web engineering approaches do not consider how to derive conceptual models of the Web application from requirements analysis we also propose the use of MDA (*Model Driven Architecture*) in Web engineering for: (i) the definition of the requirements of a Web application in a Computational Independent Model (CIM), (ii) the description of Platform Independent Models (PIMs), and (iii) the definition of a set of QVT (*Query/View/Transformation*) transformations for the derivation of PIMs from requirements specification (CIM), thus to enable the automatic generation of Web applications. Finally, we include a sample of our approach in order to show its applicability and we describe a prototype tool as a proof of concept of our research.
Key Words: MDA, Web engineering, goal-oriented requirements, requirements analysis, model transformations
Category: D.2.1, D.2.2

1 Introduction

Web applications have certain characteristics that make them different from traditional software or information systems such as the amount of information they offer (*content*), the access to the different scenarios where they offer information (*navigation*) and how providing information to the users (*functionality*) of the website. These unique characteristics of Web applications have enforced new Web engineering methodologies to cope with those new requirements and Web developers need to adopt it [Casteleyn et al. 2007, Cachero and Gómez 2002, Casteleyn et al. 2005, Koch 2000, Ceri and Manolescu 2003].

In this context, Web engineering approaches should consider how to gather and process requirements of different stakeholders, resulting in a requirements specification. The term Requirements Engineering (RE) is widely used to indicate that only requirements elicitation is not enough, since requirements have to be processed to resolve conflicts, prioritized, and captured in a consistent requirements specification [Almeida et al. 2006].

However, due to the idiosyncrasy of the audience, traditionally methodologies for Web engineering have not taken into serious consideration the requirement analysis phase. As mentioned above, Web applications have certain characteristics that make them different from traditional software or information systems. Currently, one of the main characteristics of Web applications is that they typically serve large and heterogeneous audience, since respectively (i) everybody can access to the website and (ii) each user has different needs, goals and preferences. Interestingly, this is the opposite situation from the traditional software development where the users are well known. Due to the heterogeneity of the users of a Web application any Web engineering method should consider a requirements analysis phase indicating the users needs and every feature that the Web application must satisfy [Escalona and Koch 2004].

Nevertheless, current effort for requirement analysis in Web engineering is rather focused on the system and the needs of the users are figured out by the designer. This scenario leads us to websites that do not assure the achievement of *real* user requirements and goals, thus producing user disorientation and comprehension problems. Hence, there may appear development and maintenance problems for designers, since costly, time-consuming and rather non-realistic mechanisms (e.g. surveys among visitors) should be developed to improve the already implemented website, thus increasing the initial project budget.

Lately, MDA was established by the OMG (Object Management Group) as architecture for application development. MDA identifies three types of models: the Computational Independent Model (CIM), describes the business logic (requirements), the Platform Independent Model (PIM), which specifies the CIM logic independent of software technology platforms and, finally, the Platform Specific Model (PSM) specifies the model in a specific technology platform. The key idea of this architecture is that if the software development is guided by models that represent the final developed software, some benefits will be obtained in some aspects like functionality, interoperability and maintenance [Brown 2004].

As an alternative to solve the problems associated with RE, specifically in the Web engineering field, in this paper, a complementary viewpoint should be adopted: modeling which are the expectations, intentions and goals of the users when they are browsing the site and determining how they can affect the definition of a suitable Web design.

MDA-based proposals [Meliá and Gómez 2006] have traditionally focused on the definition of transformations from the PIM level to the code, optionally passing through a PSM level. Unfortunately, these proposals usually ignore the CIM level where user requirements can be defined. Bearing these considerations in mind, in this paper we introduce an MDA alignment with the proposal

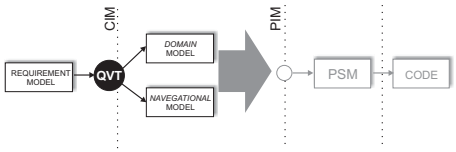


Figure 1: Our MDA approach for goal-oriented requirement analysis in Web engineering.

presented in [Garrigós et al. 2009] to specify requirements in a model that represents a CIM level in MDA. To this aim, we propose to use the *i** modeling framework [Yu 1995, Yu 1997], one of the most valuable approaches for analyzing stakeholders' goals and how the intended system would meet them. For representing the PIM level in our approach we considered two of the models (Domain and Navigational models) proposed by the A-OOH method [Garrigós 2008] as a PIM.

This approach is intended to support an automatic derivation of the Domain and Navigational models (PIMs) from the requirements model (CIM). In order to achieve this, we have defined a set of QVT rules [QVT Specification]. An overview of our proposal can be seen in Figure 1.

The remainder of this paper is structured as follows: Section 2 presents current requirements analysis approaches for Web engineering. Our approach for goal-oriented Web requirement analysis in an MDA context is presented in Sect. 3. Section 4 describes an example based on an *Online Bookstore* to show the applicability of our proposal. Section 5, describes the implementation *framework* of our approach. Finally, in Sect. 6, we present our conclusions and sketch some future work.

2 Related Work

Nowadays few approaches have focused on defining an explicit requirement analysis (RA) phase to model the real user goals and needs in an MDA context. Some of them consider a requirements phase using techniques such as use cases and text templates among others [Aguilar et al. 2010], but as mentioned before, leaving aside the requirements specification from a CIM. Among the approaches, including RA in Web Engineering, we can stress the following:

UWE (*UML-based Web Engineering*) [Koch et al. 2006]. Regarding to the requirements specification, UWE initially used use case diagrams with textual descriptions, currently this phase is carried out by means of UML profiles in combination with a metamodel called WebRE (developed by

the authors of NDT and UWE), more information can be consulted in [Escalona and Koch 2006]. Through the requirements model, UWE can generate content, navigation, presentation and process models through a series of QVT transformations. With regard to the implementation, a plugin called MagicUWE was developed to be used within the CASE tool MagicDraw (http://www.magicdraw.com) [Bush and Koch 2009], which allows UWE to provide a professional CASE tool, although it does not take into account the analysis of requirements.

NDT (*Navigational Development Techniques*) [Escalona and Aragón 2008]. Requirements are specified in NDT by using use case diagrams and textual templates. When a complex application is developed with this approach, it is difficult to maintain due to the use of textual templates for requirements specification; in the words of the authors of the methodology: *the templates are not easy to complete as they require intensive interviews* [Escalona and Koch 2006]. The RA phase and traceability is supported by the NDT-Suite tool (NDT-Profile, NDT-Driver-Quality NDT) by using profiles to work with Enterprise Architect (http://www.sparxsystems.com). By combining NDT, UWE and WebRE, the conceptual models (content, navigational and abstract interface) can be derived from requirements specification [Escalona and Koch 2006]; the main drawback of this approach is the lack of tool support for model-to-model transformations.

WebML (*Web Modeling Language*) [Ceri et al. 2000]. In this method, a distinctive feature is the use of XML (eXtensible Markup Language) to represent the models (data, hypertext, presentation and customization) generated in each stage of development. The RA phase is not described in detail, in [WebML], but authors propose the use of UML use case and activity diagrams for requirements specification.

WSDM (*Web Site Design Method*) [De Troyer and Leune 1998]. Requirements management is carried out through techniques such as concept maps (of roles and activities) and the data dictionary for the definition of functional and security requirements. The form in which this approach considers the requirements can cause precision errors when they are specified, because this is done in a textual form. The lack of transformations between models as well as the lack of a prototype tool which supports the requirements specification demonstrates the limitations of this approach.

OOHDM (*Object Oriented Hypermedia Design Model*) [Schwabe 1998]. This approach is based on the Object Oriented paradigm. The requirements specification is divided in (i) identification of roles, (ii) specification of scenarios, (iii) specification of Use Cases, (iv) specification of User Interaction Diagrams and (v) validation of User Interaction Diagrams and Use Cases. This approach defines guidelines to define conceptual and navigational schemas by means of rules described in natural language, this rules indicate how the conceptual and

navigational schema can be defined from User Interaction Diagrams. OOHDM does not have a tool support for the RA phase. Furthermore, this approach does not support the definition of conceptual models from requirements specification.

OOWS (*Object-Oriented Approach for Web Solutions*) [Fons et al. 2003]. The RA phase is carried out through a set of strategies (i) FRT (Function Refinement Tree), (ii) Use Cases and (iii) Tasks, Task Specification and Data Description Diagrams for navigation requirements. The authors are currently working on a technique for the specification of requirements through ontologies. The task analysis is a technique that in most cases where it is implemented is time-consuming, complex and depends largely on the experience of the analyst for its correct implementation. Moreover, according to [Bolchini and Mylopoulos 2003], the user needs are not necessarily well defined within his own mind as to be defined as tasks. In regard to tool support, this approach has an environment called OOWS-Suite [Valverde et al. 2007], which is integrated with the OlivaNova tool (http://www.integranova.com) to provide support for requirements gathering phase.

Table 1: Summary of methodological approaches.

Approach	Techniques	Tool support	Tool support for RA
NDT	Use cases, textual templates	NDT-Tool	NDT-Tool
WebML	Use cases, activity diagrams	WebRatio	No
WSDM	Concept maps, data dictionary	No	No
UWE	Use cases, UML profiles	ArgoUWE	MagicUWE, ArgoUWE
OOWS	Use cases, task diagrams, FRT	OlivaNova	OlivaNova, OOWS-Suite
OOHDM	Use cases, User Interaction Diagrams, Conceptual maps	OOHDM-WEB	No
A-OOH	Use cases, i*, UML profiles	VisualWade	Eclipse plugin

Table 1 shows a summary of the reviewed approaches. It is shown a tendency towards the application of UML profiles as a technique for requirements specification, and the persistence of other one, the use cases. Except WSDM, each approach has a tool to support it. In the requirements phase, only NDT, UWE, OOWS and A-OOH have a tool support.

Is worth mentioning the support offered by the approaches WSDM, NDT, UWE and WebML through its website, they offer examples, published papers and their respective tools for everyone who visits their website, except WSDM which only offers the download of published papers due to that is a tool that has proprietary license. In the particular case of UWE and WebML is necessary to mention that in their website they have guided step by step examples to study and practice the developing of a Web application using their respective support tool. These two approaches are mostly used in the academic enviroment.

Generating conceptual models from requirements (CIM) is an important

issue to bridge the gap between requirements and Web design, i.e., to ensure that the Web application will satisfy real user needs and expectations. There are two approaches to the authors knowledge that support this in some way: OOWS provides automatic generation of (only) navigation models from the tasks description by means of graph transformation rules. NDT [Koch et al. 2006] defines a requirement metamodel and allows to transform the requirements model into a content and navigational model by means of QVT rules. Our approach resembles NDT since we have also adopted QVT in order to obtain design artifacts from Web requirements, but we have kept the benefits of the i^* framework by means of the defined profiles, furthermore, we consider a formal mechanisms (MDA) to achieve the automatic derivation of conceptual models (PIMs) from requirements model (CIM).

However, OOWS and NDT present some of the following drawbacks: (i) they do not take into consideration a complete taxonomy of requirements which is suitable in Web applications, or (ii) they consider non-functional requirements in an isolated manner, or (iii) they mainly focus on design aspects of the intended Web system without paying enough attention to Web requirements. Furthermore, none of them perform the analysis of the users needs. To the best of our knowledge, the only approaches that use goal oriented techniques have been presented in [Molina et al. 2008, Bolchini and Paolini 2004]. They propose complete taxonomies of requirements for the Web by using the i^* notation to represent them. Unfortunately, they do not benefit from every i^* feature, since they only use a metamodel that has some of its concepts. For example, means-end, decomposition or contribution links from i^* are not specified in the approach presented in [Bolchini and Paolini 2004].

Modeling users requirements by using i^* in a CIM allows us to ensure that the Web application satisfies real user needs and goals and the user is not overwhelmed with not needed or not expected functionalities, at the same time that required functionalities are not missed. In the next section, our proposal for goal-oriented requirements analysis using A-OOH method in an MDA context is presented.

3 MDA artifacts for modeling requirements in Web engineering

In this section, we present a proposal which provides a way of specifying requirements by using i^* and A-OOH *(Adaptive Object Oriented Hypermedia method)* [Garrigós 2008] in an MDA context. A-OOH is the extension of the OO-H modeling method [Cachero and Gómez 2002], which includes the definition of adaptation strategies. This approach has also been extended with UML-profiles so all the conceptual models are UML-compliant (see Sect. 3.1). It is worth

noting that we use A-OOH for demonstration purposes but the proposal could be applied to any Web modeling method. The automatic generation of both the Domain and Navegational conceptual models from the specified requirements model is also supported (see Sect. 3.2) by means of QVT rules. Designers only have to focus on specifying the requirements, and refining the generated conceptual models if needed.

3.1 Web requirements specification as a CIM

The development of Web applications involves different kind of stakeholders with different needs and goals. Interestingly, these stakeholders depend on each other to achieve their goals, perform several tasks or obtain some resource, i.e. *the Web administrator relies on new clients for obtaining data in order to create new accounts*. In the requirements engineering community, goal-oriented techniques, such as the *i** framework [Yu 1997, Yu 1995], are used in order to explicitly analyze and model these relationships among multiple stakeholders (actors in the *i** notation). The *i** modeling framework has been proven useful for representing (i) intentions of the stakeholders, i.e. their motivations and goals, (ii) dependencies between stakeholders to achieve their goals, and (ii) the (positive or negative) effects of these goals on each other in order to be able to select alternative designs for the system, thus maximizing goals' fulfillment. In this work, specification of requirements is carried out by means of *i** framework and represents the CIM level of MDA.

Next, we briefly describe an excerpt of the *i** framework which is relevant for the present work. For a further explanation, we refer the reader to [Yu 1997, Yu 1995]. The *i** framework consists of two models: the strategic dependency (SD) model to describe the dependency relationships (represented as ) among various actors in an organizational context, and the strategic rationale (SR) model, used to describe actor interests and concerns and how they might be addressed. The SR model (represented as ) provides a detailed way of modeling internal intentional elements and relationships of each actor (). Intentional elements are goals (), tasks (), resources () and softgoals (). Intentional relationships are means-end links () representing alternative ways for fulfilling goals; task-decomposition links () representing the necessary elements for a task to be performed; or contribution links () in order to model how an intentional element contributes to the satisfaction or fulfillment of a softgoal. A sample application of the *i** modeling framework is shown in Fig. 2, which represents the SR model of our case study (see Sect. 4) for the client stakeholder. It is our requirements model and corresponds to our CIM. The main goal of the client is to *"buy books"*. In order to do this, the client should *"choose a book to buy"* and *"provide his/her own data"*. The task *"choose a book to buy"* should be decomposed in several subtasks: *"consult books", "search for a specific*

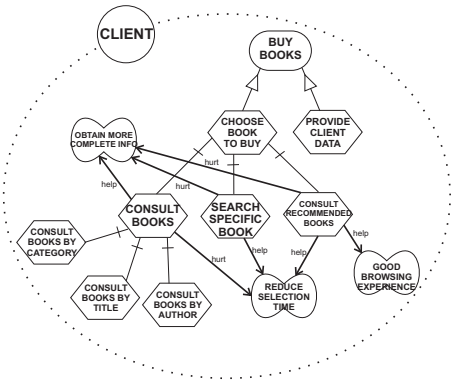


Figure 2: Modeling the client in an SR model.

book", "consult recommended books". These tasks can have positive or negative effects on some important softgoals. For example, while *"consult books"* helps to satisfy the softgoal *"obtain more complete information"*, it hurts the softgoal *"reduce selection time"*. Moreover, *"consult books"* can be further decomposed according to the way in which the book data is consulted.

Although *i** provides good mechanisms to model actors and relationships between them, it needs to be adapted to the Web engineering domain to reflect special Web requirements that are not taken into account in traditional requirement analysis approaches, thus being able to assure the transition of requirements to conceptual models to be used in Web design. Web functional requirements are related to three main features of Web applications [Escalona and Koch 2004] (besides of the non-functional requirements): navigational structure, user interface and personalization capability. Furthermore, the required data structures of the website should be specified as well as the required (internal) functionality provided by the system. To cover these features, in this paper, we use the taxonomy of Web requirements presented in [Escalona and Koch 2004] *(Content, Navigational, Service, Layout, Personalization and Non-functional)*. The *Content Requirements* define the content that the website presents to its users. Some examples might be: *"book information"* or *"product categories"*. The *Service Requirements* refer to the internal functionality the system should provide to its users. For instance: *"register a new client"*, *"add product"*, etc. The *Personalization Requirements* allow the designer to specify the desired personalization actions to be performed in the final website (e.g. show recommendations based on interest, adapt font for visual impaired users, etc.). Finally, the *Navigational Requirements* are useful to define the navigational paths available for the existing users. Some examples are: *"consult products by category"*, *"consult shopping*

cart", etc. Although in the examples provided throughout this paper all these kind of requirements are used, it is worth to point out that, we focus on how to include *Content, Service, and Navigational requirements* in our MDA approach.

Once this classification has been adopted, the i^* framework needs to be adapted. As the considered Web engineering approach (A-OOH) is UML-compliant, we have used the extension mechanisms of UML to (i) define a profile for using i^* within UML; and (ii) extend this profile in order to adapt i^* to specific Web domain terminology. Therefore, new stereotypes have been added according to the different kind of Web requirements [Garrigós et al. 2009]: *Navigational*, *Service*, *Personalization* and *Layout* stereotypes extend the *Task* stereotype and *Content* stereotype extends the *Resource* stereotype. It is worth noting that non-functional requirements can be modeled by directly using the softgoal stereotype.

Finally, several guidelines should be provided in order to support the designer in defining i^* models for Web domain, i. e. (i) Determine the kind of users for the intended Web and model them as actors. The website is also considered as an actor. Dependencies among these actors must be modeled in an SD model, (ii) Define actors' intentions by using i^* techniques in an SR model [i Star Wiki]: modeling goals, softgoals, tasks and resources, and the relationships between them and (iii) Annotate tasks as navigational, service, personalization or layout requirements. Also, annotate resources as content requirements. It is worth noting that goals and softgoals should not be annotated.

3.2 Definition of conceptual models as PIM

Once the requirements have been defined they can be used to derive the conceptual models for the website. In our work we only focus on the derivation of the Domain and Navigational models. Once these models are derived the designer has only to refine them, avoiding the task of having to create them from scratch.

Since the i^* framework does not well support the definition of other design artifacts by its own, domain-oriented mechanisms should be considered to perform this task [Estrada et al. 2006]. In our approach we have defined a UML profile in which the new stereotypes (presented in the previous subsection) allow us to prepare models for this definition phase. We have detected several i^* patterns [Strohmaier et al. 2008] in order to define a set of QVT transformation rules to map elements from the SR metamodel to their counterparts in the A-OOH metamodel.

The metamodel used for the A-OOH Domain model is the UML metamodel. It describes the objects, attributes, and relationships necessary to represent the concepts of UML within a software application. The modeling notation necessary for the A-OOH Domain model are the UML Class Diagrams. The purpose of a class diagram is to depict the classes within a model. In an object oriented

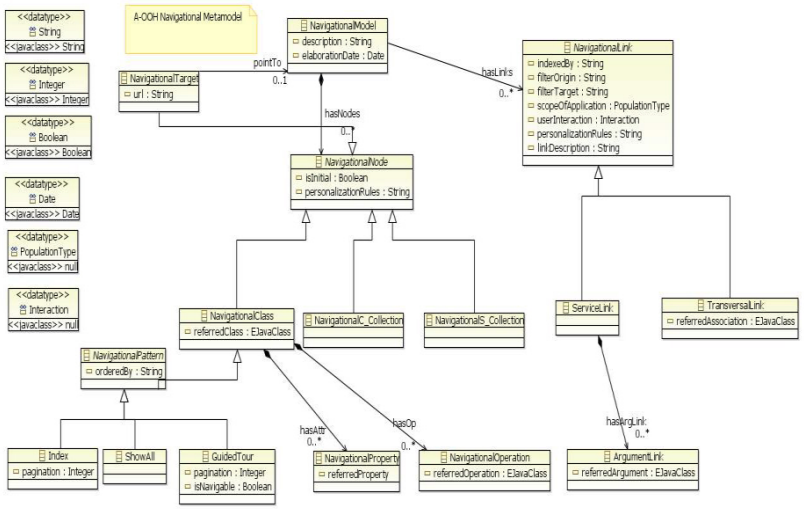
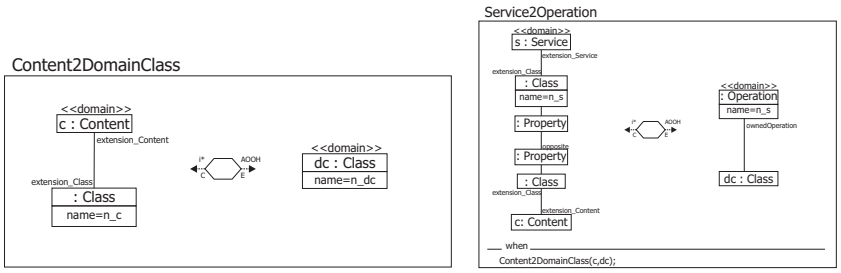


Figure 3: Excerpt of the A-OOH Navigational metamodel.

application, classes have attributes (member variables), operations (member functions) and relationships with other classes.

On the other hand, the Navigational metamodel used for the A-OOH Navigational model is shown in Fig. 3.2. The main elements of the metamodel are the Navigational Node and the Navigational Link. The Navigational Node represents restricted views of the domain concepts and their relationships indicating the navigation paths the user can follow in the final website. Each Node has associated an owner Root Concept from the Domain model. There are three different types of Navigational nodes, the *Navigational Classes* (domain classes enriched with attributes and methods which visibility has been constrained depending on the access permissions of the user and navigational requirements), the *Navigational Targets* (model elements which collaborate in the fulfillment of every navigation requirement of the user) and the *Collections* (wich provide the user new ways of accessing the information, i. e. menu). On the other hand, the Navigational Links define the navigational paths that the user can follow through the system. The A-OOH navigational metamodel defines two main types of links, the *T-Links (Transversal Links)* (defined between two navigational nodes) and the *S-Links (Service Links)* (navigation is performed to activate an operation which modifies the business logic).

After analyzing and modeling the requirements of the website according to the guidelines presented in the previous subsection, the Domain model (DM) and Navigational model (NM) are generated from the specified requirements (CIM).



(a) How to derive the DM classes.

(b) How to obtain the operations for the DM classes.

Figure 4: QVT rules for the Domain Model.

3.3 Deriving the Domain model

The A-OOH DM is expressed as a UML-compliant class diagram. It encapsulates the structure and functionality required of the relevant concepts of the application and reflects the static part of the system. The main modeling elements of a class diagram are the classes (with their attributes and operations) and their relationships. We have defined a set of QVT rules to derive the DM from requirements model (see Figs. 4-5). In this transformation rules, the source model correspond to our CIM for Web requirements specification in i^* while the DM as a representation model for the Web application domain is the PIM level, both of them in a MDA context.

Content2DomainClass. By using this transformation rule, each content requirement is detected and derived into one class of the DM (see Fig. 4(a)).

Service2Operation. Detects a service requirement with an attached content requirement in the SR model, each service requirement is transformed into one operation of the corresponding class (represented by the content requirement), as shown in Fig. 4(b).

Navigation2Relationship. To generate the associations in the DM we have to detect a navigational root requirement (i.e. task) in the SR model which can contain one or more navigational requirements attached. Each of the navigational requirement can have attached a resource (i.e. content requirement)(see Fig. 5).

Once the DM skeleton has been obtained it is left to the designer to refine it, who will also have to specify the most relevant attributes of the classes, identify the cardinalities and define (if existing) the hierarchical relationships. After the preliminary DM is created, a skeleton of the NM is also derived from the specified requirements. This diagram enriches the DM with navigation and interaction features.

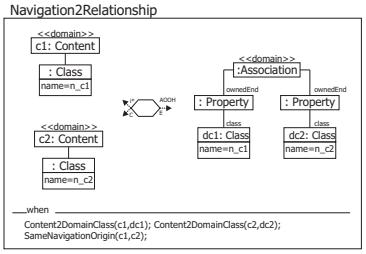


Figure 5: QVT rule for associations between DM classes.

3.4 Deriving the Navigational model

The A-OOH Navigational model corresponds to a PIM in the MDA context. To derive the NM we take into account the content, service, navigation and personalization requirements. We also take into consideration a set of QVT transformation rules, the QVT rules defined to obtain the Navigational model are described (see Fig. 6). In this set of relationships the source model is our CIM for Web requirements specified in *i**, while the NM of the Web application represents our PIM.

Nav&Pers2NavClass. By using this rule, a "home" navigational class is added to the model. From each navigational and personalization requirement with an associated content requirement a navigational class (NC) is derived. From the "home" NC a transversal link is added to each of the generated NCs.

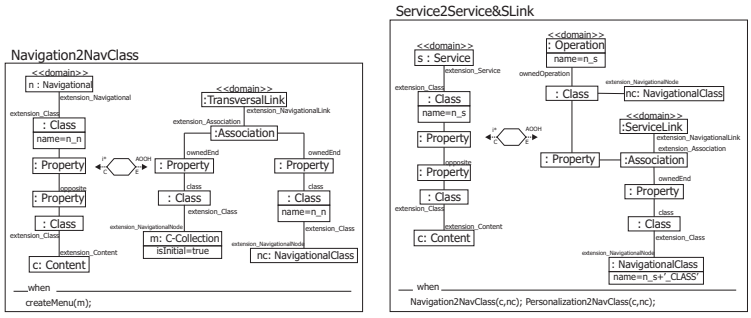
Navigation2TLink. This rule checks the navigational requirements associated with one or more content requirements in the SR model , if it is detected, then a transversal link is added from the NC that represents the root navigational requirement to each of the NCs representing the associated navigational requirements.

Service2Service&SLink. Finally, if a service requirement associated with a content requirement is found, then an operation to the class representing the resource is added and service link is created from each of the operations, with a target navigational class which shows the termination of the service execution.

For the NM and DM, there are some features that can not be automatically derived. So, the designer should refine these models in order to add some minor elements to obtain a more efficient implementation.

4 Sample Application of our Approach

In this section, we provide an example of our approach based on a company that sells books on-line. In this case study, a company would like to manage book



(a) How to obtain navigational classes for NM.

(b) How to obtain the service links for the navigational classes.

Figure 6: QVT rules for the Navigational Model.

sales via an online bookstore, thus attracting as many clients as possible. Also there is an administrator of the Web to manage clients.

4.1 Requirements specification

As mentioned above, the requirements specifications have been implemented by means of our *i** framework for Web requirements (see [Garrigós et al. 2009]). For the *Online Bookstore*, three actors are detected that depend on each other, namely *"Client"*, *"Administrator"*, and *"Online Bookstore"*. A client depends on the online bookstore in order to *"choose a book to buy"*. The administrator needs to use the online bookstore to *"manage clients"*, while the *"client data"* is provided by the client. These dependencies are modeled by an SD model (see Fig. 7). Once the actors have been modeled in an SD model, their intentions are specified in SR models.

The SR model for the client actor was previously explained in Sect. 3.1. The SR model of the online bookstore is shown in Fig. 7. The main goal of this actor is to *"manage book sales"*. To fulfill this goal the SR model specifies that two tasks should be performed: *"books should be sold online"* and *"clients should be managed"*. We can see in the SR model that the first of the tasks affects positively the softgoal *"attract more users"*. Moreover, to complete this task four subtasks should be obtained: *"provide book info"* (which is a navigational requirement), *"provide recommended books"* (which is a personalization requirement), *"search engine for books"*, and *"provide a shopping cart"*. We can observe that some of these tasks affect positively or negatively to the non-functional requirement *"easy to maintain": "Provide book information"* is easy to maintain, unlike *"provide recommended books"* and *"use a search*

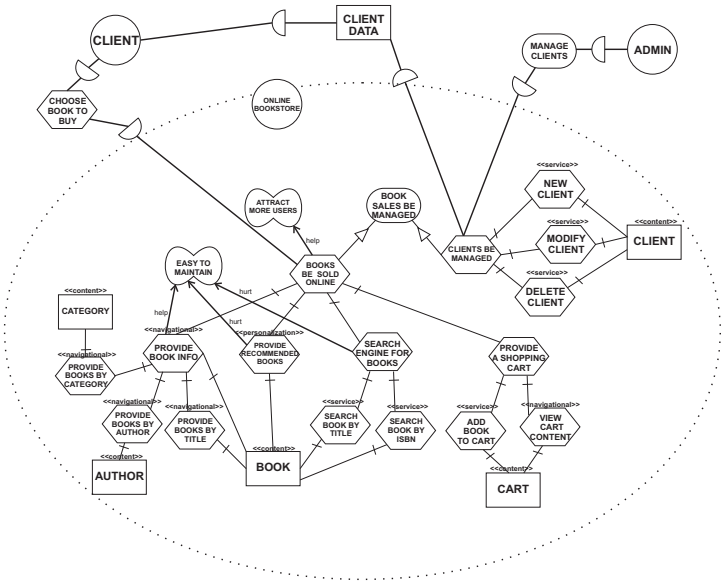
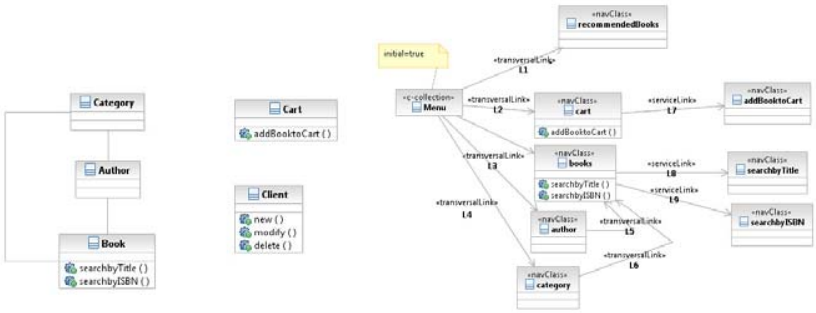


Figure 7: Modeling the online bookstore in an SR model and the SD model.

engine for books". The navigational requirement *"provide book information"* can be decomposed into several navigational requirements according to the criteria used to sort the data. These data is specified by means of content requirements: *"book"*, *"author"* and *"category"*. The personalization requirement *"provide recommended books"* is related to the content requirement *"book"* because it needs the book information to be fulfilled. The task *"search engine for books"* is decomposed into a couple of service requirements: *"search book by title"* and *"search book by ISBN"*, which are also related to the content requirement *"book"*. In the same way, the task *"provide a shopping cart"* is decomposed into two service requirements: *"add book to cart"* and *"view cart content"*. These service requirements are related to the content requirement *"cart"*. Finally, the task *"clients be managed"* is decomposed into three service requirements: *"new client"*, *"modify client"* and *"delete client"*, which are related to the content requirement *"client"*.

4.2 Domain Model

In Fig. 8(a) we can see the derived DM (PIM) from the specified requirements (CIM) as a result of the QVT rules execution. To derive the DM we take into account the content and service requirements as well as the existence of



(a) The Domain Model. (b) The Navigational Model.

Figure 8: The Domain and Navigational Models.

associations between navigation and service requirements. In this case we can see that five domain classes are created by applying the *Content2DomainClass* transformation rule: one class is generated for each content requirement specified in the SR model. Moreover, we detect three service requirements, so operations are added to the classes *client, cart* and *book* by executing the *Service2Operation* rule. Finally we detect that the *Provide Book Info* requirement (navigational) is associated with a content requirement , in this case the rule *Navigation2Relationship* adds associations among all the resources found in this association. The generated DM is shown in Fig. 8(a).

4.3 Navigational Model

In the case of the Navigational model (PIM) (see Sect. 8(b)), the rule *Nav&Pers2NavClass* is performed adding a home page with a collection of links (i.e. menu). Afterwards, one NC is created for each navigational and personalization requirement with an attached resource, in this case we have five NC created from navigational and personalization requirements. From the menu, a transversal link to each of the created NCs is added (L1 to L4).

The next step is checking the navigational and service requirements. In this example, we find a navigational requirement applying the *Navigation2TLink* it implies creating a transversal link from the NCs created by the associated navigational requirements, to the NC that is represented by the root navigational requirement. In this case two links are added: L5 and L6.

Finally, as we are referring to the website stakeholder, we find three service requirements from which the operations of the NCs books and cart are added and the service links L7, L8 and L9 are created with an associated target NC by applying the *Service2Service&SLink* (see Fig. 8(b)).

5 Implementation Framework

In this section we introduce the implementation of our approach for goal-oriented requirements analysis in Web engineering. To this aim, we have combined a set of technologies:

- Eclipse. The main feature of this open source IDE (*Integrated Development Environment*) [Eclipse] is that it is primarily a software platform used to create integrated development environments.
- EMF *Eclipse Modeling Framework*. The EMF project [EMF] is a modeling framework and code generation facility for building tools and other applications based on a structured data model. Facilities for creating metamodels and models are provided by the metametamodel Ecore.
- UML profiles. A profile in the Unified Modeling Language (UML) provides a generic extension mechanism for customizing UML models for particular domains and platforms.

After the introduction of the tools used in implementing this proposal, we explain the implementation of the metamodels for Web requirements (*i** for Web), domain (UML class diagram) and navigation (A-OOH) in the Eclipse framework by using the Ecore metametamodel.

Implementation of Web requirements metamodel. The requirements metamodel consists of a UML profile which incorporates a number of taxonomic features that enable Web requirements specification. With the implementation of this metamodel has been possible to implement the *i** framework in Web to model the needs and expectations of the stakeholders of the Web application. The taxonomy of requirements presented in 3.1 have been incorporated as stereotyped *i** elements in our UML profile. Figure 9 shows a screenshot where the reader can see the implementation of this metamodel in Eclipse IDE.

Implementation of A-OOH Domain metamodel. The Domain model in A-OOH is represented by a UML class diagram, for this reason we have implemented the UML 2.0 metamodel by using the Eclipse Ecore metametamodel to represent only the elements necessary to model a UML class diagram.

Implementation of A-OOH navigational metamodel. The A-OOH Navigational metamodel represents the key to the derivation of the navigational model. The implementation was developed by using the EMF Ecore metametamodel. In this way, the A-OOH navigational metamodel has been modeled via a class diagram. Figure 10 shows a screenshot of the Navigational metamodel implemented using Eclipse and EMF.

Implementation of the QVT transformation rules. By implementing these rules the Domain and Navigational models (PIMs) are generated from the

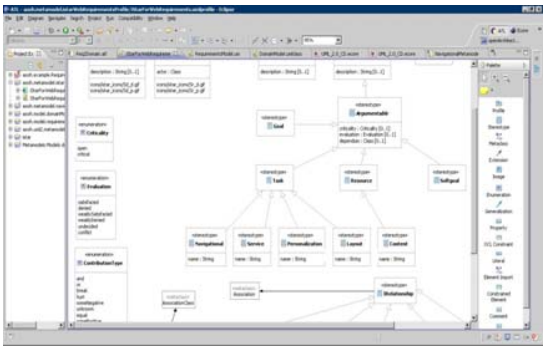


Figure 9: UML profile for Web requirements implemented in Eclipse.

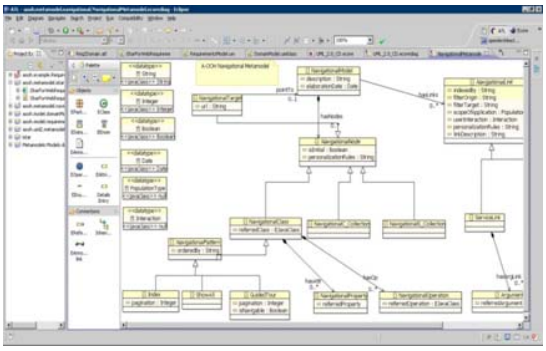


Figure 10: A-OOH Navigational metamodel implemented in Eclipse.

requirements model (CIM). In order to take advantage of every feature of a model-driven engineering approach as MDA, transformations should be viewed from a modeling perspective as transformation models. Following this perspective, throughout the paper, QVT has been used as a metamodel for formalizing transformations between models by abstracting them as models, thus ameliorating the understandability of the transformation process. However, once the transformations have been modeled, they have to be implemented. To this aim, the QVT transformation rules presented before have been implemented by using the facilities provided by Eclipse framework and EMF.

Finally, a plugin that supports both of the defined profiles has been developed (for CIM and PIM). This new plugin implements several graphical and textual editors (Fig. 11 shows an overview of the tool). The palette for drawing the different elements of i^* can be seen on the right-hand side of this figure.

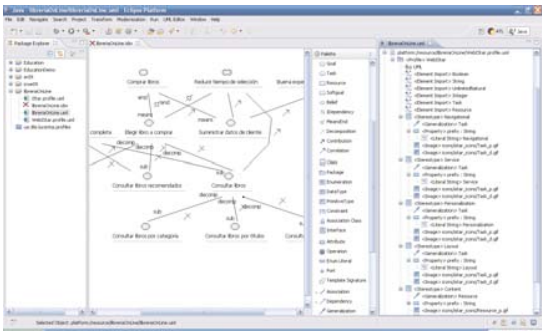


Figure 11: Screen capture of our prototype.

6 Conclusions and Future Work

Websites require special techniques for requirement analysis in order to reflect, from early stages of the development, specific needs, goals, interests and preferences of each user or user type. However, Web engineering field does not pay the attention needed to this issue since most of the approaches lack a fully guided methodology for Web requirements analysis. We have presented a goal oriented approach on the basis of the i^* framework to specify Web requirements. It allows the designer to make decisions from the very beginning of the development phase that would affect the structure of the envision website in order to satisfy users.

Moreover, the following guidelines are provided to the designer to properly define i^* models for the Web domain: (i) discovering the intentional actors (i.e. Web users and the Web application) and their dependencies in an SD model, (ii) discovering their intentional elements, thus defining SR models for each one, and (iii) annotating intentional elements with Web concepts. We can use this model to check the current website or to make the appropriate decision to build a new one. Moreover, we have defined a set of transformation rules in order to obtain the Domain and Navigational conceptual models (PIMs) from requirements specification (CIM). Although this approach is presented in the context of the A-OOH modeling method it can be applied to any Web modeling approach.

Our short-term future work consists of completing the transformation rules in order to obtain the rest of the A-OOH models (i.e. presentation and personalization models).

Acknowledgements

This work has been partially supported by the MANTRA project (GRE09-17) from the University of Alicante, and by the MESOLAP (TIN2010-14860) from the Spanish Ministry of Education and Science. José Alfonso Aguilar is subventioned by CONACYT (Consejo Nacional de Ciencia y Tecnología) Mexico and University of Sinaloa, Mexico.

References

[Aguilar et al. 2010] Aguilar, J. A., Garrigós, I., Mazón, J. N., and Trujillo, J. "Web engineering approaches for requirements analysis: a systematic litarature review". In Proceedings of the WEBIST, pages 187–190, 2010.

[Almeida et al. 2006] Almeida, J., Iacob, M. and van Eck, P. "Requirements traceability in model-driven development: Applying model and transformation conformance", 2006.

[Bolchini and Mylopoulos 2003] Bolchini, D., Mylopoulos, J. "From task-oriented to goal-oriented Web requirements analysis". In Proceedings of the Fourth International Conference on Web Information Systems Engineering, WISE, pages 166–175, 2003.

[Bolchini and Paolini 2004] Bolchini, D., Paolini, P. "Goal-driven requirements analysis for hypermedia-intensive web applications". In Proceedings of the Requirements Engineering Conference, pages 85–103, 2004.

[Brown 2004] Brown, A. "Model driven architecture: Principles and practice". Software and Systems Modeling, pages 314–327, 2004.

[Bush and Koch 2009] Busch, M., Koch, N. "MagicUWE–A CASE Tool Plugin for Modeling Web Applications". In Proceedings of the 9th International Conference on Web Engineering, pages 505–508. 2009.

[Cachero and Gómez 2002] Cachero, C., Gómez, J. "Advanced conceptual modeling of web applications: Embedding operation interfaces in navigation design". In Proceedings of 21th International Conference on Conceptual Modeling. JISBD, pages 235–248, 2002.

[Casteleyn et al. 2005] Casteleyn, S., Garrigós, I. and De Troyer, O. "Automatic runtime validation and correction of the navigational design of web sites". In Proceedings of Web Technologies Research and Development, APWeb, pages 453–463, 2005.

[Casteleyn et al. 2007] Casteleyn, S., Van Woensel, W., Houben, G. "A semantics-based aspect-oriented approach to adaptation in web engineering". In Proceedings of the eighteenth conference on Hypertext and Hypermedia, pages 189–198, 2007.

[Ceri et al. 2000] Ceri, S. Fraternali, P. and Bongio, A. "Web Modeling Language (WebML): a modeling language for designing Web sites". In Proceedings of the First ICSE Workshop on Web Engineering, International Conference on Software Engineering, pages 137–157, 2000.

[Ceri and Manolescu 2003] Ceri, S. and Manolescu, I. "Constructing and integrating data-centric web applications: Methods, tools, and techniques". In Proceedings of the 29th international conference on Very large data bases, VLDB, page 1151, 2003.

[Escalona and Koch 2004] Escalona, M.J., and Koch, N. "Requirements engineering for web applications - a comparative study". Journal of Web Engineering, Vol. 2, No.3, pages 193–212, 2004.

[Escalona and Koch 2006] Escalona, M.J., and Koch, N. "Metamodeling the requirements of web systems". In Proceedings of WEBIST, pages 310–317, 2006.

[Daniel et al. 2007] Daniel, F., Matera, M., Morandi,A., Mortari, M. and Pozzi, G. "Active rules for runtime adaptivity management". In Proceedings of AEWSE, 2007.

[De Troyer and Leune 1998] De Troyer, O. and Leune, C. "WSDM: a user centered design method for Web sites". Computer Networks and ISDN Systems, pages 85–94, 1998.

[Eclipse] Eclipse. `http://www.eclipse.org/`.

[EMF] EMF. `http://www.eclipse.org/emf/`.

[Escalona and Aragón 2008] Escalona, M.J. and Aragón, G. "NDT. A model-driven approach for web requirements". IEEE Transactions on Software Engineering, pages 34(3):377–390, 2008.

[Estrada et al. 2006] Estrada, H., Martínez, A., Pastor,O. and Mylopoulos, J. "An empirical evaluation of the * framework in a model-based software generation environment". In Lecture Notes in Computer Science, CAiSE, pages 513–527, 2006.

[Fons et al. 2003] Fons, J., Valderas, P., Ruiz, M., Rojas, G., Pastor, O. "Oows: A method to develop web applications from web-oriented conceptual models.". In Proceedings of IWWOST, 2003.

[Garrigós 2008] Garrigós, I. "A-OOH: Extending Web Application Design with Dynamic Personalization". PhD thesis, University of Alicante, Spain, 2008.

[Garrigós et al. 2009] arrigós, I., Mazón, J.N. and Trujillo, J. "A requirement analysis approach for using i* in web engineering". In Lecture Notes in Computer Science, Volume 5648, pages 151-165, 2009.

[i Star Wiki] i* wiki. `http://istar.rwth-aachen.de`.

[Koch 2000] Koch, N. "Reference model, modeling techniques and development process software engineering for adaptive hypermedia systems". PhD thesis, LMU, Munich, 2000.

[Koch et al. 2006] Koch, N., Zhang, G. and Escalona, M.J. "Model transformations from requirements to web system design". In Proceedings of ICWE, pages 281–288, 2006.

[Meliá and Gómez 2006] Meliá, S. and Gómez, J. "The WebSA approach: Applying model driven engineering to web applications". Journal of Web Engineering, pages 121–149, 2006.

[Molina et al. 2008] Molina, F., Pardillo, J. and Ambrosio Toval, J. "Modelling web-based systems requirements using wrm". In Lecture Notes in Computer Science in WISE Workshops, pages 122-131, 2008.

[QVT Specification] QVT Language. `http://www.omg.org/cgi-bin/doc?ptc/2005-11-01`.

[Rossi et al. 2001] Rossi, G. Schwabe, D. and Guimarães, R. "Designing personalized web applications". In Proceedings of the 10th international conference on World Wide Web, WWW, pages 275-284, 2001.

[Schwabe 1998] Schwabe, D. and Rossi, G. "An object oriented approach to web-based applications design". Theory and practice of object systems, TAPOS, pages 207-225, 1998.

[Strohmaier et al. 2008] Strohmaier, M. Horkoff, J., Yu, E., Aranda, J. and Easterbrook, E. "Can patterns improve i* modeling? two exploratory studies". In Proceedings of REFSQ, pages 153-167, 2008.

[Valverde et al. 2007] Valverde, F., Valderas, P. and Fons, J. "OOWS Suite: Un Entorno de desarrollo para Aplicaciones Web basado en MDA". In Proceedings of IDEAS, 2007.

[WebML] Web Modeling Language. `http://www.webml.org/`.

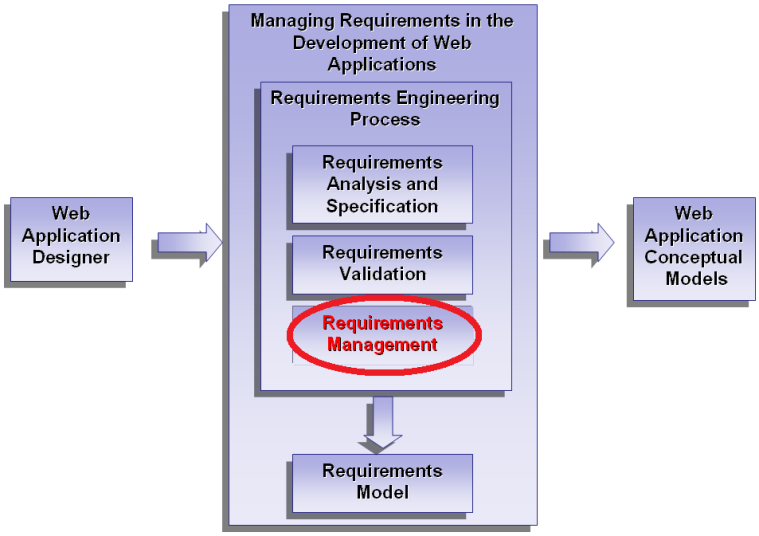
[Yu 1995] Yu, E. "Modelling Strategic Relationships for Process Reenginering". PhD thesis, University of Toronto, Canada, 1995.

[Yu 1997] Yu, E. "Towards modeling and reasoning support for early-phase requirements engineering". In Proceedings of RE, pages 226–235, 1997.

4

Impact Analysis of Goal-Oriented Requirements in Web Engineering

El contenido del capítulo corresponde con el artículo: ***J.A. Aguilar, I. Garrigós, J.-N. Mazón. Impact Analysis of Goal-Oriented Requirements in Web Engineering. The 11th International Conference on Computational Science and Its Applications (ICCSA 2011), June 20-23, 2011, Santander, Spain. Part V, Lecture Notes in Computer Science, Vol. 6786, pp. 421-436, 2011.***



En capítulos anteriores se ha resaltado la importancia de la etapa de análisis y especificación de requisitos en la ingeniería Web, obligada, principalmente, por las características particulares de este tipo de aplicaciones, tales como su audiencia heterogénea y por la evolución constante en las tecnologías de implementación. Este tipo de características originan que la aplicación Web sea propensa a sufrir cambios, por eso, es importante conocer en qué medida impactarán los cambios a los requisitos, así como qué partes de la aplicación Web se verán afectadas. Para lograrlo, es necesario comprender y analizar las dependencias entre los requisitos, es decir, cuales requisitos están relacionados o cuales dependen uno del otro para cumplirse y con ello brindar soporte al diseñador por medio de una mejor gestión y mantenimiento de la aplicación Web.

En este capítulo, se presenta un algoritmo para manejar las dependencias entre los requisitos funcionales y los requisitos no-funcionales de la aplicación Web en un contexto orientado a objetivos (*goal-oriented*). Con el algoritmo, es posible comprender cuál es el impacto en los requisitos procedente de un cambio en la estructura de modelos conceptuales que conforman la aplicación Web, así como saber qué requisitos necesitan ser implementados para cumplir, en medida de lo posible, los propósitos establecidos en el análisis orientado a objetivos.

Impact Analysis of Goal-Oriented Requirements in Web Engineering

José Alfonso Aguilar[1,2], Irene Garrigós[1], and Jose-Norberto Mazón[1]

[1] Department of Software and Computing Systems
University of Alicante, Spain
[2] Computer Science Faculty
University of Sinaloa, Mexico
{ja.aguilar,igarrigos,jnmazon}@dlsi.ua.es

Abstract. Due to the continuous changes and heterogeneous audience of the Web, a requirement engineering stage is crucial for Web development. Importantly, this stage should consider that Web applications are more likely to rapidly evolve during the development process, thus leading to inconsistencies among requirements. Therefore, Web developers need to know dependencies among requirements to ensure that Web applications finally satisfy the audience. The understanding of requirement dependencies also helps in better managing and maintaining Web applications. In this work, an algorithm has been defined in order to deal with dependencies among functional and non-functional requirements to understand which is the impact of making changes when developing a Web application.

Keywords: Impact analysis, goal-oriented requirements engineering, Web engineering.

1 Introduction

Requirements in Web engineering tend to rapidly evolve due to the dynamic nature of the Web [1]. This continuous evolution may lead to inconsistencies among requirements which hinder Web developers from understanding the impact of a change in the Web application. To tackle this problem, dependencies among requirements should be explicitly considered, thus better managing and maintaining requirements in Web applications. Dependencies are essential elements among the requirements of a real software system to achieve certain stakeholder goals. On the other hand, inconsistencies are the negative dependencies among the set of requirements, caused by the fact that requirements often originate from stakeholders with different or conflicting viewpoints [2].

Impact analysis is the task of identifying the potential consequences of a change, or estimating what needs to be modified to accomplish a change [3]. We define a "change" as any modification on a Web requirement. Usually, impact analysis has been done intuitively by Web applications developers, after some cursory examination of the code and documentation. This may be sufficient for

B. Murgante et al. (Eds.): ICCSA 2011, Part V, LNCS 6786, pp. 421–436, 2011.

small Web applications, but it is not enough for sophisticated ones. In addition, empirical investigation shows that even experienced Web applications developers predict incomplete sets of change impacts [4].

Therefore, to effectively manage changes in Web applications, information about dependencies among requirements should be considered in order to know how changes in a Web requirement affect the other requirements, thus making the right design decisions.

Usually, impact analysis is performed only on functional requirements, leaving aside the non-functional requirements as shown in [5,6]. According to [7], we believe that non-functional requirements must be considered in the Web application development from the very beginning of the development process. It is worth noting that the impact analysis should be done on both kind of requirements: functional and non-functional.

Functional requirements describe system services, behavior or functions, whereas non-functional requirements, or quality requirements, specify a constraint on the system or on the development process [8]. Functional requirements are related to *goals* and *sub-goals* in goal-oriented modeling. Non-functional requirements are named *softgoals* in goal-oriented modeling to represent objectives that miss clear-cut criteria. Specifically, finding the right tradeoff for non-functional requirements is an important step for achieving successful software [9,10], i.e., a Web application without passwords is usable, but not very secure, increased usability reduce security or increased security reduce usability. Therefore, it is necessary to identify the dependencies among non-functional and functional requirements for their achievement. Unfortunately, finding this tradeoff is not a trivial task due to the conflicts that commonly arise among them. Interestingly, the recent inclusion of goal-oriented techniques in Web requirements engineering [11,12,13,14] offer a better analysis in Web application design, due to the fact that requirements are explicitly specified in goal-oriented models. This has allowed the *stakeholders* to understand among the design decisions that can be taken to satisfy their goals and evaluating the implementation of certain functional and non-functional requirements in particular. However, this is not enough to ensure that the Web application satisfies the goals.

This paper presents a goal-oriented proposal for supporting Web developers analyzing the impact of a requirement change in Web applications in order to provide information about the different design alternatives. Therefore, more informed design decisions can be made for developing a Web application that fully-satisfies goals, while there is a tradeoff among softgoals.

The remainder of this paper is structured as follows: Section 2 presents some related work relevant to the context of this work. Section 3 describes the proposal for goal-oriented requirements analysis where is found the contribution of this work and introduces a running example for demonstration purposes. The algorithm for impact analysis in goal-oriented requirements is presented in Section 4. The application of the algorithm to perform the impact analysis is described step by step in Section 5. Finally, the conclusion and future work is presented in Section 6.

2 Related Work

In our previous work [15], a systematic literature review has been conducted for studying requirement engineering techniques in the development of Web applications. Our findings showed that most of Web engineering approaches focus on the analysis and design phases and do not give a comprehensive support to the requirements phase (such as OOHDM [16], WSDM [17] or Hera [18]). We can also conclude that the most used requirement analysis technique is UML use cases (applied by OOWS [19], WebML [20], NDT [21] and UWE [22]).

Furthermore, none of the aforementioned Web engineering approaches perform the analysis and modeling of the users' needs for ensuring that the Web application satisfies real goals, i.e. users are not overloaded with useless functionalities, while important functionalities are not missed. We believe that these facts are an important issue that limits a broaden use of these approaches. In this sense, to the best of our knowledge, the only approaches that use goal-oriented requirements analysis techniques for Web engineering have been presented in [23,24]. Unfortunately, although these approaches use the i^* modeling framework [25,26] to represent requirements in Web domain, they do not benefit from every i^* feature. To overcome this situation, our previous work [13] adapts the well-known taxonomy of Web requirements presented in [27] for the i^* framework.

Regarding approaches that consider non-functional requirements from early stages of the development process, in [28] the authors propose a metamodel for representing usability requirements for Web applications. Moreover, in [7] the authors present the state-of-the-art for non-functional requirements in model-driven development, as well as an approach for integrating non-functional requirements into a model-driven development process by considering them from the very beginning of the development process. Unfortunately, these works overlook how to analyze and evaluate the impact among functional and non-functional requirements. However, some interesting works have been done in this area [29] and [30]. These works evaluate i^* models based upon an analysis question (what-if) and the human judgment. To this aim, this procedure uses a set of evaluation labels that represent the satisfaction or denial level of each element in the i^* model. First of all, initial evaluation labels reflecting an analysis question are placed in the model. These labels are then propagated throughout the model by using a combination of set propagation rules and the human judgment. The results of this propagation are interpreted in order to answer the stated question. Unfortunately, these general approaches have not been adapted to Web engineering.

The motivation regarding this proposal relies in the fact that, the works previously mentioned are focused on how to analyze i^* models to answer a particular question (what-if) without considering the goal satisfaction (the organizational objectives). Thus, our proposal is focused on how to evaluate the impact derived from a change in a i^* requirements model in the Web engineering field. For this purpose, in the i^* model the requirements are classified according the taxonomy of Web requirements defined in [27]. Also, our proposal brings out alternative

paths to satisfy the goals bearing in mind the softgoals tradeoff, hence, considering the softgoals from the beginning of the Web application development process.

To sum up, there have been many attempts to provide techniques and methods to deal with some aspects of the requirements engineering process for Web applications. However, there is still a need for solutions that allow an impact analysis by considering non-functional requirements.

3 Goal-Oriented Requirements Analysis in Web Engineering

This section briefly describes our proposal to specify requirements in the context of a Web modeling method by using *i** models [14]. As a goal-oriented analysis technique, the *i** framework focuses on the description and evaluation of alternatives and their relationships to the organizational objectives. This proposal supports an automatic derivation of Web conceptual models from a requirements model by means of a set of transformation rules [31]. The proposal presented in this paper is defined upon our Web engineering method A-OOH *(Adaptive Object Oriented Hypermedia method)* [32] although it can be applied to any other Web engineering approach.

Next, we shortly describe an excerpt of the *i** framework which is relevant for the present work. For a further explanation, we refer the reader to [25,26]. The *i** framework consists of two models: the strategic dependency (SD) model to describe the dependency relationships (represented as ⊣◗⊢) among various actors in an organizational context, and the strategic rationale (SR) model, used to describe actor interests and concerns and how they might be addressed. The SR model (represented as ◌) provides a detailed way of modeling internal intentional elements and relationships of each actor (◯). Intentional elements are goals (⬭), tasks (⬡), resources (▭) and softgoals (☁). Intentional relationships are means-end links (⊣▷⊢) representing alternative ways for fulfilling goals; task-decomposition links (—┼—) representing the necessary elements for a task to be performed; or contribution links (help / hurt →) in order to model how an intentional element contributes to the satisfaction or fulfillment of a softgoal.

Although *i** provides good mechanisms to model actors and relationships between them, it needs to be adapted to the Web engineering domain to reflect special Web requirements that are not taken into account in traditional requirement analysis approaches. To do this, in first place, we use the taxonomy of Web requirements presented in [27]:

- **Content Requirements.** With this type of requirements, is defined the website content presented to users. For example, in a book on-line store some examples might be: "book information" or "book categories".
- **Service Requirements.** This type of requirement refers to the internal functionality the system as Web application should provide to its users. Following the example of the Content Requirements, for instance: "register a new client", "add book to cart", etc.

- **Navigational Requirements.** A Web system must also define the navigational paths available for the existing users. In this sense, some examples are: the user navigation from index page to "consult products by category" or to "consult shopping cart" options.
- **Layout Requirements.** Requirements can also define the visual interface for the users. For instance: "present a color style", "multimedia support", "the user interaction", among others.
- **Personalization Requirements.** The designer can specify the desired personalization actions to be performed in the final website (e.g. "show recommendations based on interest", "adapt font for visual impaired users", etc.)
- **Non-Functional Requirements.** These kind of requirements are related to quality criteria that the intended Web system should achieve and that can be affected by other requirements. Some examples can be "good user experience", "attract more users", "efficiency", etc.

As the considered Web engineering approach (A-OOH) is UML-compliant, we have used the extension mechanisms of UML to (i) define a profile for using *i** within UML; and (ii) extend this profile in order to adapt *i** to specific Web domain terminology. Therefore, new stereotypes have been added according to the different kind of Web requirements (see Fig. 1): *Navigational*, *Service*, *Personalization* and *Layout* stereotypes extend the *Task* stereotype and *Content* stereotype extends the *Resource* stereotype. It is worth noting that non-functional requirements can be modeled by directly using the softgoal stereotype. The UML-Profile for goal-oriented requirements using *i** has been implemented in Eclipse [33].

A sample application of the *i** modeling framework for Web domain is shown in Figure 2, which represents the SR model of our running example for the Conference Management System (CMS). The purpose of the system is to support

Fig. 1. Overview of the UML-Profile for *i** modeling in the Web domain implemented in Eclipse

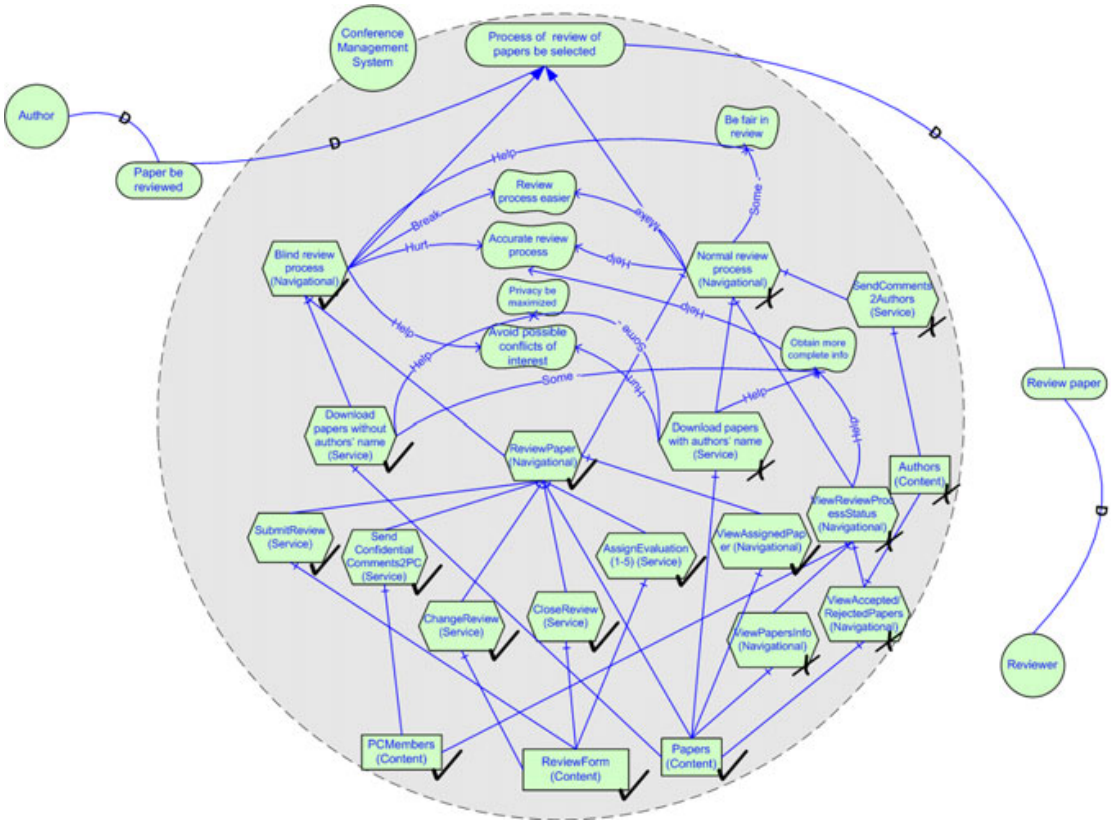


Fig. 2. Part of the Conference Management System requirements expressed in a SR and SD Models

the process of submission, evaluation and selection of papers for a conference [34].[1]

In Figure 2 is modeled a part of the CMS focused in the process of selecting the review process. Four actors participate in the CMS, but due to space limitations, for this example only the author, reviewer and system actors were considered. It is important to highlight that each element from Figure 2 corresponds to a requirements type from the taxonomy previously mentioned, i.e., the content requirement (Content) from the taxonomy is displayed with the notation *"Resource"* from *i** and the navigational (Navigational) and service (Service) requirements with the symbol *"Task"* from *i**, both with their respective associations (*decomposition-links*). A decomposition-link between two elements means that a requirement (*"Task"*) is decomposed in one or more sub-requirements (*"Sub-Tasks"*), Figure 2 depicts a correct scenario of the requirement decomposition by introducing the navigational requirement *"Blind Review Process"*, decomposed in two sub-requirements named *"Download papers without authors' name"* (Service) and *"Review Paper"* (Navigational). Also, the labels (✓) and (✗) are used to represent the requirements currently implemented in the Web application.

[1] The complete specification of the case study can be found at: `http://users.dsic.upv.es/~west/iwwost01`

Three actors are detected that depend on each other, namely *"Reviewer"*, *"Author"* and *"Conference Management System"*. The reviewer needs to use the CMS to *"Review paper"*. The author depends on the CMS in order to *"Paper be reviewed"*. These dependencies and the CMS actor are modeled by a SD and SR models in Figure 2.

The goal of the CMS actor is *"Process of review of papers be selected"*. To fullfill this goal, the SR model specifies that one of the two navigational requirements: *"Blind review process"* or *"Normal review process"* should be performed. In this running example, the path to achieve the goal of the CMS actor is by means of the navigational requirement *"Blind review process"*, all the requirements implemented for this path are labeled with (✓). We can observe in the SR model that some navigational and service requirements are decomposed in other requirements, some of them affects positively or negatively some non-functional requirements, i.e., the service requirement *"Download paper without authors' name"* needs the content requirement *"Papers"*, also, affects positively the softgoal *"Privacy be maximized"* and in some negatively form the softgoal *"Obtain more complete info"*. This fact is very important to see how to satisfy the goal *"Process of review of papers be selected"* considering the Web application softgoals. Therefore, maximizing or minimizing the contribution from requirements to softgoals is a viable solution to find a path to fully-satisfy the goal.

4 An Impact Analysis Algorithm for Goal-Oriented Requirements in Web Engineering

In this section, we present a proposal which provides a way to analyze the impact of a change in an A-OOH conceptual model for the goal-oriented requirement approach for Web engineering described in the previous section. Therefore, the Web developer will be able to evaluate the effect of the change and select the best among several design options to fully satisfy goals based on maximizing softgoals.

The algorithm is designed to be applied in i^* requirements model considering the type of contributions made by the intentional elements to the softgoals. This algorithm allows to evaluate the impact in the requirements model resulting from removing any element of the A-OOH conceptual models. In this way, it is determined which new requirements should be included in the A-OOH conceptual models for maximizing softgoal satisfaction although some requirements have been removed. To this aim, some heuristics have been defined.

4.1 Heuristics

Some heuristics have been defined for determining the impact of the contribution links between intentional element and softgoals. Table 1 summarizes some terms for understanding the heuristics presented in this section. These terms correspond to the most common types of contribution links of the i^* modeling framework.

The *"Help"* contribution link is a partial positive contribution, not sufficient by itself to satisfy the softgoal. The contribution link named *"Hurt"* is partial negative contribution, not sufficient by itself to deny the softgoal. *"Some +"* is a positive contribution whose strength is unknown. *"Some -"* is the opposite contribution type to *"Some +"*, is a negative contribution whose strength is unknown. The *"Break"* contribution link refers to a negative contribution enough to deny a softgoal. Finally, the *"Make"* contribution link is a positive contribution strong enough to satisfy a softgoal.

Table 1. The *i** contributions types

Heuristics Terms	***i**** **Contribution Type**
Strongly-positive	Help
Weakly-positive	Some +
Strongly-negative	Hurt
Weakly-negative	Some -
Dependent-negative	Break
Dependent-positive	Make

The heuristics defined are:

- **H1.** If the contribution of the requirement to remove is *strongly-positive*, and the contribution of the requirement to implement is *strongly-negative*, **do not** implement the requirement.
- **H2.** If the contribution of the requirement to remove is more *strongly-positive* than the contribution of the requirement to implement, but the contribution to be implemented is *weakly-negative*, the requirement **could be** implemented.
- **H3.** If the contribution of the requirement to remove is *strongly-negative* or *weakly-negative*, and the contribution of the requirement to implement is *strongly-positive* or *weakly-positive*, the requirement **should be** implemented.
- **H4.** If the polarity of the contribution of the requirement to remove is as negative as the contribution of the requirement to implement, the requirement to implement **should not be** implemented to maximize the satisfaction of the softgoal.
- **H5.** If the polarity of the contribution of the requirement to remove is as positive as the contribution of the requirement to implement, the requirement to implement **should be** implemented to maximize the satisfaction of the softgoal.
- **H6.** If the contribution of the requirement to remove is *Dependent-positive*, then the developer should consider whether that requirement should be removed or not considering the need to implement this softgoal.
- **H7.** If the contribution of the requirement to remove is *Dependent-negative*, then the developer should consider whether that requirement should be removed or not considering the impact of not implementing this softgoal.

4.2 Preconditions and Postconditions

The preconditions should be launched before the execution of the algorithm and can be only applied to the elements implemented in the requirements model described in Section 3. Specifically, these preconditions permit the execution of the algorithm when:

1. The requirement to remove does not affect the goal by the *"means-end"* contribution type.
2. When there is more than one *"means-end"* contribution type, it means that the impact analysis will be possible by means of the softgoals tradeoff.
3. The requirement to remove affects other requirements and these requirements (not shared) are not in the possible paths to satisfy the goal.

Moreover, there is one postcondition to be applied to the elements defined in the *i** requirements model. This postcondition is applied when a requirement has been selected to be implemented in the requirements model as an alternative solution for the satisfaction of the goal: if the requirement to be implemented has associated requirements, these requirements must be implemented automatically.

4.3 Impact Analysis Algorithm

This algorithm considers the contributions made by the intentional elements from the requirements model to find a path to fully satisfy, where possible, the main goal. To do this, the designer must have to find tradeoffs between the softgoals. The algorithm is presented next.

```
1  FUNCTION TradeOffAlgorithm (RequirementsModel)
2     TR= task to remove; TI= task to implement;
3     SN= new softgoal; IEList= intentional elements list;
4     ASList= affected softgoals list;
5     TIList= list of task to implement; Value= false;
6     P = PreConditions();
7     IF (P=true) THEN
8        IEList= CreateIntentionalElementsList(RequirementsModel);
9        IF(TR.Contributes2Softgoals()) THEN
10          ASList=CreateAffectedSoftgoalsList(TR);
11          FOREACH s FROM ASList:
12             TI= SearchTaskToApply(IEList);
13             Value= Heuristics(ASList,IEList,TI);
14             TI.AddValue(Value);
15             TIList.Add(TI);
16                IF (TI.Contributes2Softgoals(TI)) THEN
17                   ASList.add(SN);
18                END IF
19          END FOREACH
20          FOREACH v FROM TIList:
21             CalculateAverage(v);
22             IF(CalculateAverage(v)) THEN
```

```
23                    Implements(v);
24                END IF
25            END FOREACH
26          END IF
27        PostCondition();
28        ELSE
29          ShowMessage(P.message());
30        END IF
31  END PROGRAM
```

Algorithm 1.1. Algorithm for impact analysis in goal-oriented Web engineering

A snippet of code that represents our algorithm for impact analysis of goal-oriented requirements in Web engineering is shown in 1.1. First, in lines 6 and 7 the pre-conditions are evaluated (Section 4.2), these must be "true" to proceed with the execution of the algorithm. Next, from lines 8 to 19, the algorithm creates a list of intentional elements. In these lines, all types of requirements from the requirements model are stored in this list. The next step is to extract those softgoals that receive a contribution from the requirement to remove, for each softgoal from the list, finding a non-implemented requirement and applying the heuristics introduced in Section 4.1. Each of these requirements must be stored in the list, and if it contributes to a softgoal, the softgoal must be stored in the list too. Then, lines 20 to 29 are used to evaluate each element from the list according to the weight of each element assigned by the heuristics to determine when a requirement must be implemented. Finally, the postcondition is executed and the alternative path to fully satisfy the goal from requirements model is obtained.

5 Performing the Impact Analysis

Lets suppose the following scenario: the Web developer decides deleting from the domain model of A-OOH the elements that correspond to the requirement *"Download papers without authors' name"*. It is necessary to know which other requirements are affected by this change. In addition, this action implies that the goal *"Process of review of papers be selected"* can not be satisfied. Thus, it is necessary to search for alternative paths in the *i** requirements model (if there any) in order to fully-satisfy the goal *"Process of review papers be selected"*. To this aim, our algorithm is triggered. The execution of the impact analysis algorithm is detailed next.

The first step to execute our algorithm consists of applying the preconditions. For this running example, the preconditions result true, it means that there is any problem to the algorithm has been executed.

Next, it is necessary to develop a list of the requirements (implemented or not) that contribute to any softgoal in the *i** requirements model (see Table 2). Also, if a softgoal contributes to other one, the softgoal must be added to the list too.

Table 2. The requirements contributions to softgoals

Requirements	*"S1"*	*"S2"*	*"S3"*	*"S4"*	*"S5"*	*"S6"*
"Blind review process"	Help	Break	Hurt	Help	-	-
"Download papers without authors' name"	**-**	**-**	**-**	**-**	**Help**	**Some -**
"Normal review process"	Some -	Make	Help	-	-	-
"Download paper with authors' name"	-	-	-	Hurt	Some -	Help
"View review process status"	-	-	-	-	-	Help
"Obtain more complete info"	-	-	Help	-	-	-

Table 2 highlights in bold the requirement to be removed. This table shows a requirements list (functional and non-functional) and their type of contributions to the softgoals where S1 corresponds to softgoal *"Be fair in review"* from requirements model, S2 to *"Review process easier"*, S3 represents *"Accurate review process"*, S4 conforms to *"Avoid possible conflicts of interest"*, S5 its the *"Privacy be maximized"* softgoal and S6 refers to *"Obtain more complete info"*.

The next step is to identify the number of softgoals affected by the requirement to be removed. If necessary, a list of the softgoals that receive a contribution from the requirement to be removed is made. In this example, the requirement to be removed is *"Download papers without authors' name"*, this one affects two softgoals: *"Privacy be maximized"* and *"Obtain more complete info"* S5 y S6 respectively (see Table 2).

For each softgoal that receives a contribution from the requirement to be removed, we search for a non-implemented requirement of which contribution compensates the possible elimination of the requirement to be removed. To do this, it is necessary to apply the heuristics defined in Section 4.1.

For example, the softgoal *"Privacy be maximized"*, according to Table 1, receives a *strongly-positive* contribution (Help) from the requirement to be removed, thus being necessary searching for a non-implemented requirement to contribute to this softgoal. In this case, only the requirement *"Download papers with authors' name"* contributes (negatively) to this softgoal (*weakly-negative*, i.e. Some -). Therefore, applying the heuristics described in Section 4.1, specifically the heuristic number 2 (H2), the requirement *"Download papers with authors' name"* could be implemented.

Considering the softgoal *"Obtain more complete info"* according to Table 1, it receives a *weakly-negative* contribution (Some -) from the requirement to be removed, thus being necessary searching for a non-implemented requirement to contribute to this softgoal. In this case, two requirements (positively) contribute to this softgoal, *"Download papers with authors' name"* and *"View review process status"* (*strongly-negative*, i.e. Help). Therefore, the heuristic H3 applies for this softgoal, thus, these requirements should be implemented.

After analyzing the softgoals contributions, the next step is searching for any softgoal in the requirements list that contributes to another softgoal. In this example, the softgoal *"Obtain more complete info"* makes a *strongly-positive* contribution (Help) to the softgoal *"Accurate review process"*, thus, the next step consists of searching for the requirement that makes a contribution to the softgoal and applying the heuristics. Therefore, the requirement that makes a

contribution to the softgoal *"Accurate review process"* is *"Normal review process"*, this contribution is *strongly-positive* (see Figure 2), hence, according to H3 this requirement must be implemented.

After these steps, Table 3 shows requirements that could be implemented to fully-satisfy the goal *"Process of review papers be selected"* after having removed the requirement *"Download papers without authors' name"*. Next, it is necessary to evaluate the heuristics assigned to each requirement to know what could be implemented.

Table 3. Non-implemented requirements that contributes to softgoals

Intentional element	*"S5"*	*"S4"*	*"S6"*	*"S3"*	*Result*
"Download papers with authors' name"	H2 (Some -)	H1 (Hurt)	H3 (Help)	-	Implement
"Normal review process"	-	-	-	H3 (Help)	Implement
"View review process status"	-	-	H3 (Help)	-	Implement

Table 3 shows the results after having performed the algorithm. In this table the requirements that must be implemented in order to fully-satisfy the goal *"Process of review papers be selected"* are shown. To do this, it is necessary to evaluate the contribution type of each requirement, i.e., the navigational requirement *"Download papers with authors' name"* negatively contributes (Some -) to the softgoal *"Privacy be maximized"*, thus hurting the softgoal *"Avoid possible conflicts of interest"* and helping the softgoal *"Obtain more complete info"*. Therefore, by using the human judgment the navigational requirement *"Download papers with authors' name"* can be implemented. For the navigational requirement *"Normal review process"*, it is easier to determine whether it can be implemented because it only contributes to one softgoal, the *"Accurate review process"*, hence its contribution is Help, therefore this requirement must be implemented. Finally, the navigational requirement *"View review process status"* positively contributes to the softgoal *"Obtain more complete info"*, consequently this requirement must be implemented.

The final step is to apply the postcondition from Section (4.2). In this running example, according to the postcondition, it is necessary to implement the navigational requirements *"View papers info"* and *"View Accepted/Rejected papers"* because these requirements are associated with the navigational requirement *"View Review Process Status"*. In addition, the content requirement *"Authors"* and the service requirement *"Send Comments to Authors"* must be implemented too in order to implement the alternative path to fully satisfy the goal *"Process of review papers be selected"*. Hence, the content requirement *"Authors"* is associated with the navigational requirement *"View Accepted/Rejected papers"* and the service requirement *"Send Comments to Authors"* is related with the navigational requirement *"Normal review process"*.

After finishing the execution of the algorithm, we obtain the requirements that are directly and indirectly affected by the deletion of the requirement *"Download papers without authors' name"*. Moreover, the algorithm can find out which requirements must be implemented to continue satisfying the goal considering the contributions received from the softgoals. In this running example the

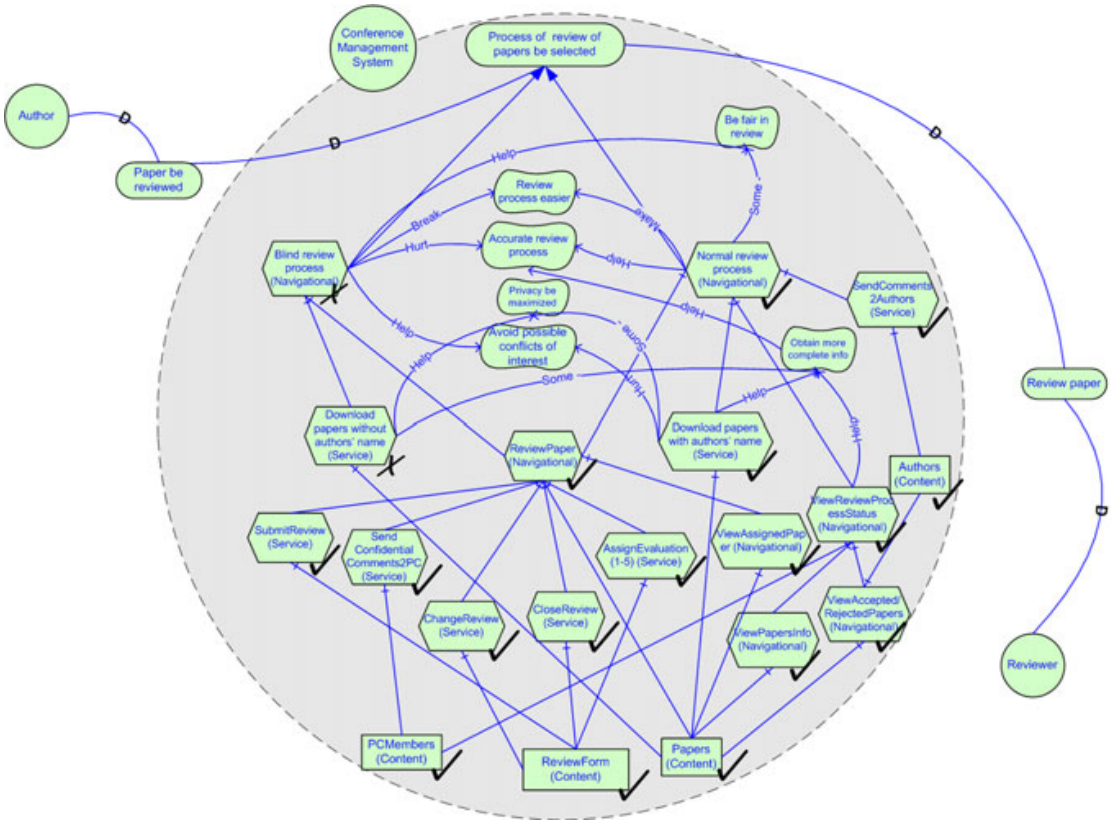


Fig. 3. Conference Management System requirements expressed in a SR and SD Models with the alternative path to fully satisfy the goal *"Process of review papers be selected"*

requirements to implement are: *"Download papers with authors' name"*, *"Normal review process"* and *"View review process status"*. Finally, according to the post-condition the requirements *"View papers info"*, *"View Accepted/Rejected papers"*, *"Authors"* and *"Send Comments to Authors"* must be implemented too. Figure 3 shows the final requirements model with the alternative path implemented to fully-satisfy the goal *"Process of review papers be selected"*.

6 Conclusions and Future Work

Due to the dynamic idiosyncrasy of the Web and its heterogeneous audience Web applications should consider a requirement analysis phase in order to reflect, from the early stages of the Web application development process, specific needs, goals, interests and preferences of each user or user type, and due to the fast evolution of the Web, possible changes in those requirements should also be managed.

In this work, we have presented a methodology based on the *i** modelling framework to specify Web requirements. Moreover, an algorithm to analyze the impact derived from a change done in the requirements model is presented.

Benefits of applying the algorithm described in this work include both the analysis of the impact derived from a change in a conceptual model and the

ability to find an alternative path to fully-satisfy the goal by means of the softgoals tradeoff.

Nevertheless, according to [7], the softgoals are not considered with sufficient importance from the early stages of the development process. In this context, our proposal makes a contribution to the requirements analysis field considering the softgoals, hence it allows the designer to make decisions from the very beginning stages of the development process that would affect the structure of the envision website in order to satisfy users needs. Therefore, the designer can improve the quality of the requirements model analyzing the balance of the softgoals with the stakeholders.

Our short-term future work includes the definition of a metamodel to help to record the relationships among functional and non-functional requirements to adapt the tradeoff algorithm presented in this work.

Finally, note that this work has been done in the context of the A-OOH modeling method, however it can be applied to any Web modeling approach.

Acknowledgments. This work has been partially supported by the following projects: MANTRA (GV/2011/035) from Valencia Ministry, MANTRA (GRE09-17) from the University of Alicante, SERENIDAD (PEII-11-0327-7035) from Junta de Comunidades de Castilla La Mancha (Spain) and by the MESOLAP (TIN2010-14860) project from the Spanish Ministry of Education and Science. José Alfonso Aguilar is subventioned by CONACYT (Consejo Nacional de Ciencia y Tecnología) Mexico and University of Sinaloa, Mexico.

References

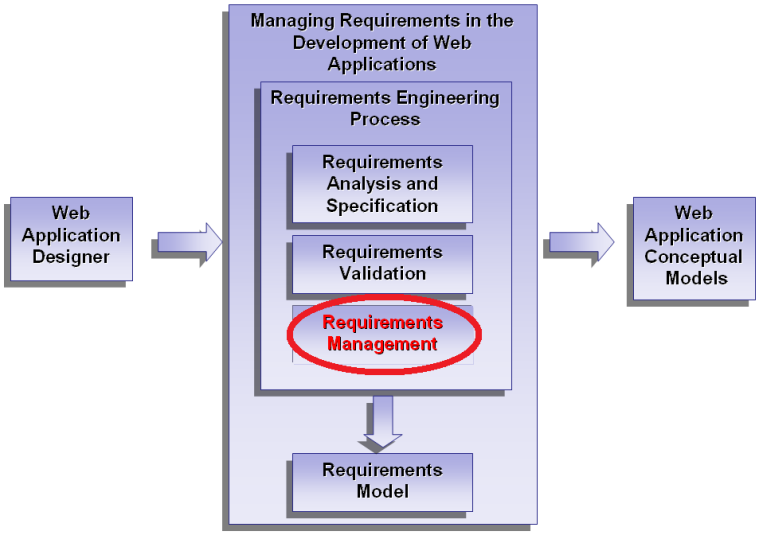
1. Ginige, A.: Web engineering: managing the complexity of web systems development. In: SEKE, pp. 721–729 (2002)
2. Zhang, W., Mei, H., Zhao, H.: A feature-oriented approach to modeling requirements dependencies (2005)
3. Arnold, R., Bohner, S.: Impact analysis-towards a framework for comparison. In: Proceedings of Conference on Software Maintenance, CSM 1993, pp. 292–301. IEEE, Los Alamitos (2002)
4. Lindvall, M., Sandahl, K.: How well do experienced software developers predict software change? Journal of Systems and Software 43(1), 19–27 (1998)
5. Zhang, S., Gu, Z., Lin, Y., Zhao, J.: Celadon: A change impact analysis tool for Aspect-Oriented programs. In: Companion of the 30th International Conference on Software Engineering, pp. 913–914. ACM, New York (2008)
6. Gupta, C., Singh, Y., Chauhan, D.: Dependency based Process Model for Impact Analysis: A Requirement Engineering Perspective. International Journal 6
7. Ameller, D., Gutiérrez, F., Cabot, J.: Dealing with non-functional requirements in model-driven development. In: 18th IEEE International Requirements Engineering Conference, RE (2010)
8. Sommerville, I.: Software Engineering, 6th edn. Addison-Wesley, Reading (2001)
9. Boehm, B., In, H.: Identifiing Quality-Requirement Conflicts (2002)

10. Elahi, G., Yu, E.: Modeling and analysis of security trade-offs - a goal oriented approach. Data and Knowledge Engineering 68(7), 579–598 (2009); Special Issue: 26th International Conference on Conceptual Modeling (ER 2007) - Six selected and extended papers
11. Nuseibeh, B., Easterbrook, S.M.: Requirements engineering: a roadmap. In: ICSE - Future of SE Track, pp. 35–46 (2000)
12. Bolchini, D., Mylopoulos, J.: From task-oriented to goal-oriented web requirements analysis. In: WISE 2003: Proceedings of the Fourth International Conference on Web Information Systems Engineering, p. 166. IEEE Computer Society, Washington, DC, USA (2003)
13. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: An mda approach for goal-oriented requirement analysis in web engineering. J. UCS 16(17), 2475–2494 (2010)
14. Garrigós, I., Mazón, J.-N., Trujillo, J.: A requirement analysis approach for using i* in web engineering. In: ICWE, pp. 151–165 (2009)
15. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: Web Engineering approaches for requirement analysis- A Systematic Literature Review. In: 6th Web Information Systems and Technologies (WEBIST), Valencia, Spain, vol. 2, pp. 187–190. SciTePress Digital Library (2010)
16. Schwabe, D., Rossi, G.: The object-oriented hypermedia design model. Communications of the ACM 38(8), 45–46 (1995)
17. De Troyer, O.M.F., Leune, C.J.: Wsdm: a user centered design method for web sites. Comput. Netw. ISDN Syst. 30(1-7), 85–94 (1998)
18. Casteleyn, S., Van Woensel, W., Houben, G.-J.: A semantics-based aspect-oriented approach to adaptation in web engineering. In: Hypertext, pp. 189–198 (2007)
19. Fons, J., Valderas, P., Ruiz, M., Rojas, G., Pastor, O.: Oows: A method to develop web applications from web-oriented conceptual models. In: International Workshop on Web Oriented Software Technology (IWWOST), pp. 65–70 (2003)
20. Ceri, S., Fraternali, P., Bongio, A.: Web modeling language (webml): a modeling language for designing web sites. The International Journal of Computer and Telecommunications Networking 33(1-6), 137–157 (2000)
21. Escalona, M.J., Aragón, G.: Ndt. a model-driven approach for web requirements. IEEE Transactions on Software Engineering 34(3), 377–390 (2008)
22. Koch, N.: The expressive power of uml-based web engineering. In: International Workshop on Web-oriented Software Technology (IWWOST), pp. 40–41 (2002)
23. Bolchini, D., Paolini, P.: Goal-driven requirements analysis for hypermedia-intensive web applications, vol. 9, pp. 85–103. Springer, Heidelberg (2004)
24. Molina, F., Pardillo, J., Toval, A.: Modelling web-based systems requirements using wrm. In: Web Information Systems Engineering (WISE) Workshops, pp. 122–131. Springer, Heidelberg (2008)
25. Yu, E.: Modelling Strategic Relationships for Process Reenginering. PhD thesis, University of Toronto, Canada (1995)
26. Yu, E.: Towards modeling and reasoning support for early-phase requirements engineering. In: RE, pp. 226–235 (1997)
27. Escalona, M.J., Koch, N.: Requirements engineering for web applications - a comparative study. J. Web Eng. 2(3), 193–212 (2004)
28. Molina, F., Toval, A.: Integrating usability requirements that can be evaluated in design time into model driven engineering of web information systems. Adv. Eng. Softw. 40, 1306–1317 (2009)
29. Horkoff, J., Yu, E.: Evaluating Goal Achievement in Enterprise Modeling–An Interactive Procedure and Experiences. The Practice of Enterprise Modeling, 145–160 (2009)

5

A Goal-Oriented Approach for Optimizing Non-Functional Requirements in Web Applications

El contenido del capítulo corresponde con el artículo: ***J.A. Aguilar, I. Garrigós, J.-N. Mazón. A Goal-Oriented Approach for Optimizing Non-Functional Requirements in Web Applications. The 8th th International Workshop on Web Information Systems Modeling (WISM 2011), held in conjunction with the International Conference on Conceptual Modeling (ER 2011), 31 October - 03 November 2011, Brussels, Belgium. Lecture Notes in Computer Science, Vol. 6999, pp. 14-23, 2011.***



En el Capítulo 5 se presenta una adaptación del algoritmo Optimización de Pareto para evaluar y seleccionar la configuración de requisitos óptima que maximice los requisitos no-funcionales de la aplicación Web. La idea es considerar a los requisitos no-funcionales desde la etapa de análisis y especificación de requisitos, con el fin de mejorar la calidad de la aplicación a desarrollar. La hipótesis del capítulo se fundamenta en cómo es que la implementación de los requisitos funcionales afecta o beneficia a los requisitos no-funcionales. Para esto, los requisitos no-funcionales deben de ser priorizados de acorde al contexto de los usuarios de la aplicación Web. Finalmente, la solución del algoritmo proporciona al diseñador de la aplicación un conjunto de configuraciones de entre las cuales podrá elegir qué requisitos funcionales implementar (configuración óptima) considerando la prioridad establecida por los *stakeholders* sobre los requisitos no-funcionales.

A Goal-Oriented Approach for Optimizing Non-functional Requirements in Web Applications

José Alfonso Aguilar, Irene Garrigós, and Jose-Norberto Mazón

Lucentia-DLSI
University of Alicante, E-03080, San Vicente del Raspeig, Alicante, Spain
{ja.aguilar,igarrigos,jnmazon}@dlsi.ua.es

Abstract. Web design methodologies should be able to provide a requirements analysis stage to consider the large and heterogeneous audience of Web applications. In this stage, non-functional requirements (NFRs) should be addressed in order to improve the quality of the Web application perceived by users. To this aim, different configurations of requirements could be implemented depending on the NFRs preferred by the Web user. Furthermore, prioritizing and making tradeoffs between NFRs is crucial for satisfying the audience of Web applications. Therefore, this work presents an algorithm based on the Pareto optimal approach to evaluate and select the optimal configuration of requirements for a Web application. To do this, the NFRs are maximized according to a priority list provided by the audience. Our approach is illustrated with a running example.

Keywords: non-functional requirements, Web engineering, goal-oriented requirements engineering.

1 Introduction

Unlike traditional stand-alone software, the audience of Web applications is both open and large. Therefore, users may have different goals and preferences and stakeholders (in this context, stakeholders are individuals or organizations who affect or are affected directly or indirectly by the development project in a positive or negative form [9]) should be able to cope with these heterogeneous needs by means of an explicit requirements analysis stage in which functional and non-functional requirements are considered [2].

Functional requirements (FRs) describe the system services, behavior or functions, whereas non-functional requirements (NFRs), also known as quality requirements, specify a constraint in the application to build or in the development process [7]. An effective definition of requirements improves the quality of the final product. Unfortunately, in most of the Web engineering approaches, a complete analysis of requirements is performed considering only FRs, thus leaving aside the NFRs until the implementation stage. We totally agree with [3] the argument that NFRs are a very important issue and must be considered from

O. De Troyer et al. (Eds.): ER 2011 Workshops, LNCS 6999, pp. 14–23, 2011.

the very beginning of the development process, to be analyzed in depth, in order to improve the quality of the Web application perceived by users.

Interestingly, the recent inclusion of goal-oriented techniques in Web requirements engineering [4] offers a better analysis in the requirements stage since requirements are explicity specified in goal-oriented models in order to support reasoning about organizational objectives, alternatives and implications, thus having a deep understanding about the domain. This has allowed the stakeholders to choose among the design decisions that can be taken to satisfy the goals and evaluate the implementation of certain requirements in particular (including NFRs). However, this is not enough to ensure that the Web application satisfies the real user needs because they do not offer mechanisms to maximize NFRs, i.e., to find a tradeoff between the FRs and NFRs.

Therefore, not paying attention to eliciting, documenting and tracking NFRs makes harder for the stakeholders to take design choices. Consequently, the quality of the Web application perceived by users will be affected negatively. Thus, NFRs need to be addressed to enable the stakeholder to choose among multiple configurations maximizing the NFRs, helping them to take design choices which positively affects the quality of the Web application, i.e., maximizing the NFRs navigability and accessibility, improves the browsing user experience.

Bearing these considerations in mind, this paper presents an algorithm that allows the stakeholder to evaluate the implementation of certain requirements considering the NFRs maximization. The algorithm is based on the Pareto optimal approach [10] , which is useful when there are multiple competing and conflicting objectives that need to be balanced. The algorithm thus constructs a group of configurations (called Pareto front) optimizing the NFRs. A configuration only can be considered on the Pareto front, if and only if, it maximizes a NFR and the other ones remain the same, or they are maximized too. From these balanced configurations, the stakeholder can select the final configuration taking into account the different NFRs from the beginning of the development process. The proposal presented in this paper is defined upon our Web engineering method A-OOH *(Adaptive Object Oriented Hypermedia method)* [1] although it can be applied to any other Web engineering approach.

The remainder of this paper is structured as follows: Section 2, presents related work relevant to the context of this work. Section 3, describes the proposal for goal-oriented requirements analysis where is found the contribution of this work and introduces a running example for demonstration purposes. The Pareto algorithm for softgoals maximization and its application (described step by step) is presented in Section 4. Finally, the conclusion and future work is presented in Section 5.

2 Related Work

In our previous work [2], a systematic literature review has been conducted for studying requirement engineering techniques in the development of Web applications. Our findings showed that most of the Web engineering approaches focus on

the analysis and design phases and do not give a comprehensive support to the requirements phase. Furthermore, the NFRs are considered in a isolated form, leaving them out of the analysis stage. In addition, we can also conclude that the most used requirement analysis technique is UML use cases and profiles. On the other side, with regard to approaches that consider NFRs from early stages of the development process, in [8] the authors propose a metamodel for representing usability requirements for Web applications. Moreover, in [3] the authors present the state-of-the-art for NFRs in a MDD (Model-Driven Development), as well as an approach for a MDD process (outside the field of Web engineering). Unfortunately, these works overlook how to maximize the NFRs.

To sum up, there have been many attempts to provide techniques and methods to deal with some aspects of the requirements engineering process for Web applications. Nevertheless, there is still a need for solutions that considers NFRs from beginning of the Web application development process, in order to assure that they will be satisfied at the same time that the functional requirements are met, improving the quality of the Web application perceived by users.

3 Specifying Requirements in Web Engineering

This section briefly describes our proposal to specify requirements in the context of a Web modeling method by using *i** models [6], [1]. As a goal-oriented analysis technique, the *i** framework focuses on the description and evaluation of alternatives and their relationships to the organizational objectives. This proposal supports an automatic derivation of Web conceptual models from a requirements model by means of a set of transformation rules.

Following, we shortly describe an excerpt of the *i** framework which is relevant for the present work. For a further explanation, we refer the reader to [11]. The *i** framework consists of two models: the strategic dependency (SD) model to describe the dependency relationships (represented as ) among various actors in an organizational context, and the strategic rationale (SR) model, used to describe actor interests and concerns and how they might be addressed. The SR model (represented as ) provides a detailed way of modeling internal intentional elements and relationships of each actor (). Intentional elements are goals (), tasks (), resources () and softgoals (). Intentional relationships are means-end links () representing alternative ways for fulfilling goals; task-decomposition links () representing the necessary elements for a task to be performed; or contribution links () in order to model how an intentional element contributes to the satisfaction or fulfillment of a softgoal. Possible labels for a contribution link are “Make”, “Some+”, “Help”, “Hurt”, “Some-”, “Break”, “Unknown”, indicating the (positive, negative or unknown) strength of the contribution.

To adapt *i** framework to the Web engineering domain we use the taxonomy of Web requirements presented in [5]. Next, we have used the extension mechanisms of UML to define a profile for using *i** to specific Web domain terminology. Therefore, new stereotypes have been added according to the different kind of Web requirements (NFRs are modeled directly using the *i** softgoal element).

A sample application of the *i** modeling framework for Web domain is shown in Figure 1, which represents the SR model of our running example for the Conference Management System (CMS), the complete specification of the case study can be found at: `http://users.dsic.upv.es/~west/iwwost01`. The purpose of the system is to support the process of submission, evaluation and selection of papers for a conference. It is important to highlight that each element from Figure 1 corresponds to a requirement type from the taxonomy previously mentioned, i.e., the content requirement (Content) from the taxonomy is displayed with the notation *"Resource"* from *i** and the navigational (Navigational) and service (Service) requirements with the symbol *"Task"* from *i**, both with their respective associations (*decomposition-links*). The extract of the CMS example is focused on the selection of the review process. Four actors participate in the CMS, but due to space limitations, for this example only the author, reviewer and system actors were considered. In this case, three actors are detected that depend on each other, namely *"Reviewer"*, *"Author"* and *"Conference Management System"*. The reviewer needs to use the CMS to *"Review paper"*. The author depends on the CMS in order to *"Paper be reviewed"*. These dependencies and the CMS actor are modeled by a SD and SR models in Figure 1.

The goal of the CMS actor is *"Process of review of papers be selected"*. To fullfill this goal, the SR model indicates that one of the two navigational requirements: *"Blind review process"* or *"Normal review process"* should be performed. In this running example, the path to achieve the goal of the CMS actor is by means of the navigational requirement *"Blind review process"*. We can observe in the SR model that some navigational and service requirements are decomposed in other requirements, some of them affects positively or negatively some NFRs (sofgoals hereinafter), i.e., the service requirement *"Download paper without authors' name"* needs the content requirement *"Papers"*, also, affects positively the softgoal *"Privacy be maximized"* and in some negatively form the softgoal *"Obtain more complete info"*. This fact is very important to see how to satisfy the goal *"Process of review of papers be selected"* considering the Web application softgoals.

4 Optimizing NFRs in Web Applications

In this section, we present a proposal (see Figure 2) which provides support for the stakeholder in order to evaluate and decide which functional requirements have to be implemented to improve the Web application functionality while NFRs are maximizing. To do this, we extend the goal-oriented requirement approach for Web engineering, described in Section 3, with the Pareto front algorithm, in order to achieve the optimal configuration based on the softgoals maximization. Also, a set of steps in order to fully satisfy the stakeholder goals have been defined. Therefore, the effect of the implementation of a set of FR can be assessed and the best among several implementation options can be selected by prioritizing the softgoals while still satisfying the goals.

The Pareto front algorithm is useful when there are multiple competing and conflicting objectives [10] that need to be balanced. The Pareto front is a notion

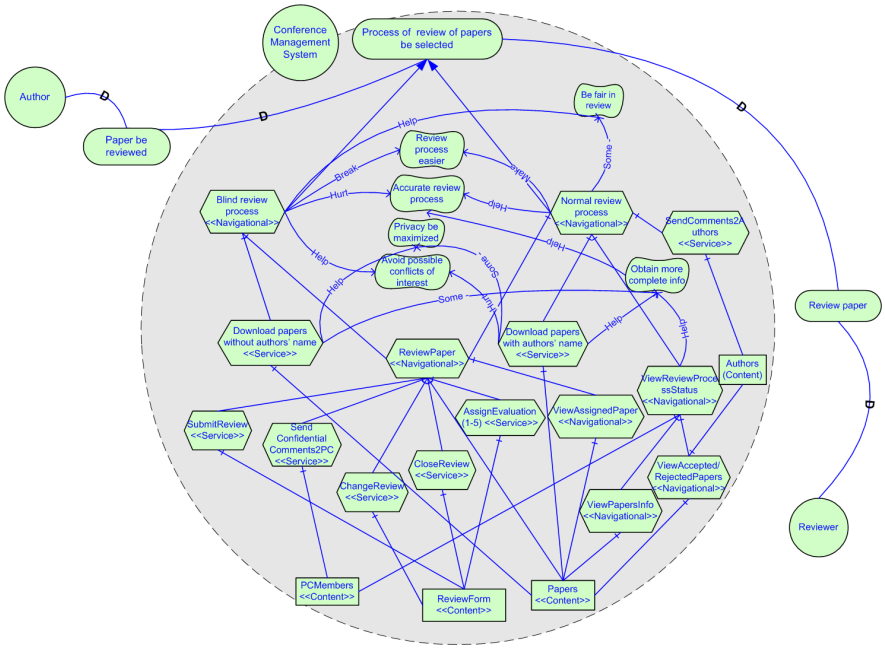


Fig. 1. Part of the Conference Management System requirements expressed in a SR and SD Models

from economics widely applied to engineering, which is described as follows: *"given a set of alternative allocations and a set of individuals, allocation A is an improvement over allocation B only if A can make at least one individual better than B, without making any other worse"*. In this sense, a set of individuals refers to the set of requirements, also, a set of alternative allocations corresponds to the state of the requirement (implemented or not implemented), and make an individual better by means of maximizing softgoals, and the opposite, means weakening softgoals. Therefore, a Pareto front, is one that no other configuration better satisfies a single softgoal, while satisfying the others equally. The set of Pareto configurations can be used to make a well-informed decision about which requirements configuration is the optimal to balance the tradeoff between softgoals.

Finding the set of Pareto optimal configuration can be defined as the problem of finding a (decision) vector of decision variables X (i.e., a valid implemented/not implemented requirements configuration), which maximizes a vector of M objective functions $f_i(X)$ (i.e., the satisfaction of softgoal i in configuration X) where $i = 1..M$ (with M the amount of softgoals). To do so, the concept of domination between vectors is defined as follows: a decision vector X is said to dominate a decision vector Y (also written $X \succ Y$) if and only if their corresponding objective vectors of objective functions $f_i(X)$ and $f_j(X)$ satisfies: $\forall i \in \{1...M\} f_i(X) \geq f_i(Y)$ and $\exists i \in \{1...M\} f_i(X) > f_i(Y)$, it is then said that

all decision vectors that are not dominated by any other decision vectors form the Pareto optimal set, while the corresponding objective vectors are said to form the Pareto front. Our approach as a running example is described next.

4.1 The Pareto Algorithm as a Running Example

Following the Pareto efficiency, we have defined the following steps to determine the Pareto optimal configuration for requirements implementation while the softgoals are balanced and maximized (see Figure 2). At the same time, these steps are applied in our running example presented in Section 3.

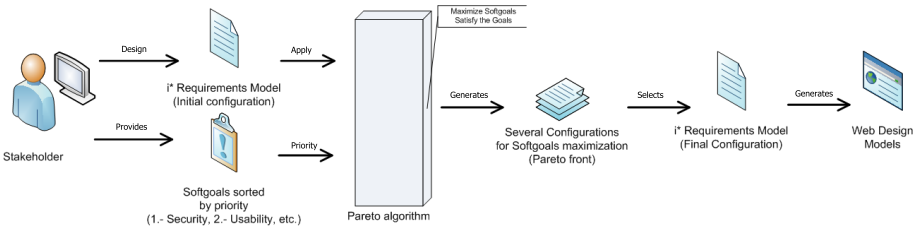


Fig. 2. Overview of the Pareto approach

Step 1. Create the Initial Requirements Model. The stakeholder creates an initial requirements model (IRM) using the *i** framework with which specifies the goals, softgoals and functional requirements (*Navigational*, *Service*, *Personalization*, *Layout* and *Content*) that the Web application must satisfy. This IRM specifies the requirements that must be implemented as a first prototype of the final Web application. At the same time, the stakeholder defines a list of softgoals sorted by priority, with which specifies the softgoals that the Web application must accomplish. For this running example, the requirements model is the one described in Section 3.

Step 2. Create a Requirements List. The second step consists of developing a list of the requirements (implemented or not) that contribute to any softgoal in the *i** requirements model (see Table 1). This table shows a requirements list and their type of contributions to the softgoals, where "S1" corresponds to softgoal *"Be fair in review"* from requirements model, "S2" to *"Review process easier"*, "S3" represents *"Accurate review process"*, "S4" conforms to *"Privacy be maximized"*, "S5" *"Avoid possible conflicts of interest"* and "S6" it is the *"Obtain more complete info"*.

Step 3. Store Each Possible Requirements Configuration. In this step, each possible implementation (configuration) of N requirements is stored in a decision vector X_v: $\forall v0 \preceq v < 2^N$, $\forall i \in \{1...N\} X_{v_i} = T_i$, where X_{v_i} is the *ith* element of X_v, $T_i = I$ if the requirement is implemented and $T_i = N$ if the requirement is not implemented.

Table 1. The requirements contributions to softgoals

Requirements	*"S1"*	*"S2"*	*"S3"*	*"S4"*	*"S5"*	*"S6"*
R1.- "Blind review process"	Help	Break	Hurt	-	Help	-
R2.- "Download papers without authors' name"	-	-	-	Help	-	Some -
R3.- "Normal review process"	Some -	Make	Help	-	-	-
R4.- "Download paper with authors' name"	-	-	-	Some -	Hurt	Help
R5.- "View review process status"	-	-	-	-	-	Help

Step 4. Assign a Weight to Each Contribution from Requirements to Softgoals. The contribution of each requirement (implemented or not) must be quantified. To this aim, the stakeholder creates a matrix by using the following weights to each kind of contribution: w= 0 if the requirement does not contribute to any softgoal, w= +1 if there is a Help contribution link, w= -1 if there is a Hurt contribution, w= +2 if there is a Some + link, w= -2 if the contribution is Some -, w= +4 if there is a Make and w= -4 if there is a Break contribution link.

Therefore, the matrix is defined, so that each entry W_{ij}^{k} corresponds to the contribution of the ith requirement to the jth softgoal on the k status (implemented or not): $\forall i \in \{1...N\}$, $\forall j \in \{1...M\}$, $\forall k \in \{I, N\}$ $W_{ij}^{k} = w$, where N is the number of requirements and M is the number of softgoals, and w is defined as previously described.

For computing the objective functions in this running example, the following matrix (1) is defined containing the quantification of each requirement to softgoals, as explained in the previous section. As an example, row 3, (requirement "Normal review process"), column 2 (softgoal "Review process easier"), shows "+4" in the matrix, indicating a "Make" contribution if the requirement is implemented, and on the other side, if the requirement "Blind review process" (row 1) is implemented, column 2 will be indicating "-4" in the matrix, this means a "Break" contribution.

$$M_{ij}^{k} = \begin{pmatrix} +1 & -4 & -1 & 0 & +1 & 0 \\ 0 & 0 & 0 & +1 & 0 & -2 \\ -2 & +4 & +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 & -1 & +1 \\ 0 & 0 & 0 & 0 & 0 & +1 \end{pmatrix} \quad (1)$$

Step 5. The Objective Function. For each softgoal j the corresponding objective function F_j with respect to a decision vector X_v is calculated by summing the contributions of all requirements to each softgoal j taking into account the requirements configuration defined in x_v: $\forall j \in \{1...M\}$, $\forall v0 \preceq v < 2^N$ $F_j(X_v) = \Sigma_{j=1}^{M} W_{ij}^{k}$, where N is the number of requirements, M is the number of softgoals.

Finally, the sum of all objective functions with respect to a decision vector X_v is computed to obtain the overall fitness of the decision vector X_v: $\forall j \in \{1...N\}$, $\forall v0 \preceq v < 2^N$ $\Sigma_{j=1}^{M} F_j(X_v)$, where N is the number of requirements and M is the number of softgoals.

Table 2 shows all possible decision vectors (column 2 to 6, all rows), in other words, all possible requirements configurations, where "I" represents the status "Implemented" and "N" represents "Not implemented". The results of the corresponding objective functions are shown in columns 7 to 12, and the overall

Table 2. The posible requirements to implement or not for the softgoal tradeoff

Configuration	R1	R2	R3	R4	R5	F(S1)	F(S2)	F(S3)	F(S4)	F(S5)	F(S6)	Pareto front
X1	I	I	I	I	I	-1	0	0	-1	0	0	No
X2	I	I	I	I	N	-1	0	0	-1	0	-1	No
X3	I	I	I	N	I	-1	0	0	1	1	-1	Yes
X4	I	I	I	N	N	-1	0	0	1	1	-2	No
X5	I	I	N	I	I	1	-4	-1	-1	0	0	Yes
X6	I	I	N	I	N	1	-4	-1	-1	0	-1	No
X7	I	I	N	N	I	1	-4	-1	1	1	-1	Yes
X8	I	I	N	N	N	1	-4	-1	1	1	-2	No
X9	I	N	I	I	I	-1	0	0	-2	0	2	Yes
X10	I	N	I	I	N	-1	0	0	-2	0	1	No
X11	I	N	I	N	I	-1	0	0	0	1	1	Yes
X12	I	N	I	N	N	-1	0	0	0	1	0	No
X13	I	N	N	I	I	1	-4	-1	-2	0	2	Yes
X14	I	N	N	I	N	1	-4	-1	-2	0	1	No
X15	I	N	N	N	I	1	-4	-1	0	1	1	Yes
X16	I	N	N	N	N	1	-4	-1	0	1	0	No
X17	N	I	I	I	I	-2	4	1	-1	-1	0	Yes
X18	N	I	I	I	N	-2	4	1	-1	-1	-1	No
X19	N	I	I	N	I	-2	4	1	1	0	-1	Yes
X20	N	I	I	N	N	-2	4	1	1	0	-2	No
X21	N	I	N	I	I	0	0	0	-1	-1	0	No
X22	N	I	N	I	N	0	0	0	-1	-1	-1	No
X23	N	I	N	N	I	0	0	0	1	0	-1	Yes
X24	N	I	N	N	N	0	0	0	1	0	-2	No
X25	N	N	I	I	I	-2	4	1	-2	-1	2	Yes
X26	N	N	I	I	N	-2	4	1	-2	-1	1	No
X27	N	N	I	N	I	-2	4	1	0	0	1	Yes
X28	N	N	I	N	N	-2	4	1	0	0	0	No
X29	N	N	N	I	I	0	0	0	-2	-1	2	Yes
X30	N	N	N	I	N	0	0	0	-2	-1	1	No
X31	N	N	N	N	I	0	0	0	0	0	1	Yes
X32	N	N	N	N	N	0	0	0	0	0	0	No

fitness for each decision vector is shown in column 13. Finally, in the last column, we indicate if the corresponding decision vector is in the Pareto front. Grey rows are the Pareto front.

Step 6. Maximize the Softgoals and Still Satisfaying the Goals. In this step the stakeholder creates a list of softgoals sorted by priority (the softgoals priority was stablished in the list from *Step 1* by the stakeholder) and a list of goals that the Web application has to achieve. For this case, the softgoals priority list is shown in Table 3.

Table 3. Softgoals priority list for achieve the goal *"Process of review of papers be selected"*

Order	Softgoal
1	*"S4.- Privacy be maximized"*
2	*"S2.- Review process easier"*
3	*"S3.- Accurate review process"*
4	*"S1.- Be fair in review"*
5	*"S5.- Avoid possible conflicts os interest"*
6	*"S6.- Obtain more complete info"*

Step 7. Select the Pareto Optimal Configuration. Finally, according to Table 3, the two most important softgoals to maximize are "S4" and "S2". Therefore, it is necessary to select the final solution according to the priorities established over the softgoals.

First of all, it is necessary to select the configurations that besides being Pareto front satisfy the goal *"Process of review of papers be selected"*, for this running example, the configurations "X3", "X7", "X17" and "X25" are the only ones that satisfy the goal from all the 14 configurations that are Pareto front (see Table 2). Importantly, this step could be done in *Step 5*, i.e., calculating the

objective function for those configurations that satisfy the goals, but not doing it in this form allows us to select between different configurations considering only the softgoals maximization, leaving aside the *goals*, this gives a wider scope to the stakeholder for the final implementation.

The next step consists in selecting from the configurations that are Pareto front and satisfy the goal, the ones that maximize the softgoals according with the list from Table 3. To do this, it is necessary to check all the configurations with the requirements model to select the configurations that allow to achieve the goal (in this case there are two paths, i.e., two means-ends links), these are "X3", "X7", "X17" and "X25" . Then, it is necessary to select the best option according to the softgoals to maximize. For the softgoal "S4", "X3" and "X7" are the configurations which its overall is maximized and, for the softgoal "S2" are "X17" and "X25".

For this running example, the configuration "X3" is the best option, because according with the priority list, "S4" and "S2" are the softgoals to prioritize. The configurations "X17" and "X25" maximize "S2", however the contributions of both to softgoal "S4" (which is the number one from the priority list) are -1 and -2 (see Table 2). Furthermore, besides that the configuration "X3" has an overall fitness of $+1$ for "S4" as same as the configuration "X7", the configuration "X3" has an overall fitness of 0 for "S2" and, "X7" has an overall fitness of -4 for "S2", resulting more affected that the configuration "X3" (see Table 2), with which indicating that optimizing security comes at a high cost with respect to other softgoals (usability). The rest of solutions of the Pareto front are intermediate configurations that lead us to different tradeoffs.

Finally, the final requirements model (FRM) is the configuration "X3" (see Table 2). Therefore, the requirements "R1.- Blind review process", "R2.- Download papers without authors name", "R3.- Normal review process" and "R5.- View review process status" must be implemented in order to maximize the softgoals "S4.- Privacy be maximized" and "S2.- Review process easier". In "X3" only "R4" is not implemented. These requirements enable alternative paths (means-ends links) to satisfy the goal.

5 Conclusion and Future Work

In this work, we have presented an extension to our goal-oriented requirements analysis approach for the development of Web applications. Our approach allows the stakeholder to evaluate and decide which requirements configuration to implement. To do so, we devised an algorithm based on the Pareto optimal approach that is particularly suited to balance and maximize the conflicting softgoals. Furthermore, it facilitates the evaluation of the obtained (Pareto) optimal solutions and the selection of the final solution taking into account the priorities of softgoals. To do this, it includes weighted contributions according to the importance of softgoals, in order to further help the stakeholder to balance and optimize the different softgoals. Future work consists in the integration of

our goal-oriented approach for requirements analysis and the Pareto algorithm in a MDD solution for the development of Web applications, within the A-OOH approach.

Acknowledgments. This work has been partially supported by the MANTRA project (GV/2011/035) from the University of Alicante, and by the MESOLAP (TIN2010-14860) from the Spanish Ministry of Education and Science. José Alfonso Aguilar is subventioned by CONACYT (Consejo Nacional de Ciencia y Tecnología) Mexico and University of Sinaloa, Mexico.

References

1. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: An MDA Approach for Goal-Oriented Requirement Analysis in Web Engineering. J. Univ. Comp. Sc. 16(17), 2475–2494 (2010)
2. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: Web Engineering Approaches for Requirement Analysis- A Systematic Literature Review. In: 6th Web Information Systems and Technologies (WEBIST), vol. 2, pp. 187–190. SciTePress Digital Library, Valencia (2010)
3. Ameller, D., Gutiérrez, F., Cabot, J.: Dealing with Non-Functional Requirements in Model-Driven Development. In: 18th IEEE International Requirements Engineering Conference (RE), pp. 189–198. IEEE, Los Alamitos (2010)
4. Bolchini, D., Paolini, P.: Goal-Driven Requirements Analysis for Hypermedia-Intensive Web Applications. J. Req. Eng. 9(2), 85–103 (2004)
5. Escalona, M.J., Koch, N.: Requirements Engineering for Web Applications - A Comparative Study. J. Web Eng. 2(3), 193–212 (2004)
6. Garrigós, I., Mazón, J.N., Trujillo, J.: A requirement analysis approach for using i* in web engineering. In: Gaedke, M., Grossniklaus, M., Díaz, O. (eds.) ICWE 2009. LNCS, vol. 5648, pp. 151–165. Springer, Heidelberg (2009)
7. Gupta, C., Singh, Y., Chauhan, D.S.: Dependency Based Process Model for Impact Analysis: A Requirement Engineering Perspective. J. Comp. App. 6(6), 28–30 (2010)
8. Molina, F., Toval, A.: Integrating Usability Requirements that can be Evaluated in Design Time into Model-Driven Engineering of Web Information Systems. J. Adv. Eng. Softw. 40, 1306–1317 (2009)
9. Sommerville, I.: Software Engineering, 6th edn. Addison-Wesley, Reading (2001)
10. Szidarovszky, F., Gershon, M., Duckstein, L.: Techniques for Multiobjective Decision Making in Systems Management. Elsevier, Amsterdam (1986)
11. Yu, E.S.K.: Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering. In: 3rd IEEE International Symposium on Requirements Engineering (RE), p. 226. IEEE, Washington, DC, USA (1997)

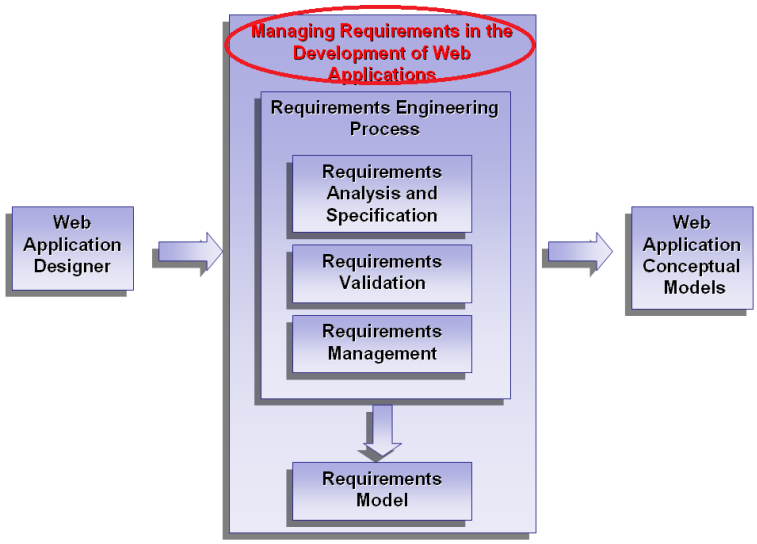
Parte III

Apéndice: Artículos enviados

A

Requirements in Web engineering: a systematic literature review.

El contenido de este apéndice ha sido enviado a Journal of Web Engineering (JWE)



El artículo presentado en este apéndice es una extensión del Capítulo 2 de la tesis doctoral en el que se destaca la importancia de considerar a los requisitos en el desarrollo de sistemas Web. En particular, en la revisión sistemática de la literatura presentada en el apéndice, se ha mejorado el trabajo descrito en el Capítulo 2 de la siguiente forma:

- La estrategia de búsqueda ha sido mejorada, por lo tanto se han añadido más métodos a la revisión.
- El estudio se ha centrado en analizar el proceso de ingeniería de requisitos en el desarrollo de aplicaciones Web, es decir, en qué forma los requisitos son tratados en lo que respecta al análisis, especificación, validación y la gestión de los mismos.
- La revisión sistemática analiza el vocabulario que ha sido adoptado por cada método de ingeniería Web de una manera metódica y completa mediante el uso de la clasificación propuesta por Escalona y Koch [25].

Journal of Web Engineering, Vol. 0, No. 0 (2003) 000–000

REQUIREMENTS IN WEB ENGINEERING: A SYSTEMATIC LITERATURE REVIEW

JOSÉ ALFONSO AGUILAR; IRENE GARRIGÓS; JOSE-NORBERTO MAZÓN
Lucentia Research Group, DLSI, University of Alicante
San Vicente del Raspeig, 03080, Spain.
ja.aguilar@dlsi.ua.es; igarrigos@dlsi.ua.es; jnmazon@dlsi.ua.es

Received (received date)
Revised (revised date)

Requirements engineering is one of the most crucial phases in software development in which designers can attempt to fully satisfy users' needs, and an effective definition of requirements both contributes towards making the right design decisions and helps to support change management, thus improving the quality of the final software product. Web engineering methods should consider a requirement engineering phase that suits the Web's large heterogeneous user community. The objective of this work is to classify the literature with regard to the requirements engineering applied in the Web domain, with which has allowed us to formally obtain the current state-of-the-art. The present work is based on the systematic literature review method proposed by Barbara Kitchenham, we reviewed publications from ACM, IEEE Computer Society Digital Library, Science Direct, DBLP and World Wide Web. From a population of 3059 papers, we identified 43 primary studies that provide information about dealing with requirements in Web engineering.

Keywords: Web Engineering, Requirements Engineering, Systematic Literature Review
Communicated by: to be filled by the Editorial

1 Introduction

A Web system is an application that is invoked with a Web browser commonly over the Internet and it differs from traditional stand-alone software in that their user community is both large and heterogeneous. In this respect, Web applications are designed to be widely accessed, and their may have therefore different needs, goals and preferences. Furthermore, a Web system must satisfy the needs of many types of stakeholders[a]apart from the users themselves, i.e., the persons who maintain the system; the organization wishes the system; and those who fund the system development budget.

Consequently, several special requirements must be considered in the development of Web systems: (i) what information the Web application should offer (content requirements), (ii) which scenarios should be defined in the Web application in order to offer this information (navigational requirements), and (iii) how the user or groups of users should be provided with this information (functional requirements). As a result, the development of a Web system calls

[a]Stakeholders are individuals or organizations who affect or are affected directly or indirectly by the development project in a positive or negative form [1]

for knowledge and expertise from many different disciplines and requires a team of diverse group of people with a high degree of expertise in different areas [2]. This makes the design and development of the Web system further complex and difficult. In this respect, Web engineering methods [3, 4, 5, 6, 7] have provided different mechanisms with which to consider the content, composition, and navigation features of Web applications; including appropriate steps to consider requirements [8].

Bearing these considerations in mind, this paper presents a systematic literature review [9] with which to analyse the current state-of-the-art with regard to requirements engineering in the Web domain in order to reveal their advantages and disadvantages. Roughly speaking, a systematic literature review is useful for formally defining several important issues to obtain those relevant papers related to a research topic: (i) the problem to be addressed, (ii) the information sources, (iii) keywords to search for information, (iv) the inclusion and exclusion criteria to be applied to the papers found in the searches; and (v) the templates used to order and classify the information collected from the papers found. This research technique originated in the field of medical research and was successfully adapted to software engineering by Barbara Kitchenham [9][b]

This paper is an extension of our previous review [17] in which we highlighted the importance of considering requirements in the development of Web systems. In particular, in the present systematic literature review we have improved our previous work as follows.

- The search strategy has been improved, and more methods have been therefore added to the review.

- The study has been focused on analyzing the requirements engineering process in the Web systems development, specifically, in which form the requirements are treated with regard to elicitation, analysis, specification, validation and the management of them.

- How the studied methods have been spreaded in the academic community has been shown.

- Our systematic review analyzes the requirements vocabulary adopted by each Web engineering method in a more methodical and comprehensive manner by using the classification previously proposed by Escalona and Koch [8].

In summary, the aim of this review is to: (i) structure the conceptual basis to deal with the requirements engineering process in the development of Web systems, and (ii) shed light on potential avenues for future research in this area.

The remainder of this paper is structured as follows: Section 2 presents those requirements engineering and Web engineering concepts which are relevant to the context of this paper. The systematic literature review is detailed in Section 3. The analysis and discussion of this work and our suggestions for future research are presented in Section 4 and Section 5, respectively. Finally, our conclusions are provided in Section 6.

[b] It is worth noting that the approach followed for conducting the systematic literature review presented in this paper has been inspired by others such as [10, 11, 12, 13, 14, 15, 16]

2 Requirements and Web engineering concepts

Requirements engineering is the process of discovering, analysing, documenting and verifying the services that should be provided by a software system, along with its operational constraints. According to [18], a requirements engineering process can be divided into five steps:

- **Requirements elicitation.** This is the initial step in requirements engineering whose objective is to discover what problem needs to be solved. The main goals of this step are the identification of the *stakeholders*, the objectives a software system must meet, the tasks that users currently perform and those that they might wish to perform. Requirements elicitation is often carried out through the application of various techniques [19, 20, 21], such as (i) traditional techniques (questionnaires, interviews, etc.), (ii) group elicitation techniques (brainstorming or focus groups), (iii) prototyping (e.g., provoking discussion in a group elicitation), (iv) modelling techniques (e.g., goal-based methods such as *i** [22]), (v) cognitive techniques (specially developed for knowledge acquisition), and (vi) contextual techniques (e.g. the use of ethnographic techniques such as participant observation).

- **Requirements analysis.** This step can include the creation of conceptual models or prototypes with which to achieve the completeness of the requirements. Requirements analysis deals with understanding an organization's structure, its business rules, and the goals, tasks and responsibilities of its members and the data that is needed. This stage allows designers to detect and resolve conflicts that may occur among different *stakeholders* [23].

- **Requirements specification.** This step is an integral description of the behavior of the system to be developed. This description aims to facilitate an effective communication of requirements among different *stakeholders*. The most widely used techniques for requirements specification are, among others, templates, scenarios, use case modelling, and natural language [24].

- **Requirements validation.** This step aims to establish whether the requirements and models elicited provide an accurate representation of the actual *stakeholder* requirements. Some techniques for requirements validation are reviews, audit and traceability matrices. A requirement must be traceable, since it is defined throughout the development process. A traceable requirement ensures that change management is able to assess the impact of a change on the rest of the application. It is therefore necessary for *stakeholders* to be able to see that the final *work product*[a]is useful for achieving their requirements, and that each requirement has been derived within a particular work product. To do this, requirements engineering needs to ensure traceability. Requirements traceability refers to the ability to describe and follow the life of a requirement, in both a forward and a backward direction [25]. While forward traceability is related to following the requirement to its final implementation, backward traceability refers to following the *work product* to its source requirement (i.e., the associated requirement

[a]A *work product* represents all the documents and models produced through a software development process.

that originated it). Interestingly, traceability between requirements and *work products* is a recommendation made by the CMMI (Capability Maturity Model Integration)[b], specifically in CMMI's Level 2 (within the requirements management process area).

- **Requirements management.** This step consists of recognizing changes through continuous requirements elicitation, the re-evaluation of risks, and the evaluation of systems in their operational environment. Requirement management includes techniques and tools for configuration management and version control [26].

After an overview of requirements engineering concepts, it is worth noting that the development of Web applications have some particular requirements that differ from the traditional requirements. These new requirements are defined in the seminal work of Escalona and Koch [8]. In this work, the authors put forward the argument that functional requirements for Web engineering are related to three main features of Web applications: navigational structure, user interface and personalization capability, and that the data structures required by the Web application should also be specified. An overview of each kind of requirement for Web engineering is described below:

- **Content Requirements.** These define the information that is useful for the Web application which should be presented to users. For example, in an online book store some examples might be the information about a "book" or a "book category".

- **Service Requirements.** This refers to the internal functionality that the Web application should provide to its users. Following the online book store example, service requirements might be: "register a new client" or"add book to shopping cart".

- **Navigational Requirements.** A Web application must also define the navigational paths which are available. In this respect, some examples are: the user navigation from "index page" to different options such as"consult products by category" or "consult shopping cart".

- **Layout Requirements.** Requirements can also define the visual interface for users, such as "present a colour style", "multimedia support", or "user interaction", among others.

- **Personalization Requirements.** The designer should specify the desired personalization actions to be performed in the final Web application (e.g., "show recommendations based on interest in previously acquired books", "adapt font for visually impaired users", etc.).

- **Non-Functional Requirements.** These are related to quality criteria that the intended Web application should achieve and that may be affected by other requirements. Some examples might be "good browsing experience", "attract more users", "improve efficiency", etc.

[b] CMMI provides organizations with the essential elements for the effective improvement of a process. CMMI is a trademark owned by the Software Engineering Institute of Carnegie Mellon University.

This classification of requirements for Web engineering is used throughout this systematic literature review for the sake of understandability and completeness.

These types of requirements are considered in the Web engineering methodologies for the definition of several models. Web engineering approaches commonly use five main models to define a Web application [27], namely:

- **Domain model (DM).** This encapsulates the structure and functionality required of the relevant concepts of the application domain and reflects the static part of the Web application.
- **Navigation model (NM).** This model aims to specify the structure and behavior of the navigation view over the domain data defined. It defines each path on which users can navigate through the Web application.
- **Presentation model (PM).** This model defines the layout of the Web application, i. e. style, font color, etc.
- **Personalization model (PM).** Personalization strategies (Adaptive Web Sites) are specified in this model. Web pages are personalized based on the context, the socioeconomic level and the interest of an individual user. Personalization implies that the changes are based on the items purchased or in the pages viewed.
- **User model (UM).** This enables the users to be described in terms of their personal information and their relations with a particular application domain as well as the navigational actions performed at execution-time. The structure of the information needed for the personalization strategies is also described in this model.

3 Reviewing literature about requirements engineering in Web domain

This study has been undertaken as a systematic literature review (SLR) based on the original guidelines proposed by Kitchenham [9]. A systematic literature review is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, topic area or phenomenon of interest. The goal of the review is to analyse the current state-of-the-art with regard to requirements engineering in the Web domain. Although, there are several reasons for performing a systematic literature review, the most common are:

- To sum up the remaining data (information) concerning a treatment of technology.
- To identify any gaps in research in progress, thus highlighting areas of potential interest for investigation.
- To serve as a framework in which to properly place new research activities.

A systematic literature review consists of three main phases, namely:

- **Planning the review.** This aims to develop a protocol that specifies the plan that the systematic literature review will follow to identify, assess and accumulate evidence.
- **Conducting the review.** This phase is responsible for achieving the protocol planned in the previous phase.

- **Reporting the review.** This phase is responsible for extending the review to the research community. It culminates with the elaboration of a review report.

Each of these phases is made up of a sequence of steps in order to be applied iteratively, thus simultaneously refining them. These steps are as follows:

- Planning the review.
  - Constructing the research questions that will guide the analysis of the research.
- Conducting the review.
  - Defining a search strategy.
  - Defining the inclusion and exclusion criteria for the studies.
  - Study quality assessment.
  - Study selection.
  - Data extraction.
  - Analysis of the primary studies.
- Reporting the review.
  - Elaborating a review report (extending the results to others).

The steps performed in our systematic literature review are based on the proposal from [28], these are detailed next.

3.1 Research questions

As mentioned in Section 1, the goal of this systematic literature review is to summarize the information concerning how requirements engineering process is applied in Web engineering, thus detecting avenues for future research. According to the systematic review process [9], the question structure is divided in four aspects (PICO):

- **Population:** people, projects and application types affected by the intervention. They should be directly related to the question.
- **Intervention:** software technology, tool or procedure, which generates the outcomes.
- **Comparison:** some other type of intervention - only if applicable -
- **Outcomes:** technology impact in relevant information terms for practical professionals.

Based on this strategy, the PICO structure for the research questions is described below:

- **Population:** this question refers to the support of the methods to the Web application development process. The population is composed by designers and developers seeking a way to have a more robust process support and by researchers in Web engineering area aiming at the development of new methods.

- **Intervention:** this review must search for indications that the development of Web application can be fully supported by a systematic process.

- **Comparison:** no applicable.

- **Outcomes:** the objective of this study is to demonstrate how the development of Web applications is supported by a systematic process and if the process is not fully supported with regard to requirements engineering area, since there are many gaps in the development of Web applications.

As a result, four research questions (RQ) should be asked, focusing on specific aspects of the evaluation. These are defined next:

1. **(RQ1) Which are the existing methods for the development of Web applications based on a systematic process?** There are several Web engineering approaches for the development of Web applications, but not all of them necessarily cover requirements engineering. It is therefore necessary to detect to what extent different Web engineering approaches cover the requirements engineering phase.

2. **(RQ2) What are the phases of the requirement engineering process of each method?** The main features, advantages and drawbacks of requirements engineering techniques defined for or adapted to the Web engineering field are studied.

3. **(RQ3) Is there any common terminology with regard to requirements engineering applied by the existing methods?** This research question refers to detect the type of requirements defined for each Web engineering method with regard to requirements phase.

4. **(RQ4) Which tools supports the development process?** The main idea of this research question is to analyse those Web engineering methods that offer tools in order to verify the requirements engineering support.

The next step in the systematic literature review is to determine and follow a search strategy. The objective of the search strategy is to create the right search statements with which to answer the research questions. The search strategy defined in the present systematic literature review is described as follows.

3.2 Search strategy

The search strategy should be systematic, comprehensive and explicit in order to make a formal search of primary studies with a predetermined order. According to [9], in this phase it is necessary to use the search engines by applying a combination of search terms (keywords) extracted from the research questions. Various experts should then verify and review the search results. Once the steps to be followed in the search process have been defined, it is necessary to define the resources that are available to conduct the review of primary studies. In this systematic literature review the research sources used are as follows:

- **eJournals (repositories with restricted access):** ACM,[c] IEEE Computer Society Digital Library,[d] Science Direct.[e]
- **Digital library of scientific literature:** DBLP Computer Science Bibliography.[f]
- **Research resources in the World Wide Web:** Google Scholar.[g]

According to [29], these libraries were chosen because they are some of the most relevant sources in software engineering (see appendix). On the other hand, Google scholar was selected to complete the set of conferences and workshops searched and to seek grey literature in the field (i.e., white papers, technical reports and PhD thesis).

After defining the research questions and the sources to be consulted, the strings to be used in the search were then defined. Based on the structure of the research questions, some keywords were extracted and used to search the primary study sources. We initially had the following keywords: *Web*, *engineering*, *requirements*, *development*, *method* and *tool*. However, in order to obtain more concrete and specific results in the field, we decided to link Web with the keywords *engineering* and *requirements*, *requirements* with the keyword *engineering*, and the keyword *Web* with the keywords *engineering* and *methods*. In this respect, the choice to concatenate "Web" with "engineering" was motivated by our goal, which was to retrieve papers specifically focused on requirements engineering in the Web domain. As a result, the search string "(Web OR WWW OR World-Wide Web OR Internet) AND engineering" was not considered. The search string was used in all instances, even when examining papers from special issues in Web engineering. Also, a list of synonyms was constructed for each of these keywords. Nevertheless, other words were also added to increase the size of potential relevant studies: *system*, *techniques*, *phase*, *design*. These words were linked with the keywords *Web*, *requirements*, *methods*, *engineering* and *tool*.

The search string was thus constructed by using the boolean operators (AND and OR): (*Requirement* OR *requirements* OR *requirements engineering* Or *Web engineering*) OR ((*Web requirements*) OR (*Web system*)) AND (*method* OR (*techniques*)) AND (*Tool support*). However, the results from this search string were too imprecise in the case of research sources such as IEEE and ACM because of by using their search engines were retrieved a few papers. Bearing this consideration in mind, we decided to adapt the search keywords to match each research question and the individual requirements of each of the search engines on our search resources. Based on the SLR proposed by [30], we created a specific type of string for each search engine (using the synonyms list) making the search as accurate and comprehensive as possible.

Moreover, corresponding authors of main texts were e-mailed directly to find out some particular issue about their Web method. Finally, it is important to highlight the fact that we conducted a secondary search based on the references found in our primary studies in order to ensure that we do not leave aside any major work in this area. This idea is based on the SLR performed by [31].

[c] http://portal.acm.org
[d] http://ieeexplore.ieee.org
[e] http://www.sciencedirect.com
[f] http://www.informatik.uni-trier.de/ ley/db/index.html
[g] http://scholar.google.com

3.3 Inclusion and Exclusion criteria

In this phase, the inclusion and exclusion criteria were defined. These criteria were used, respectively, to both identify potentially relevant studies and exclude non-relevant studies from the primary study search results.

Three inclusion criteria were defined as follows: the first consisted of reading the title and abstract of the primary studies. Documents were considered to be sufficiently relevant if some of the terms used in the search strings appeared in the title or the abstract.

The second inclusion criterion consisted of reading the introduction and conclusion to those primary studies dealing with a specific requirements engineering issues. Lastly, the third criterion involved reading the whole document to discover those primary studies related to requirements engineering process.

Basically, this systematic literature review included any primary studies related to the requirements engineering in the Web field. In this respect, we deemed that at least the part related to requirements specification in Web engineering must be present in each primary study, since we assumed that not all approaches implement another phase of the requirements engineering process such as validation or management.

With regard to the exclusion criteria, those primary studies whose topic did not match with the requirements engineering in Web domain were excluded, along with duplicated documents of the same study, short papers and tutorials for the development of Web applications, since they would have made a limited impact and contribution.

3.4 Study quality assessment

The quality assessment of the selected studies was performed using as indicators the publication place and the diffusion of the methods. The publication place refers to the journals and conferences where the primary studies were published (this applies to Google Scholar which go in search of a wider spectrum of papers such as white papers). The diffusion of the methods corresponds to the academic or professional application of the method, also if the method has a tool support and if the tool is a prototype or a professional tool.

3.5 Study selection

The search for primary studies was conducted by using the search strategy defined in Section 3.2. In this phase, the *Title*, *Author*, and *Year* of the paper were used to keep track of the information extraction after the first exclusion. Table 1 shows the results of the search.

Table 1. Search results by source and exclusion criteria.

	ACM	**IEEE**	**Science Direct**	**DBLP**	**WWW**	**Total**
Found	416	234	294	44	2071	3059
Duplicates	19	0	5	1	45	70
1st Exclusion	19	11	6	6	35	77
2nd Exclusion	11	5	3	3	21	43
3rd Exclusion	5	2	3	0	3	13

The selection of primary studies was addressed by using the exclusion and inclusion criteria. Table 1 shows the number of publications retrieved after applying the three exclusion

criteria, divided by the search source. The first search (without any exclusion criteria) returned a total of 3059 documents. The publications retrieved after applying the first and second exclusion criteria are shown in Tables 2, 3, 4, 5 and 6, separated by search sources (ACM, IEEE, Science Direct, DBLP and WWW). The publications selected when the third exclusion criterion was applied are highlighted in bold type.

Table 2. Search results in ACM after applying first and second exclusion criteria. Final relevant documents (after applying third exclusion criteria) are highlighted in bold type.

ACM

Title	Author	Year
A Model-Driven Architecture Approach Using Explicit Stakeholder Quality Requirement Models for Building Dependable Information Systems	Stefan Biffl, Richard Mordinyi, Alexander Schatten	2007
The object-oriented hypermedia design model.	**Daniel Schwabe,Gustavo Rossi**	**1995**
A Requirement Analysis Approach for Using *i in Web Engineering**	**Irene Garrigos, Jose-Norberto Mazon, Juan Trujillo**	**2009**
Model-driven in reverse. The practical experience of the AQUA project	M.J. Escalona, J.J. Gutirrez	2009
On the generation of requirements specifications from software engineering models: A systematic literature review	Joaqun Nicolas, Ambrosio Toval	2009
Transformation techniques in the model-driven development process of UWE	**Nora Koch**	**2006**
A Conceptual Modeling Approach for the Design of Web Applications based on Services	Ricardo Quintero, Victoria Torres, Marta Ruiz, Vicente Pelechano	2006
Requirements traceability in model-driven development: Applying model and transformation conformance	Joao Paulo A. Almeida, Maria-Eugenia Iacob	2007
Supporting Requirements in a Traceability Approach between Business Process and User Interfaces	Kenia Sousa, Hildeberto Mendona, Jean Vanderdonckt	2008
Model Transformations from Requirements to Web System Design	**Nora Koch, Gefei Zhang, María José Escalona**	**2006**
Introducing requirements traceability support in model-driven development of web applications	**Pedro Valderas, Vicente Pelechano**	**2008**

Table 3. Search results in IEEE after applying first and second exclusion criteria. Final relevant documents (after applying third exclusion criteria) are highlighted in bold type.

IEEE

Title	Author	Year
An analysis of the requirements traceability problem	Orlena C. 2. Gotel. Anthony C. W. Finkelstein	1994
A navigational Web requirements validation Trhough Animation	Joumana Dargham, Rima Semaan	2008
From Task-Oriented to Goal-Oriented Web Requirements Analysis.	**David Bolchini, John Mylopoulos**	**2003**
NDT. A Model-Driven Approach for Web Requirements	**María José Escalona**, Gustavo Aragn	**2008**
Impact Analysis from Multiple Perspectives: Evaluation of Traceability Techniques	Salma Imtiaz, Naveed Ikram, Saima Imtiaz	2008

The primary studies selected according to the search performed are shown in Table 7. A total of 13 relevant documents were found.

Table 4. Search results in Science Direct after applying first and second exclusion criteria. Final relevant documents (after applying third exclusion criteria) are highlighted in bold type.

Science Direct

Title	Author	Year
A scoped approach to traceability management	Patricia Lago, Henry Muccini, Hans van Vliet	2008
Using established Web Engineering knowledge in model-driven approaches	Ralf Gitzel, Axel Korthaus, Martin Schader	2006
Integrating usability requirements that can be evaluated in design time into Model Driven Engineering of Web Information Systems	**Fernando Molina, Ambrosio Toval**	**2009**

Table 5. Search results in DBLP after applying first and second exclusion criteria. Final relevant documents (after applying third exclusion criteria) are highlighted in bold type.

DBLP

Title	Author	Year
Conceptual Modeling for System Requirements Enhancement	Eric Le Pors, Olivier Grisvard	2009
From Conceptual Modeling to Requirements Engineering	Colette Rolland	2006
Requirements engineering for web applications-a comparative study.	**María José Escalona, Nora Koch**	**2004**

3.6 Data extraction

The goal of this phase is to design data extraction forms with which to accurately record the information obtained from the primary studies. The data extraction form must be designed to collect all the information required in order to fully address the research questions [9]. In this phase, the form shown in Table 8 was used to store the information extracted from the search results. This form contains the title of the publication, the journal or workshop in which the paper was published, the publication date, the main author, the requirements engineering process, the shortcomings with regard to the requirements engineering and the tool support.

At this stage of our systematic literature review, the data extraction of primary studies was performed by applying the different inclusion and exclusion criteria. Quality assessment was then performed separately to verify the information extracted. It is better to perform data extraction and quality assessment separately owing to the large quantity of items extracted.

Data synthesis was then performed. Data synthesis involves collating and summarizing the results of the primary studies included. The synthesis is descriptive (non-quantitative) [9] and is carried out by answering the research questions, as in this review (see questions presented in Section 3.1).

4 Analysis of the primary studies

This section presents and analyses the results obtained after performing the extraction and data synthesis phases on the primary studies. Several features (derived from the research questions) of Web engineering methods in the primary studies are described. These features are related to the requirements engineering process and the tool support. The different types of requirements used by each approach are also discussed (according to the classification described in Section 2).

Table 6. Search results in Google after applying first and second exclusion criteria. Final relevant documents (after applying third exclusion criteria) are highlighted in bold type.

WWW (Google Scholar)

Title	Author	Year
A Model Driven Approach for the Integration of External Functionality in Web Applications. The Travel Agency System	Nora Koch, Antonio Vallecillo, Gustavo Rossi	2005
Automatic Support for Traceability in a Generic Model Management Framework	Artur Boronat, Jos . Cars, and Isidro Ramos	2005
Hera: Development of semantic web information systems	**Geert-Jan Houben , Peter Barna , Flavius Frasincar , Richard Vdovjak**	**2003**
Metamodeling the Requirements of Web Systems	**Nora Koch, María José Escalona**	**2006**
Requirement Engineering with URN: Integrating Goals and Scenarios	Jean-Franois Roy	2007
Requirements Traceability and Transformation Conformance in Model-Driven Development	Joao Paulo Almeida, Pascal van Eck, Maria-Eugenia Iacob	2006
Applying Transformations to Model Driven Development of Web Applications	Santiago Meli and Jaime Gmez	2006
An MDA Approach for the Development of Web Applications	Santiago Meli Beigbeder and Cristina Cachero Castro	2004
Supporting Stakeholders in the MDA Process	Keith Phalp, Sheridan Jeary, Jonathan Vincent	2007
A Transformational Approach to Model Driven Engineering of Data-intensive Web Applications	Davide Di Ruscio, Henry Muccini, and Alfonso Pierantonio	2008
A Survey on Web Modeling Approaches for Ubiquitous Web Applications	Andrea Schauerhuber, Wieland Schwinger, Werner Retschitzegger, Manuel Wimmer	2002
Model driven architecture: Principles and practice	Alan W. Brown	2004
Weaving Business Requirements into Model Transformations	Ying Zou, Hua Xiao, Brian Chan	2007
ADM: Método de diseño para al generación de prototipos Web rápidos a partir de modelos	Susana Montero, Paloma Daz, Ignacio Aedo y Laura Montells	2006
From requirements to implementations: a model-driven approach for web development	Susana Montero,Paloma Diaz, Ignacio Aedo	2007
WSDM:a user centered design method for Web sites	**O. M. F. De Troyer, C. J. Leune**	**1998**
Web Modeling Language (WebML):a modeling language for designing Web sites	**Stefano Ceri, Piero Fraternali, Aldo Bongio**	**2000**
Extending UML for modeling Web applications	Luciano Baresi, Franca Garzotto, and Paolo Paolini	2001
THESIS A Requirements Engineering Approach for the Development of Web Applications	Pedro J. Valderas Aranda	2007
THESIS Model Driven Language Engineering	Octavian Patrascoiu	2005
THESIS Modelling Adaptive Web Applications in OOWS	Gonzalo Eduardo Rojas Durn	2008

4.1 UML-based Web Engineering (UWE)

UWE [43] is a methodological approach for the development of Web applications which is entirely based on UML. It covers the whole life cycle of the development of Web applications and focuses on adaptive applications.

Development process: UWE is based on Model-Driven Architecture (MDA) [44]. Within MDA, requirements are considered in the CIM (Computational Independent Model) in order

Table 7. Primary studies resulting after applying the third exclusion criteria.

Title
Metamodeling the requirements of Web systems [32].
Model transformations from requirements to web system design [33].
Requirements engineering for Web Applications-a comparative study [8].
Introducing requirements traceability support in model-driven development of web applications [34].
The object-oriented hypermedia design model. [35].
Integrating usability requirements that can be evaluated in design time into Model Driven Engineering of Web Information Systems [36].
From task-oriented to goal-oriented Web requirements analysis [37].
Transformation techniques in the Model-Driven Development Process of UWE [38].
NDT. A Model-Driven Approach for Web requirements [39].
A requirement Analysis Approach for Using i* in Web Engineering [27].
Web Modeling Language (WebML): a modeling language for designing Web sites [40].
WSDM: a user centered design method for Web sites [41].
Hera: Development of semantic web information systems [42].

Table 8. Data extraction form used in this systematic review.

Data Extraction Form	
Title	
Journal/Conference	
Date	
Main author	
Requirements engineering	
Requirements engineering shortcomings	
Tool support	

to show what the Web application is expected to do without showing details of how it is implemented. Through this requirements model, UWE can generate content, navigation, presentation and process models. Generation is done by clearly and formally establishing a set of Query/View/Transformation (QVT) transformations[h]to be automatically performed [45]. The authors claim that the main benefit of MDA is that less time and effort are needed to develop the whole Web application, thus improving productivity.

Requirements Engineering:. UWE proposes the use of interviews, questionnaires and checklists as requirements elicitation mechanisms [8]. With regard to requirements specification, UWE models requirements with UML use case diagrams together with textual descriptions of use cases and UML activity diagrams. Use case diagrams are used to represent an overview of the functional requirements, while activity diagrams provide a more detailed view. In recent works [33], UWE has been extended in order to support the UML profile for Web requirements WebRE (Web REquirements metamodel) [32]. This profile has been defined by UWE's authors in a joint work with the authors of NDT (presented below) [32]. Interestingly, UWE considers any kind of requirements according to the taxonomy presented in Section 2: content, service, navigational, layout, personalization, and non-functional requirements. This method does not support the requirements validation (i.e., traceability).

Tool support: A plugin called MagicUWE was developed to be used with the CASE tool MagicDraw[i][46], which allows UWE to provide a CASE (Computer-Aided Software En-

[h]http://www.omg.org/spec/QVT/1.0
[i] http://www.magicdraw.com

gineering) tool. Unfortunately, requirements analysis is missing in this tool.

It is worth mentioning that in [47] the author proposes an approach called UWE4JSF for the automatic generation of Web applications in JSF (Java Server Faces) derived from UWE models.

Approach limitations: Despite the fact that UWE provides its own tool, it does not provide support for the analysis of requirements, and provides only superficial support for the automatic transformations (related to traceability issues). Moreover, it may be complicated to elaborate use cases when the designer is modelling the navigation of more complex Web applications.

4.2 Navigational Development Techniques (NDT)

NDT [39] is a model-driven approach that covers the requirements analysis phase in the development of Web applications.

Development process: The development process of NDT is divided into three phases:

- Requirements treatment, in which Web application requirements are collected.
- Analysis, which consists of the derivation of the conceptual and navigational models from the requirements.
- Prototyping, in order to develop Web prototypes from the analysis phase.

Requirements Engineering: In the NDT method, the requirements process consists of three phases (capture, definition, and validation) which permit the derivation of three conceptual models (content, navigational and abstract interface). To this aim, NDT applies use case diagrams and a set of textual templates for the requirements description [39]. With regard to the types of requirements, NDT considers content, service, navigational, layout, personalization, and non-functional requirements. In [32], the authors of NDT and UWE have jointly developed a UML-Profile for Web requirements called WebRE. This metamodel is useful for specifying a set of QVT transformations in order to obtain several conceptual models (content, navigational and abstract interface) from requirements specification. NDT supports requirements validation, specifically supports the requirements traceability by means of a requirements matrix. This technique is applied to observe the correspondence between the requirements and artefacts (documents in the requirements phase) that satisfy it [48].

Tool support: NDT has tool support for the requirements engineering phase [48]: NDT-Suite. This is actually a set of tools composed of NDT-Profile, NDT-Driver, NDT Quality and NDT Report.[j]

- NDT-Profile. This is a profile defined in Enterprise Architect.[k]
- NDT-Driver. This uses NDT-Profile to allow model-driven transformations to be executed in order to obtain analysis artefacts.
- NDT Quality. This tool verifies whether NDT constraints and relations have been followed correctly. Traceability errors between phases are also numbered and grouped by phases.

[j] http://www.iwt2.org
[k] http://www.sparxsystems.com/

- NDT Report. This tool obtains the requirements, the analysis, the design and the test documents.

Approach limitations: NDT is not a complete methodology for developing Web applications since it needs to be combined with UWE and WebRE. Importantly, NDT does not consider design and implementation phases, and focuses solely on requirements analysis. NDT hampers the development of a complex Web application, since templates are difficult to complete as they require intensive interviews [32]. Furthermore, the requirements are difficult to maintain in NDT owing to the use of textual templates for their specification. Solving these drawbacks was one of the reasons for developing WebRE metamodel.

4.3 Web Modeling Language (WebML)

WebML [40] is a visual language for specifying the content structure of a Web application and the organization and presentation of such content in a hypertext.

Development process: WebML consists of different phases that must be applied in an iterative and incremental manner. The process has several cycles, each of which produces a prototype or a partial version of the application which permits conducting, testing and evaluation to take place during the early development phases[l]. The development process of this Web engineering approach is divided into six phases:

- Requirements analysis to collect requirements.
- Conceptual modelling to represent the Web system independently of any technological detail.
- Implementation to derive conceptual models in a specific technology.
- Testing and evaluation of the Web application.
- Deployment of the Web application in a specific architecture.
- Maintenance and evolution.

Requirements Engineering: Although a requirements specification phase is considered in WebML, a requirements analysis phase is missing. Requirements are specified by using UML use case and activity diagrams. WebML considers six types of requirements, namely: content, service, navigational, layout, personalization and non-functional requirements [49]. This approach does not support requirements validation.

Tool support: WebML has tool support: WebRatio[m] [50]. The support that WebRatio offers to WebML is mainly focused on providing facilities for a conceptual modeling phase and for the automatic generation of J2EE code (as opposed to traditional prototyping tools, which generate application mockups). In this context, WebML models are implicitly defined within WebRatio with a document type definition (DTD) and with a UML 2.0 profile. WebML models can therefore be used in conjunction with other notations and can ensure interoperability with other tools.

[l] *http://www.webml.org*
[m] *http://www.webratio.com*

Approach limitations: The main limitation of this Web engineering approach is that WebML uses use cases as a technique for requirements specifications without an explicit extension to support Web application requirements. Moreover, this approach has no clearly defined techniques for requirements analysis, in addition to which the requirements specification is not supported by WebRatio. Finally, this approach does not support the requirements validation.

4.4 Web Site Design Method (WSDM)

The WSDM method is a user-centered approach [41] since the Web application is defined according to the requirements of the groups of users.

Development process: The development process of this method consists of the following phases:

- Mission statement, in which the purpose of the Web application must be expressed.
- User modelling, in which the users of the Web applications are classified and grouped.
- Conceptual design, in which a navigational model and a class diagram are designed for the navigational paths.
- Implementation design, whose aim is to create a consistent, pleasing and efficient look and feel for the conceptual design made in the previous phase.
- Implementation of the Web application tailored to a chosen implementation environment.

Requirements Engineering: The initial two phases of WSDM (denominated as mission statement and user modeling) are responsible for managing requirements through techniques such as concept maps (of roles and activities) and the data dictionary for the definition of functional and security requirements. The requirements in this approach are specified in a textual form and are considered to be content, service, navigational, personalization and non-functional requirements [41, 51]. Requirements validation is not supported by WSDM.

Tool support: No tools are publicly available for WSDM. There have been some implementations to support parts of the method but these were never integrated, i.e. a code generation tool called WSDMtool [52].

Approach limitations: This approach presents the same limitation as NDT: the definition of textual requirements in a complex Web application development process is difficult to maintain owing to the use of text for requirements specification. Moreover, WSDM does not have a fully-supported tool and lacks requirements validation support.

4.5 An Object-Oriented Approach for Web Solutions Modeling (OOWS)

OOWS [53] is an object-oriented software production method that provides conceptual modeling extensions (in terms of models and abstraction primitives) to facilitate the Web application specification [54]. OOWS extends the OO-Method [55] by adding navigational and presentation characteristics.

Development process: OOWS has a model-driven development process [56] in which there are two main steps: conceptual modeling and solution development [57].

- Conceptual Modeling (i.e., problem space). The aim of this step is to obtain the system specification. It is divided into three sub-steps:
  1. Functional requirements elicitation: techniques based on use cases and scenarios are applied to build a conceptual schema.
  2. Classic conceptual modeling: structural, functional and dynamic models are used to capture the system structure and its behavior.
  3. Navigation and presentation modeling: a navigational model is built in order to specify navigational requirements from the class diagram. Once the navigational model has been built, presentation requirements are specified by using a presentation model which is strongly based on the navigational model.
- Solution development (i.e., solution space). Some patterns are applied to obtain the Web application from the system specification.

Requirements Engineering: The requirements analysis phase is carried out through a set of strategies, namely: (i) FRT (Functional Refinement Tree), (ii) use cases, and (iii) tasks, task specification and data description diagrams for navigation requirements. Finally, OOWS considers six types of requirements: content, service, navigational, layout, personalization and non-functional requirements [53, 58, 59, 60]. With regard to requirements validation, this approach focuses on tracing requirements (traceability) by using the navigational model in a model-driven process through a set of transformation rules defined by graph theory, thus omitting traceability from requirements to the other models involved in the development process [34].

Tool support: This approach has a development framework called OOWS-Suite [61], which is integrated with the OlivaNova tool[n] to provide support for requirements engineering.

Approach limitations: Task analysis is a technique used in the requirements specification which may be time-consuming, complex and depends largely on the experience of the analyst for its correct implementation. Moreover, according to [37], the users' needs are not necessarily well defined within their own mind so as to be defined as tasks. From the designer's perspective, the user goals are ill-defined (no good criterion is defined to be satisfied), i.e., they are not easily reducible to specific tasks that can be easily mapped onto interface features. Finally, the last limitation of this approach concerns requirements traceability, which is only from the requirements to the navigational model, signifying that the other conceptual models involved in the Web application development process are ignored.

4.6 Adaptive Object Oriented Hypermedia (A-OOH)

The A-OOH method [27], [62] is an extension of the Object-Oriented Hypermedia (OO-H) modelling method [63]. The main difference between A-OOH and OO-H are the personalization capabilities [64] which support the modelling of adaptive Web sites.

Development process: The A-OOH design process is based on MDA.

- Computational Independent Model (CIM). At this level, the requirements are specified by the Web application designer. This is done by means of a graphic editor that al-

[n] http://www.care-t.com/

lows the specification of the Web application requirements through the i^* modelling framework, considering the user goals, needs and the *softgoals*.

- Platform Independent Model (PIM). There are several models at this level: domain, navigational, presentation, user and personalization models.These models are generated from the CIM, and can be further completed.

- Platform Specific Model (PSM). This represents the implementation phase of the Web application.

In this approach the generation of conceptual models from requirements bridges the gap between user needs and Web design, since it is ensured (verification) that the conceptual models conform to the requirements specification (model).

Requirements Engineering: A-OOH uses the i^* framework [22] by means of a UML profile implemented such as ecore metamodel[o]. By means of the metamodel, the requirements specification is performed in order to create a requirements model. Conceptual models are then generated by means of model transformations, which are applied automatically. A-OOH applies the taxonomy presented in Section 2 to classify the types of requirements used (content, service, navigational, layout, personalization and non-functional) [27]. With regard to requirements validation, traceability in A-OOH is partially supported [65], since the authors only provide support for traceability from the CIM level (requirements) to the PIM level [66] by means of weaving models [67]. Moreover, this approach has support for impact analysis [68].

Tool support: The development process is supported by the Eclipse[p]development platform [69]. The A-OOH method has an GMF (Graphical Modeling Framework) editor[q]called WebREd (Web Requirements Editor) with which the requirements are specified, once the requirements are specified the Web application conceptual models are generated by means of transformation rules, the rules are implemented by means of the Atlas Transformation Language (ATL[r]).

Approach limitations: One limitation of this approach is that traceability in A-OOH is partially supported, since traceability is only from the CIM to the PIM level and not from CIM to PSM or code level. Moreover, A-OOH does not have a complete supported model-driven tool since it only supports the code generation from PIM level.

4.7 Object-Oriented Hypermedia Design Method (OOHDM)

The OOHDM [70, 35] is an approach for the development of Web applications based on HDM [71], one of the first methods designed for hypertext or hypermedia applications. This approach was the first to propose the separation of aspects, an idea that has been applied by different Web approaches to date.

Development process: The OOHDM development method considers five phases:

- Requirements gathering. *Stakeholder* requirements are captured in this phase.

[o]http://www.eclipse.org/emf
[p]http://www.eclipse.org/
[q]http://www.eclipse.org/gmf
[r]http://www.eclipse.org/atl

- Conceptual design. The OOHDM approach models the application domain using object oriented modelling constructs and some design patterns.
- Navigational design. In this phase the OOHDM approach takes into account the user profile and the task, and builds the navigational structure of the Web application.
- Abstract interface design. OOHDM describes the interface for navigational objects and defines the layout of interface objects (responses to external events).
- Implementation. The implementation of interface objects.

Requirements Engineering: The first step is to gather the *stakeholders'* requirements. To achieve this, it is first necessary to identify the actors (*stakeholders*) and the tasks they must perform. Next, scenarios are collected (or drafted), for each task and type of actor. The scenarios are then collected to form a use case, which is represented by using user interaction diagrams. These diagrams provide a concise graphical representation of the interaction between the user and the system during the execution of a task. This approach presents guidelines to define conceptual and navigational schemas by means of rules described in natural language, and these rules indicate how the conceptual and navigational schemas can be defined from user interaction diagrams. OOHDM only considers three types of requirements: conceptual, navigational and layout requirements. OOHDM does not support requirements validation.

Tool support: This approach has an environment called OOHDM-Web. OOHDM-Web allows the implementation of hypermedia applications such as CGI scripts which produce dynamically generated pages, whose contents are fed from a database and integrated with pre-defined templates [72]. This tool can be downloaded from the OOHDM wiki.[s]

Approach limitations: This approach does not place much importance on the requirements phase. Moreover, the navigational structure is captured narrowly and is therefore captured individually for each use case from its associated User Interaction Diagram. Navigation is not therefore considered as a part of the whole (the big picture) of the web application. What is more, OOHDM has neither tool support for requirements engineering nor tool support for traceability.

4.8 Hera

Hera [73] is a model-driven methodology that supports the design of Web information systems. This approach focuses on the processes of integration, data retrieval, and presentation generation. These processes lead to data transformations based on RDF (Resource Description Framework) models. Hera is principally focused on the development of Web semantic information systems. This approach has been extended and called Hera-S, which is a Web design method that combines the strength of Sesame (a popular open source RDF framework) and the rich modeling capabilities of Hera [74].

Development process: As with other Web engineering methods, Hera includes a phase in which the hypermedia navigation is specified. The Hera methodology has the following phases [73, 42]:

[s] http://www.tecweb.inf.puc-rio.b

- The integration and data retrieval phase, which generates a conceptual model from a user query. This model contains the data for which the application will generate a presentation (here this is considered to be the navigation).

- The presentation generation phase considers how to select and obtain the data from the storage part of the application. The handling of the interaction from users is also specified (querying, navigation, or application-specific user interaction).

Requirements Engineering: Hera does not have a well-defined requirements analysis phase, but in [75], the authors propose a specification method and a procedure for the automatic generation of a target model from a workflow model. The workflow model describes the business process in the system and the collaboration with users and external systems by means of a task model, in which the actors' tasks only contain activities providing interaction with the system process. This approach considers requirements of the content, service, navigational, layout and personalization types. This approach does not support requirements validation.

Tool support: In the Presentation generation phase, the Hera method uses Saxon 7.0[t] (XPath 2.0, XSLT 2.0). For the purpose of data retrieval, Hera uses the RDF query language RQL[u] and its Java-based interpreter called Sesame.[v]

Approach limitations: This approach does not have a well-defined requirements phase and lacks a tool to support requirement engineering. What is more, this approach does not have support for traceability.

5 Discussion

This section provides an analysis of the methods studied in the previous section, with the objective of answering the research questions presented in Section 3.1. This analysis is principally focused on the techniques applied to each method in the requirements engineering phase, requirements terminology and tool support.

5.1 Requirements engineering in Web methods (RQ1)

In Web engineering, almost all the methods studied in this systematic literature review (OOWS, NDT, OOHDM, A-OOH, UWE, WSDM, WebML, HERA) consider a requirements engineering process. Hera is the exception, because it does not have an explicit requirements analysis phase, regardless of its being an engineering method for the development of Web applications. In this context, the UWE, NDT and A-OOH are those methods which have placed greater importance on the requirements analysis phase by defining a set of formal guidelines to be used. On the one hand, A-OOH and UWE have a development process based on MDA. This process considers the CIM model in MDA as the requirements model. Therefore, in both approaches, the requirements are considered since the early stages of the Web application development process, thus making the development and maintenance of Web applications easier, whilst fulfilling the project budget. The development process of the NDT, OOWS and Hera methods are based on a model-driven development method (MDD). On the

[t] http://saxon.sourceforge.net
[u] http://www2002.org/CDROM/refereed/329
[v] http://www.openrdf.org

other hand, NDT focuses solely on the requirements analysis phase and uses the UWE models to cover the entire Web application development process.

5.2 Techniques applied in the requirements engineering process (RQ2)

The techniques applied to each approach in the requirements engineering phase involve a specific set of technologies. Table 9 shows the requirements engineering techniques applied by each Web engineering approach for the analysis and specification of Web requirements. This table shows (i) a trend towards the application of use cases, since this technique is used by OOWS, WebML, NDT, UWE and OOHDM for requirements specifications, and (ii) the persistence of the technique of UML profiles, which is applied by UWE and A-OOH respectively.

Use cases are widely accepted and used in traditional software engineering. Iit is not therefore surprising that Web engineering methodologies also use them to model the possible scenarios that may occur when the user interacts with the Web application. The second technique applied is **UML profiles**, which is used to provide a generic extension mechanism with which to customize UML models for particular domains. The UML profiles technique is, in this respect, applied by UWE and A-OOH since both have an MDA-based development process, UML profiles can therefore be used to adapt particular concepts from a particular domain to its development process, i.e. A-OOH adapts the *i** framework to the Web engineering field in order to use it as a technique for requirements analysis and specification. Another UML technique which is successfully applied in the Web engineering field is the **Activity Diagram**. This technique (applied by WebML and UWE) is a complementary technique for use case diagrams to model the logic captured by a single use case or usage scenario.

The ***i**** **framework** is a goal-oriented technique that is applied to requirements analysis in A-OOH in order to describe and evaluate design alternatives and their relationships to the organizational objectives. This framework is used in order to explicitly analyse and model relationships among multiple *stakeholders* (actors in the i* notation). The *i** framework has proved to be useful for representing: (i) the *stakeholders'* intentions, i.e. their motivations and goals, (ii) dependencies between *stakeholders* to achieve their goals, and (ii) the (positive or negative) effects of these goals on each other in order to be able to select alternative designs for the system, thus maximizing goal fulfilment.

NDT and WSDM, meanwhile, apply **Textual Templates** as a technique for requirements specification with the effort involved to do it this manner. The Textual Template technique is only applicable when the Web application project is not very large, otherwise, textual descriptions will grow significantly, making the textual descriptions difficult to maintain and analyze. This technique can be applied in combination with others, such as in the description of a use case diagram, which, as in the traditional software engineering field, is helpful to the developer, depending, of course, on the level of granularity of the diagram in question.

The OOWS approach uses **Task Analysis** as a requirements specification technique. The term *Task* is frequently applied to activity or process. Task Analysis is a hierarchical representation of which steps have to be performed by a task to achieve a goal. Task analysis is often performed by professionals, and therefore, depends in most cases on the analyst's experience. However, the authors of this approach are currently working on a technique for the specification of requirements through ontologies in order to solve this drawback.

Table 9. RE techniques by each Web engineering approach.

RE Technique	**Approach**
Use Cases	OOWS, NDT, UWE, WebML, OOHDM
Data Dictionary	WSDM
Conceptual Maps	WSDM
Functional Refinement Tree (FRT)	OOWS
UML-Profiles	A-OOH, UWE
Task Diagrams	OOWS
Textual Templates	NDT, WSDM
*i** Framework	A-OOH
Activity Diagrams	WebML, UWE

Functional Refinement Tree (FRT) is a technique applied by OOWS to represent a hierarchical decomposition of business functions of a system which is independent of the actual system structure (it is combined with Task Diagrams).

The requirements specification techniques used by WSDM are **Conceptual Maps of Roles and Activities** and **Data Dictionary**. These techniques are difficult to maintain and analyse owing to the fact that the requirements are basically defined in textual form. In this respect, describing navigational requirements by means of textual descriptions is not an easy task owing to the description of alternative navigation paths through the Web application.

The last technique is the **User Interaction Diagram (UID)**. This is used by OOHDM to specify the interaction described in a use case. This technique is basically used to support the communication between the designer and users and to validate use cases.

The techniques presented in this section have advantages and disadvantages, i. e., the use of text for requirements specifications in a complex Web application development process is a disadvantage because it is difficult to maintain. This technique could, nevertheless, be extremely useful and comprehensible in the development of simple Web applications. In the same context, task analysis is a set of techniques which is intended to provide a researcher with a complete understanding of what tasks a user really performs, what is needed to complete those tasks, and what tasks a user should be doing. On the other hand, a task analysis can be an extremely time and resource consuming affair.

With regard to the requirements validation support in the requirements engineering process, the traceability is specifically studied with reference from requirements to the work products involved in the Web application development process. The traceability is a highly important success factor in software engineering for the reason that, during the design phase, requirements traceability allows us to keep track of what happens when a change request is implemented before a system is redesigned (impact analysis). Traceability can also provide information about the justifications, important decisions and assumptions behind requirements [76]. However, one common problem in the requirements engineering phase in the development of Web applications is the absence of requirements traceability in most of the approaches studied in this SLR (Table 10). Only NDT, A-OOH and OOWS cover the traceability issue to some extent. NDT offers traceability support by means of a requirements matrix [48], OOWS focuses on tracing requirements throughout the navigational model by using a set of transformation rules defined by graph theory [34]. However, both approaches NDT and OOWS

omit traceability from requirements to the other models involved in the development process. In this respect, the A-OOH method covers traceability from requirements to the conceptual models of the Web application by means of weaving models [66], even though, this method does not cover traceability to the code level. In this respect, some approaches maintain unidirectional traceability between the models generated at the initiation of the development process and the rest of the models involved, but not from requirements specification to the rest of the models involved in a bidirectional manner. We believe that bidirectional traceability could be very useful in a Web engineering development process, particularly during validation with customers.

Finally, but of no less importance, the impact analysis is only considered by the A-OOH method, and it is done by an algorithm implemented to analyze the dependencies among requirements in the A-OOH requirements model [68].

Table 10. Requirement engineering phase and traceability support by Web engineering approach.

Method	Requirements Engineering Support	Traceability Support
NDT	+	+/-
WebML	+	-
WSDM	+	-
UWE	+	-
OOWS	+	+/-
OOHDM	+	-
Hera	-	-
A-OOH	+	-

5.3 Requirements terminology (RQ3)

The research addressing requirements in Web engineering has produced a heterogeneous terminology for requirements which hinders further progress. A unifyied vocabulary proposed by [8] (introduced in Section 2) is therefore provided to shed light on (i) the expressivity of current approaches when considering requirements in Web engineering, and (ii) the correspondences between the types of requirements used by each approach with the aforementioned taxonomy. An overview of this is shown in Table 11.

Table 11. Types of requirements and terminology used by each Web engineering approach.

	Type of Requirements					
Method	**Content**	**Service**	**Navigational**	**Layout**	**Personalization**	**NFRs**
UWE	Content	Process	Navigation	Presentation	Adaptation	NFRs
NDT	Storage Information	Functional	Interaction	Interaction	Actor	NFRs
WebML	Content	Service	Navigational	Presentation	Personalization	NFRs
WSDM	Content	Functional	Navigational	X	Personalization	Security, Usability
OOWS	Functional	Functional	Navigational	Presentational	Presentation	NFRs
OOHDM	Content	X	Navigational	Layout	X	X
HERA	Content	Service	Navigational	Presentation	Personalization	X
A-OOH	Content	Service	Navigational	Layout	Personalization	NFRs

In the same order of ideas, of all the approaches presented in Section 4, only A-OOH [27], UWE [77], WebML [49], OOWS [53], and NDT [39] cover every type of requirements relevant to the classification introduced in Section 2. However, these approaches use their own terminology to name each type of requirement, with the exception of A-OOH which applies the classification directly.

Although the approaches presented in Table 11 share, in a few cases, a term with the same name as regards the taxonomy, some types of requirements are used to consider extra functionality, i.e., NDT includes Navigational and Layout requirements in the concept of Interaction requirements and OOWS uses Content and Service requirements in the terms used for Functional requirements [58, 59, 60].

Finally, Non-Functional requirements are considered in a very general manner by almost all the approaches, as can be noted in Table 11. The only approach that considers Non-Functional requirements in a more detailed form is WSDM, which details the Security and Usability Non-Functional requirements to use. [51].

5.4 Tool support (RQ4)

This section provides an analysis of the tools offered by each approach with regard to method support and the requirements engineering process. Table 12, shows the approaches studied in this SLR as regards tool support for the method and requirements engineering phase. All the methods described in the previous section have a tool support for the methodology. NDT its supported by NDT-Tool, WSDM has a code generation tool called WSDMtool, WebML is supported by WebRatio, UWE by ArgoUWE, the OOWS approach uses OlivaNova, OOHDM has a tool support called OOHDM-WEB and Hera uses two tools, Saxon 7.0 and Sesame.

Table 12. Tool support for the method, requirements analysis phase and traceability for each Web engineering approach.

Method	Support	Requirements Engineering	Traceability
NDT	NDT-Tool	NDT-Tool	NDT-Suite
WebML	WebRatio	No	No
WSDM	WSDMtool	No	No
UWE	ArgoUWE	MagicUWE, ArgoUWE	No
OOWS	OlivaNova	OlivaNova, OOWS-Suite	AGG, TaskTracer
OOHDM	OOHDM-WEB	No	No
Hera	Saxon 7.0, Sesame	No	No
A-OOH	Eclipse plugins	Eclipse plugins	Eclipse plugins

With regard to the requirements phase, only NDT, UWE, OOWS and A-OOH have tool support. NDT does this by means of the NDT-Tool,[w] UWE provides a tool that consists of a MagidDraw[x] plugin, the so-called MagicUWE, OOWS combines OOWS-Suite with OlivaNova tool,[y] and finally, A-OOH has requirements phase support by means of an Eclipse plugin [69].

In terms of tool support for requirements validation, the three (NDT, OOWS, A-OOH) methods that have implemented traceability have tool support. A-OOH does by means of an Eclipse plugin with which the requiremens are specified and when the generation of the conceptual models is performed a weaving model is created to store the traces among requirements and the conceptual models. NDT does this by means of the NDT-Suite. The OOWS approach uses two tools, the first of which is the open source tool called AGG (Attributed

[w] http://www.iwt2.org
[x] http://www.magicdraw.com
[y] http://www.care-t.com/

Graph Grammar System) and the second of which is TaskTracer[z]developed by the authors to generate traceability reports. These reports are helpful to study aspects such as whether all the requirements are supported, the impact of changing a requirement, or how requirements are modelled [34]. With regard to the impact of changing a requirement, A-OOH supports impact analysis (verify the impact resulting from the change in any model) by means of the WebREd prototype tool. The WebREd tool generates a report with the potentially requirements affected because of a change.

5.5 Spreading of the methods for the development of Web applications

Figure 13 shows the number of publications by each Web engineering approach ordered by year with regard to requirements engineering and tool support. This figure shows a period from 1998 to 2009 based on the relevant documents for this systematic literature review, and also shows the conference, journal or workshop in which each study was published. OOWS has the greatest number of publications: three from 2002 to 2009, followed by two publications concerning UWE. According to this research, the most important conference in the Web Engineering field for the main topics in this systematic literature review is ICWE (*International Conference on Web Engineering*). The conferences, journals and workshops in which the publications were presented can be consulted in the appendix, which may serve as a future reference for the publication of research work in this area.

We shall conclude this analysis by mentioning the spreading of the different approaches, which is a highly important issue since it assists in the realisation of important advances in the standardization of Web application development. The approaches must therefore be well-known in both the academic world and the software industry. In this respect, it is worth mentioning the support offered by NDT, WSDM, UWE and WebML through their websites. They all offer examples, published papers and their respective tools to everyone who visits their website, with the exception of WSDM, which only offers the downloading of published papers because the tool's licence is not free. In the particular case of UWE and WebML, it is necessary to mention that their websites provide guided step by step examples with which to study and practice the development of Web applications using their respective support tool. This confirms why these two approaches are those most frequently used in academic (university) projects.

5.6 Suggestions for future research

The results of this systematic literature review show that Web engineering methods were not specifically developed to address the analysis of Web engineering requirements, and new adjustments are therefore necessary since:

- Those Web engineering approaches that have a requirements phase do not provide a systematic method with which to design models from requirements since they consider the requirements without real user goals and needs. This gap must be covered, because the successful development of a system depends on satisfying *stakeholders'* needs.

- Virtually none of the Web engineering approaches do not support requirements validation: they simply place less importance on traceability, do not include it at all, or its

[z]http://eecs.oregonstate.edu/TaskTracer/

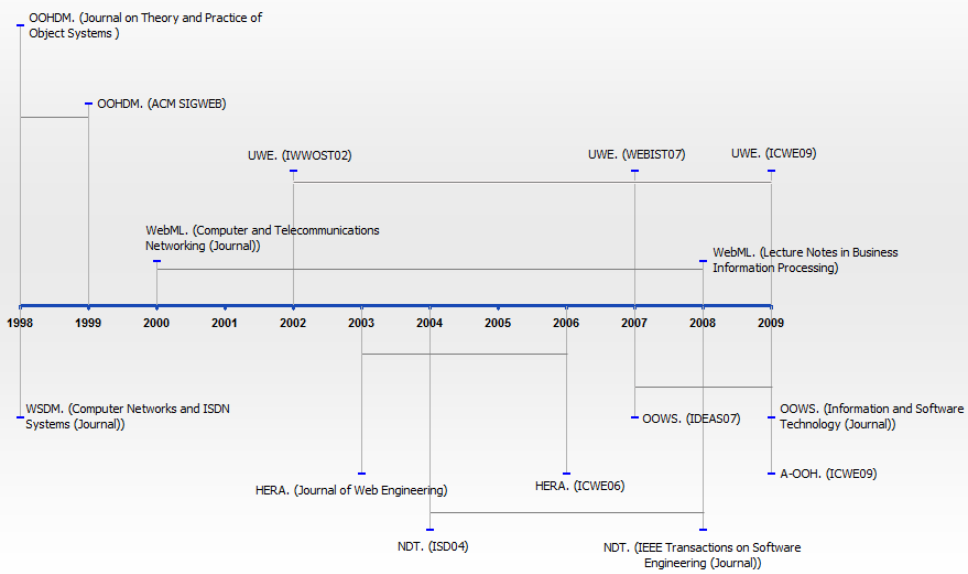


Table 13. Publications for each Web engineering approach ordered by year (refers only to RA phase).

corresponding techniques are poorly applied in the Web engineering field. This deciency must be solved, since the main purpose of traceability is to facilitate the understanding of the product under development and the ability to manage any changes that occur during the development process.

None of the approaches studied has a guided methodology for the analysis of Web engineering requirements that considers real user goals and needs from the beginning of the Web application development process. Empirical studies [18] demonstrate that efforts made to provide a detailed business modelling with which to capture the *stakeholders'* requirements in the system to be built considerably reduce drawbacks in later phases of the development process.

Web engineering methods should therefore adopt a model-driven development process since this offers important advantages, i.e., it improves the development process of a Web application and reduces the development time, thereby reducing the cost of the project. Model-driven development is applied successfully by several Web engineering methods, but its use should be generalized.

Moreover, although traceability is a very important success factor and quality criterion in software engineering, a common problem in Web development processes is the lack of requirements traceability support throughout different phases of the software development life cycle and its multiple levels of abstraction (down to the code level).

To sum up, the suggestions proposed for future research are:

- Explicitly consider the specification of truly user goals and needs from very early stages of the Web application development process in a systematic manner.
- Consider the non-functional requirements from the requirements specification phase in

a structured and comprehensive manner.

- Full support for requirements traceability (bidirectional) from requirements specification throughout the different phases of the Web application development process.

6 Conclusion

This work presents the results obtained after carrying out a systematic literature review. The aim was to create a comprehensive review and synthesis of the current state-of-the-art in the literature related to the requirements engineering in the Web domain. To do this, a total of 3059 papers published in literature and extracted from the most relevant scientific sources were considered, of which 43 were eventually analysed in depth in accordance with the systematic review process adopted.

The results of this systematic literature review have shown that Web engineering methods were not designed to properly address design through the analysis of Web engineering requirements. What is more, they simply place less relevance on requirements validation, or its corresponding techniques are poorly applied.

In this context, most of the approaches explored in this systematic literature review do not consider the real user expectations of the website or *stakeholders* from the early requirements analysis phase. A-OOH is the exception, as it considers these expectations through the i^* Framework, in which requirements are modelled based on user goals and objectives, thus avoiding the specification of requirements in textual form, and based on Task Analysis, which implies certain limitations.

The development of Web applications can no longer be considered as common software systems owing to the diversity of users who have access to these applications. It is therefore necessary to provide solutions in the Web engineering field in order to specify and develop this type of applications by considering the huge heterogeneous user community, and their needs and goals. Our future work will therefore be a model-driven design approach guided by the requirements engineering process applied in the Web engineering domain. The proposal will be entirely based on the A-OOH method. Therefore, it will be helpful in enhancing communication among *stakeholders*, designers and costumers. To this aim, it is necessary to integrate techniques and tools for the automated generation of Web systems. In this respect, this proposal will therefore be supported by an open source tool for requirements specification. The Web designer will be able to use this tool to obtain the Web application conceptual models from requirements, along with support for requirements validation and management.

7 Acknowledgements

This work has been partially supported by the MANTRA project (GRE09-17 from the University of Alicante, and GV/2011/035 from the Valencia Government), and by the MESOLAP (TIN2010-14860) from the Spanish Ministry of Education and Science. José Alfonso Aguilar is funded by CONACYT (Consejo Nacional de Ciencia y Tecnología) Mexico and University of Sinaloa, Mexico.

References

1. Sommerville, I. *Software Engineering.* Addison-Wesley, 6th edition, (2001).
2. Ginige, A. and Murugesan, S. *Cutter IT Journal* **14**(7), 24–35 (2001).
3. Cachero, C. and Gómez, J. In *Interfaces in Navigation Design. 21th International Conference on Conceptual Modeling. El Escorial*, (2002).
4. Casteleyn, S., Garrigós, I., and Troyer, O. D. In *APWeb*, 453–463, (2005).
5. Casteleyn, S., Woensel, W. V., and Houben, G.-J. In *Hypertext*, 189–198, (2007).
6. Ceri, S. and Manolescu, I. In *VLDB*, 1151, (2003).
7. Koch, N. In *International Workshop on Web-oriented Software Technology (IWWOST)*, 40–41, (2002).
8. Escalona, M. J. and Koch, N. *J. Web Eng.* **2**(3), 193–212 (2004).
9. Kitchenham, B. Technical report, Keele University and NICTA, (2004).
10. Mendes, E. In *Empirical Software Engineering, 2005. 2005 International Symposium on*, 10. IEEE, (2005).
11. Staples, M. and Niazi, M. *The Journal of Systems and Software* **80**, 1425–1437 (2007).
12. Jorgensen, M. and Shepperd, M. *Software Engineering, IEEE Transactions on* **33**(1), 33 –53 (2007).
13. Kitchenham, B., Mendes, E., and Travassos, G. *Software Engineering, IEEE Transactions on* **33**(5), 316 –329 May (2007).
14. Nicolás, J. and Toval, A. *Information and Software Technology* **51**(9), 1291–1307 (2009).
15. Lisboa, L., Garcia, V., Lucrédio, D., de Almeida, E., de Lemos Meira, S., and de Mattos Fortes, R. *Information and Software Technology* **52**(1), 1–13 (2010).
16. Kitchenham, B., Pretorius, R., Budgen, D., Pearl Brereton, O., Turner, M., Niazi, M., and Linkman, S. *Inf. Softw. Technol.* **52**, 792–805 August (2010).
17. Aguilar, J. A., Garrigós, I., Mazón, J. N., and Trujillo, J. In *6th Web Information Systems and Technologies (WEBIST)*, volume 2, 187–190 (SciTePress Digital Library, Valencia, Spain, 2010).
18. Nuseibeh, B. and Easterbrook, S. M. In *ICSE - Future of SE Track*, 35–46, (2000).
19. Maiden, N. and Rugg, G. *Software Engineering Journal* **11**(3), 183 –192 (1996).
20. Shaw, M. and Gaines, B. *Software Engineering Journal* **11**(3), 149 –165 (1996).
21. Viller, S. and Sommerville, I. In *Requirements Engineering, 1999. Proceedings. IEEE International Symposium on*, 6 –13, (1999).
22. Yu, E. *Modelling Strategic Relationships for Process Reenginering.* PhD thesis, University of Toronto, Canada, (1995).
23. Sommerville, I. *Software engineering (5th ed.).* Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, (1995).
24. Bass, L., Merson, P., Clements, P., Bergey, J., Ozkaya, I., and Sangwan, R. Technical report, Software Engineering Institute, Carnegie Mellon University, (2006).
25. Gotel, O. and Finkelstein, A. In *Proceedings of the First International Conference on Requirements Engineering*, 94–101, (1994).
26. Estublier, J. In *Proceedings of the Conference on The Future of Software Engineering*, ICSE '00, 279–289 (ACM, New York, NY, USA, 2000).
27. Garrigós, I., Mazón, J.-N., and Trujillo, J. In *ICWE*, 151–165, (2009).
28. Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., and Linkman, S. *Information and Software Technology* **51**(1), 7 – 15 (2009).
29. Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., and Khalil, M. *J. Syst. Softw.* **80**, 571–583 April (2007).
30. Walia, G. S. and Carver, J. C. *Information and Software Technology* **51**(7), 1087 – 1109 (2009).
31. Beecham, S., Baddoo, N., Hall, T., Robinson, H., and Sharp, H. *Information and Software Technology* **50**(9-10), 860 – 878 (2008).
32. Escalona, M. J. and Koch, N. In *Web Information Systems and Technologies (WEBIST),* Aalst, W., Mylopoulos, J., Sadeh, N. M., Shaw, M. J., Szyperski, C., Filipe, J., Cordeiro, J., and Pedrosa, V., editors, volume 1 of *Lecture Notes in Business Information Processing*, 267–280. Springer

Berlin Heidelberg (2007).
33. Koch, N., Zhang, G., and Escalona, M. J. In *ICWE '06: Proceedings of the 6th international conference on Web engineering*, 281–288 (ACM, New York, NY, USA, 2006).
34. Valderas, P. and Pelechano, V. *Inf. Softw. Technol.* **51**(4), 749–768 (2009).
35. Schwabe, D. and Rossi, G. *Communications of the ACM* **38**(8), 45–46 (1995).
36. Molina, F. and Toval, A. *Adv. Eng. Softw.* **40**(12), 1306–1317 (2009).
37. Bolchini, D. and Mylopoulos, J. In *WISE '03: Proceedings of the Fourth International Conference on Web Information Systems Engineering*, 166 (IEEE Computer Society, Washington, DC, USA, 2003).
38. Koch, N. In *ICWE '06: Workshop proceedings of the sixth international conference on Web engineering*, 3 (ACM, New York, NY, USA, 2006).
39. Escalona, M. J. and Aragón, G. *IEEE Transactions on Software Engineering* **34**(3), 377–390 (2008).
40. Ceri, S., Fraternali, P., and Bongio, A. *The International Journal of Computer and Telecommunications Networking* **33**(1-6), 137–157 (2000).
41. De Troyer, O. M. F. and Leune, C. J. *Comput. Netw. ISDN Syst.* **30**(1-7), 85–94 (1998).
42. Houben, G., Barna, P., Frasincar, F., and Vdovjak, R. *Web Engineering* , 529–538 (2003).
43. Koch, N. and Kraus, A. In *IWWOST 02: Second Int. Worskhop on Web-oriented Software Technology*, (2002).
44. Koch, N. In *Workshop proceedings of the sixth international conference on Web engineering*, 3. ACM, (2006).
45. Koch, N., Knapp, A., Zhang, G., and Baumeister, H. In *Web Engineering: Modelling and Implementing Web Applications*, Human-Computer Interaction Series, 157–191. Springer London (2008).
46. Busch, M. and Koch, N. In *ICWE '9: Proceedings of the 9th International Conference on Web Engineering*, 505–508 (Springer-Verlag, Berlin, Heidelberg, 2009).
47. Kroiss, C., Koch, N., and Knapp, A. In *Web Engineering,* Gaedke, M., Grossniklaus, M., and Daz, O., editors, volume 5648 of *Lecture Notes in Computer Science*, 493–496. Springer Berlin / Heidelberg (2009).
48. Escalona, M., Mejías, M., and Torres, J. In *13th International conference on information systems development: methods and tools, theory and practice, Vilna, Lithuania*, 149–59, (2004).
49. Bongio, A., Milano, P. D., Fraternali, P., Maurino, A., and Ceri, S. In *The World Wide Web and Databases (WebDB, Selected Papers*, 201–214, (2000).
50. Acerbis, R., Bongio, A., Brambilla, M., Butti, S., Ceri, S., and Fraternali, P. In *Objects, Components, Models and Patterns,* Aalst, W., Mylopoulos, J., Sadeh, N. M., Shaw, M. J., Szyperski, C., Paige, R. F., and Meyer, B., editors, volume 11 of *Lecture Notes in Business Information Processing*, 392–411. Springer Berlin Heidelberg (2008).
51. Troyer, O. D. and Casteleyn, S. In *IWWOST01*, 30–98, (2001).
52. Wilder, K. V. *Implementation Generation for WSDM using Web Applications Framework.* PhD thesis, University of Brussel, (2009).
53. Fons, J., Valderas, P., Ruiz, M., Rojas, G., and Pastor, O. In *International Workshop on Web Oriented Software Technology (IWWOST). (2003) 6570*, 65–70, (2003).
54. Pastor, O., Abrah ao, S. M., and Fons, J. In *EC-Web 2001: Proceedings of the Second International Conference on Electronic Commerce and Web Technologies*, 16–28 (Springer-Verlag, London, UK, 2001).
55. Pastor, O., Pelechano, V., Insfrn, E., and Gmez, J. In *Conceptual Modeling ER 98,* Ling, T. W., Ram, S., and Lee, M. L., editors, volume 1507 of *Lecture Notes in Computer Science*, 183–196. Springer Berlin / Heidelberg (2004).
56. Quintero, R., Pelechano, V., Pastor, O., and Fons, J. *Jornadas de Ingeniería de Software y Base de Datos (JISBD), VIII* , 84–668 (2003).
57. Cors, J. F. *OOWS: Un Metode Dirigit per Models per al Desenvolupament d Aplicacions Web.* Tesis doctoral en informtica, Departamento de Sistemas Informáticos y Computación, Universidad

Politécnica de Valencia, (2008).
58. Fons, J., Garca, F., Pelechano, V., and Pastor, O. In *Web Engineering,* Lovelle, J., Rodrguez, B., Gayo, J., del Puerto Paule Ruiz, M., and Aguilar, L., editors, volume 2722 of *Lecture Notes in Computer Science*, 457–461. Springer Berlin / Heidelberg (2003).
59. Abraho, S., Fons, J., Gonzlez, M., and Pastor, O. In *Adaptive Hypermedia and Adaptive Web-Based Systems,* De Bra, P., Brusilovsky, P., and Conejo, R., editors, volume 2347 of *Lecture Notes in Computer Science*, 358–362. Springer Berlin / Heidelberg (2006).
60. Durán, G. E. R. *Modelling Adaptative Web Applications in OOWS.* PhD thesis, Technical University of Valencia, Spain, (2008).
61. Valverde, F., Valderas, P., and Fons, J. *IDEAS, Venezuela* (2007).
62. Garrigós, I., Gomez, J., and Houben, G.-J. *Inf. Softw. Technol.* **52**(9), 991–1010 (2010).
63. Gómez, J., Cachero, C., and Pastor, O. In *CAiSE '00: Proceedings of the 12th International Conference on Advanced Information Systems Engineering*, 79–93 (Springer-Verlag, London, UK, 2000).
64. Garrigós, I. *A-OOH: Extending Web Application Design with Dynamic Personalization.* PhD thesis, University of Alicante, Spain, (2008).
65. Aguilar, J. A., Garrigós, I., Mazón, J. N., and Trujillo, J. *Journal of Universal Computer Science JUCS* **16**(17), 2475–2494 (2010).
66. Aguilar, J. A., Garrigós, I., and Mazón, J.-N. In *DSDM: Actas del VII Taller sobre Desarrollo de Software Dirigido por Modelos, JISBD, Congreso Espanol de Informatica (CEDI)*, 146–155 (SISTEDES, Valencia, Espana, 2010).
67. Barbero, M., Del Fabro, M., and Bézivin, J. In *3rd ECMDA-Traceability Workshop.* Citeseer, (2007).
68. Aguilar, J., Garrigós, I., and Mazón, J. *Computational Science and Its Applications-ICCSA 2011* , 421–436 (2011).
69. Eclipse. (2010).
70. Garzotto, F., Paolini, P., and Schwabe, D. *ACM Transactions on Information Systems (TOIS)* **11**(1), 1–26 (1993).
71. Garzotto, F., Paolini, P., and Schwabe, D. *ACM Trans. Inf. Syst.* **11**, 1–26 January (1993).
72. Schwabe, D., de Almeida Pontes, R., and Moura, I. *ACM SIGWEB Newsletter* **8**(2), 18–34 (1999).
73. Vdovjak, R., Frasincar, F., Houben, G., and Barna, P. *Journal of Web Engineering* **2**, 3–26 (2003).
74. Casteleyn, S., Van Woensel, W., and Houben, G.-J. In *HT '07: Proceedings of the eighteenth conference on Hypertext and hypermedia*, 189–198 (ACM, New York, NY, USA, 2007).
75. Barna, P., Frasincar, F., and Houben, G.-J. In *ICWE 06: Proceedings of the 6th international conference on Web engineering*, 321–328 (ACM, New York, NY, USA, 2006).
76. Ramesh, B. and Jarke, M. *IEEE Transactions on Software Engineering* **27**, 58–93 (2001).
77. Baumeister, H., Koch, N., and Zhang, G. In *International Conference on Web Engineering (ICWE 2005*, 406–416. Springer Verlag, (2005).

Appendix A

The appendix shows a list of the conferences, workshops and journals in which the authors of the Web engineering approaches studied in this systematic literature review have presented their work. This list is simply a reference of research publications which may be of interest to readers of this work in the future.

Conferences

- International Conference on Information Systems Development (ISD).
- International Conference on Web Engineering (ICWE).

- International Conference on Web Information Systems and Technologies (WEBIST).
- Conference on Hypertext and Hypermedia (HT).
- International Conference on Advanced Information Systems Engineering (CAiSE).
- International Conference on Electronic Commerce and Web Technologies (EC-Web).
- Conferencia de Ingeniera de Requisitos y Ambientes de Software (IDEAS).
- International Conference on Conceptual Modeling (ER).

Workshops

- International Workshop on Web-Oriented Software Technologies (IWWOST).
- Taller sobre Desarrollo de Software Dirigido por Modelos (DSDM) en XV Jornadas de Ingeniera del Software y Bases de Datos(JISBD).

Journals

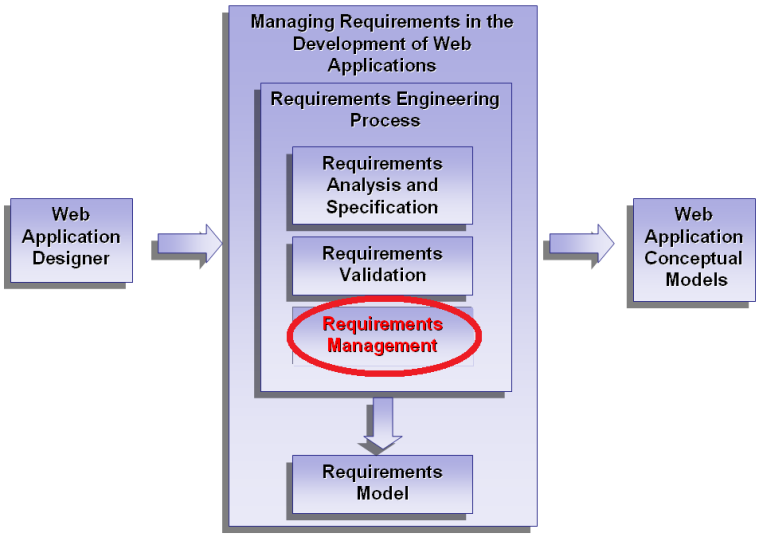
- Computer Networks: The International Journal of Computer and Telecommunications Networking[a]
- IEEE Transactions On Software Engineering[b]
- Information and Software Technology[c]
- Journal of Web Engineering[d]
- ACM Special Interest Group on Hypertext, Hypermedia and the Web[e] (SIGWEB) (Newsletter).
- Communications of the ACM[f]

[a] http://www.elsevier.com
[b] http://www.computer.org/portal/web/tse/
[c] http://www.elsevier.com
[d] http://www.rintonpress.com/journals/jwe/
[e] http://www.sigweb.org
[f] http://portal.acm.org

B

Dealing with dependencies among functional and non-functional requirements for impact analysis in Web engineering

El contenido de este apéndice ha sido enviado a International Journal of Open Source Software and Processes (IJOSSP)



En este apéndice, se presenta una extensión del Capítulo 4 el cual consiste en un algoritmo para manejar las dependencias entre los requisitos funcionales y los requisitos no-funcionales de la aplicación Web en un contexto orientado a objetivos (*goal-oriented*). Con el algoritmo, es posible comprender cuál es el impacto en los requisitos procedente de un cambio en la estructura de modelos conceptuales que conforman la aplicación Web, así como saber qué requisitos necesitan ser implementados para cumplir, en medida de lo posible, los propósitos establecidos en el análisis orientado a objetivos. En esta extensión se describe la implementación del editor gráfico WebREd para la especificación de requisitos así como la implementación del algoritmo para analizar el impacto en los requisitos derivado de un cambio en los modelos conceptuales de la aplicación Web.

Dealing with dependencies among functional and non-functional requirements for impact analysis in Web engineering

José Alfonso Aguilar
Computer Science Faculty, University of Sinaloa, Mexico
Irene Garrigós
Department of Software and Computing Systems (DLSI), University of Alicante, Spain
Jose-Norberto Mazón
Department of Software and Computing Systems (DLSI), University of Alicante, Spain

ABSTRACT
Due to the dynamic nature of the Web, as well as its heterogeneous audience, Web applications are more likely to rapidly evolve, thus leading to inconsistencies among requirements during the development process. Importantly, to deal with these inconsistencies Web developers need to know dependencies among requirements. Furthermore, the understanding of requirement dependencies also helps in better managing and maintaining Web applications. Therefore, in this paper, an algorithm has been defined in order to analyze dependencies among functional and non-functional requirements, thus understanding which is the impact derived from a change during the Web application development process. This impact analysis would support Web developer in selecting requirements to be implemented, thus ensuring that Web applications finally satisfy the audience

Keywords: Impact analysis, goal-oriented requirements engineering, Web engineering.

INTRODUCTION
Requirements in Web engineering tend to rapidly evolve due to the dynamic nature of the Web (Ginige, 2002). This continuous evolution may lead to inconsistencies among requirements which hinder Web developers from understanding the impact of a change in the Web application. Impact analysis of requirements is therefore a crucial task in Web engineering order to know how changes in a Web requirement affect the other requirements, thus supporting Web developers in making the right design decisions.
Impact analysis is the task of identifying the potential consequences of a change, or estimating what needs to be modified to accomplish a change (Arnold & Bohner, 1993). We define a "change" as any modification on a Web requirement. Usually, impact analysis has been done intuitively by Web applications developers, after some cursory examination of the code and documentation. This may be sufficient for small Web applications, but it is not enough for

sophisticated ones. In addition, empirical investigation shows that even experienced Web applications developers predict incomplete sets of change impacts (Lindvall & Sandahl, 1998). Therefore, to effectively analyze the impact of requirements in Web applications, dependencies among requirements should be explicitly considered, thus better managing changes in Web applications. Actually, inconsistencies are defined as negative dependencies among the set of requirements, caused by the fact that requirements often originate from stakeholders with different or conflicting viewpoints (Zhang, Mei, & Zhao, 2005). Usually, impact analysis is performed only by considering dependencies on functional requirements (FR), leaving aside the non-functional requirements as shown in (Zhao, 2002) and (Gupta, Singh, & Chauhan, 2010). In the software engineering area, FR describe system services, behavior or functions, whereas NFR, or quality requirements, specify a constraint on the system or on the development process (Sommerville, 2005). According to (Ameller, Franch, & Cabot, 2010), we believe that the NFRs must be considered from the very beginning of the development process, also in the development of Web applications. Therefore, impact analysis should be done on both kinds of requirements: FR and NFRs.

Interestingly, the recent inclusion of goal-oriented techniques (Nuseibeh & Easterbrook, 2000) in Web requirements engineering, (Bolchini & Mylopoulos, 2003), (J. A. Aguilar, I. Garrigós, J. N. Mazón, & J. Trujillo, 2010), (Garrigós, Mazón, & Trujillo, 2009) offer a better analysis of requirements in Web application development, due to the fact that requirements are explicitly specified in models, thus supporting developers in evaluating the implementation of certain requirements for desigining successful software. In the goal-oriented requirements engineering field, FR are related to goals and sub-goals whereas NFRs are named softgoals, commonly used to represent objectives that miss clear-cut criteria. Specifically, finding right tradeoffs for dependent NFRs is an important step when impact of requirements is being analyzed (Boehm & In, 1996) and (Elahi & Yu, 2009), i.e., a Web application without passwords is usable, but not very secure, increased usability reduce security or increased security reduce usability. Unfortunately, finding these tradeoffs among dependent Web requirements is not a trivial task due to the dynamic nature of the Web.

Therefore, this paper presents a goal-oriented requirements engineering proposal for supporting Web developers in analyzing the impact of a change by considering both FR and NFRs in the Web applications. To this aim, an algorithm has been defined to support Web developer in (i) clearly identifying dependencies among FR and NFRs, and (ii) automatically performing the impact analysis in Web engineering by determining right tradeoffs. The main benefit of our approach is that provides information about the different design alternatives, thus allowing developers to make more informed design decisions for implementing a Web application that fully-satisfies FR, while there is a tradeoff among NFRs.

This paper is as extension of our recent work (Aguilar, Garrigós, Mazón, 2011) about the importance of considering impact analysis in a goal-oriented Web requirements analysis approach. In particular, the novelty of our ongoing work presented in this paper consists of: (i) the implementation of the UML-Profile to adapt the i* framework in the Web domain as a metamodel, (ii) the development of a prototype tool for the Web requirements specification as a proof of concept of our approach, (iii) the implementation of the transformation rules (with a high degree of automation) to derive the Web application conceptual models, and (iv) the implementation of the algorithm for impact analysis in goal-oriented models.

The remainder of this paper is structured as follows: Section 2 presents some related work relevant to the context of this work. Section 3 describes the proposal for goal-oriented requirements analysis where is found the contribution of this work and introduces a running example for demonstration purposes. The algorithm for impact analysis in goal-oriented requirements is presented in Section 4. The application of the algorithm to perform the impact analysis is described step by step in Section 5. In Section 6 is presented the current implementation of this approach. Finally, the conclusion and future work is presented in Section 7.

RELATED WORK

In our previous work (J. A. Aguilar, I. Garrigós, J.-N. Mazón, & J. Trujillo, 2010), a systematic literature review has been conducted for studying requirement engineering techniques in the development of Web applications. Our findings showed that most of Web engineering approaches focus on the analysis and design phases and do not give a comprehensive support to the requirements phase (such as OOHDM (Schwabe & Rossi, 1995), WSDM (Troyer & Leune, 1998) or Hera (Casteleyn, Woensel, & Houben, 2007)). Moreover, we can also conclude that the most used requirement analysis technique is the UML (Unified Modeling Language) use cases. This technique has proved to be successful in the common software development process to deal with the requirements specification in both textual and diagram. But unfortunately, this technique is not enough to deal with aspects such as navigation in the Web application development process. Even though, this technique is applied by some of the most remarkable Web engineering approaches such as OOWS (Pastor, Fons, Pelechano, & Abrahão, 2006), WebML (Ceri, Fraternali, & Bongio, 2000), NDT (Maria J. Escalona & Aragón, 2008) and UWE (Koch, Knapp, Zhang, & Baumeister, 2008).

Furthermore, none of the aforementioned Web engineering approaches perform the analysis and modeling of the users' needs for ensuring that the Web application is not overloaded with useless functionalities, while important functionalities are not missed. We believe that these facts are an important issue that limits a broaden use of these approaches. In this sense, to the best of our knowledge, the only approaches that use goal-oriented requirements analysis techniques for Web engineering have been presented in (Bolchini & Paolini, 2004) and (Molina, Pardillo, & Toval, 2008). Unfortunately, although these approaches use the i* modeling framework (Yu, 1995, 2002) to represent requirements in Web domain, they do not benefit from every i* feature because don't use all the expressiveness of the i* framework to represent the special type of requirements of the Web applications such as the related with navigational issues. To overcome this situation, our previous work (J.A. Aguilar, et al., 2010) adapts the well-known taxonomy of Web requirements presented in (M.J. Escalona & Koch, 2004) to be used within the i* modeling framework.

On the other side, with regard to approaches that consider NFRs from early stages of the development process, in (Molina & Toval, 2009) the authors propose a metamodel for representing usability requirements for Web applications. Moreover, in (Ameller, et al., 2010) the authors present the state-of-the-art for NFRs in model-driven development, as well as an approach for integrating NFRs into a model-driven development process by considering them from the very beginning of the development process. Unfortunately, these works overlook how

to analyze and evaluate the impact among FR and NFRs. However, some interesting works have been done in this area (Horkoff & Yu, 2009a) and (Horkoff & Yu, 2009b). These works evaluate i* models based upon an analysis question (whatif) and the human judgment. To this aim, this procedure uses a set of evaluation labels that represent the satisfaction or denial level of each element in the i* model. First of all, initial evaluation labels reflecting an analysis question are placed in the model. These labels are then propagated throughout the model by using a combination of set propagation rules and the human judgment. The results of this propagation are interpreted in order to answer the stated question. Unfortunately, these general approaches have not been adapted to Web engineering.

The motivation with regard this proposal relies on the deficiencies from the works previously mentioned. Since, they are focused on how to analyze i* models with which to answer a particular question (what-if) without considering the goal satisfaction (the organizational objectives). Thus, our proposal is focused on how to evaluate the impact derived from a change in the i* requirements model in order to use it to offer to the designer a better form to analyze design options from the Web application. For this purpose, the requirements are classified according the taxonomy of Web requirements defined in (M.J. Escalona & Koch, 2004) in the i* requirements model. Moreover, our proposal brings out alternative paths to satisfy the goals bearing in mind the softgoals tradeoff, thus considering the softgoals from the beginning of the Web application development process.

In summary, there have been many attempts to provide techniques and methods to deal with some aspects of the requirements engineering process for the development of the Web applications. However, there is still a need for solutions which enable the designer to know which requirements are affected because of a change in the Web application design besides of considering the NFRs involved in the development process.

GOAL-ORIENTED REQUIREMENTS ANALYSIS IN WEB ENGINEERING

This section describes our proposal to specify requirements in the context of the A-OOH (Adaptive Object-Oriented Hypermedia) Web modeling method (Garrigós, 2008) by using goal-oriented models (J.A. Aguilar, et al., 2010).

A-OOH is an extension of the OOH (Object-Oriented Hypermedia) (Gómez, Cachero, & Pastor, 2000) method with the inclusion of personalization strategies. The development process of this method is founded in the MDA (Model Driven Architecture) (Melia, Cachero, & Gomez, 2003). MDA is an OMG's standard and consists of a three-tier architecture with which the requirements are specified at the Computational Independent Model (CIM), from there are derived the Web application conceptual models which corresponds with the Platform Independent Model (PIM) of the MDA. Finally, the Web application conceptual models are used to generate the implementation code; this stage corresponds with the Platform Specific Model (PSM) from the MDA standard.

The A-OOH approach uses five models at the PIM level to define a Web application, namely:

- Domain model (DM). This encapsulates the structure and functionality required of the relevant concepts of the application domain and reflects the static part of the Web application, is represented such as UML (Unified Modeling Language) class diagram.
- Navigation model (NM). This model aims to specify the structure and behavior of the navigation view over the domain data defined. It defines each path on which users can navigate through the Web application.
- Presentation model (PM). This model defines the layout of the Web application, i. e. the page style, the font color, etc.
- Personalization model (PM). Personalization strategies (Adaptive Web Sites) are specified in this model. Web pages are personalized based on the context, the socioeconomic level and the interest of an individual user. Personalization implies that the changes are based on the items purchased or in the pages viewed.
- User model (UM). This enables the users to be described in terms of their personal information and their relations with a particular application domain as well as the navigational actions performed at execution-time. The structure of the information needed for the personalization strategies is also described in this model.

On top of these models, the A-OOH approach defines Web requirements in a CIM (J.A. Aguilar, et al., 2010). From this CIM, a several transformation rules have been defined for supporting the derivation of the navigational (NM) and domain (DM) conceptual models with a high level of automation (J.A. Aguilar, et al., 2010).

Requirements specification at the computational independent model

The requirements specification in the A-OOH method is performed by means of the i* modeling framework (Yu, 1995, 2002). As a goal-oriented analysis technique, the i* framework focuses on the description and evaluation of alternatives and their relationships to the organizational objectives.

The i* framework consists of two models: the strategic dependency (SD) model () to describe the dependency relationships among various actors in an organizational context, and the strategic rationale (SR) model (), used to describe actor interests and concerns and how they might be addressed. The SR model provides a detailed way of modeling internal intentional elements and relationships of each actor (). The essential elements to perform a goal-oriented requirements analysis using the i* modeling framework are described next.

- Goal () represents an (intentional) desire of an actor; regrettably, goals are not enough for describing how the goal will be satisfied.
- Means-end links are a kind of links representing alternative ways for fulfilling the goals.
- Task () describes some work to be performed in a particular form. The task is decomposed in the necessary intentional elements (tasks and resources) to be performed. This decomposition is by means of the decomposition links ().
- Resource () represents some physical or informational entity required for the actor, in particular by the task element.
- A softgoal () is a goal whose their satisfaction criterion is not clear cut.

In which form contributes an intentional element to the satisfaction or fulfillment of a softgoal is determined via contribution links ($\xrightarrow[hurt]{help}$). The possible labels for a contribution link are “make”, “some+”, “help”, “hurt”, “some-”, “break”, “unknown”, this labels are used to indicate the strength of the contribution made, this could be positive, negative or unknown.

Even though the i* modeling framework provides good mechanisms to model stakeholder’s (actors) and relationships between them, the Web applications have some particular requirements that differs from the traditional ones. These new requirements are defined in the seminal work of Escalona and Koch (M.J. Escalona & Koch, 2004). In this work, the authors put forward the argument that FRs for Web engineering are related to three main features of Web applications: navigational structure, user interface and personalization capability, and that the data structures required by the Web application should also be specified. Therefore, to adapt the i* modeling framework to the Web engineering domain we use the clasification described in the work of Escalona and Koch. As a strong point, with this clasification is possible to reflect special Web requirements that are not taken into account in traditional requirement analysis approaches. An overview of each kind of requirement for Web engineering is described next:

- Content Requirements. With this type of requirements, is defined the website content presented to users. For example, in a book on-line store some examples might be: “book information” or “book categories”.
- Service Requirements. This type of requirement refers to the internal functionality the system as Web application should provide to its users. Following the example of the Content Requirements, for instance: “register a new client”, “add book to cart”, etc.
- Navigational Requirements. These type of requirements refers to the navigational paths available for the users of the Web system. In this sense, some examples are: the user navigation from index page to “consult products by category” or to “consult shopping cart” options.
- Layout Requirements. Requirements can also define the visual interface for the users. For instance: “present a color style”, “multimedia support”, “the user interaction”, among others.
- Personalization Requirements. The designer can specify the desired personalization actions to be performed in the final website (e.g. “show recommendations based on interest”, “adapt font for visual impaired users”, etc.)
- Non-Functional Requirements. These kinds of requirements are related to quality criteria that the intended Web system should achieve and that can be affected by other requirements. Some examples can be “good user experience”, “attract more users”, “efficiency”, etc.

The A-OOH approach is UML-compliant, therefore we have used the extension mechanisms of UML in order to adapt the i* modeling framework to specific Web domain terminology. To do so, (i) we defined a profile to formally represent the adaptation of each one of the i* elements with each requirement type from the Web requirements clasification adopted (Garrigós, et al., 2009); and (ii) we implemented this profile in an EMF (Eclipse Modeling Framework) metamodel. Therefore, new EMF clases have been added according to the different kind of Web requirements (see Fig. 1): the Navigational, Service, Personalization and Layout requirements extends the Task element and the Content requirement extends the Resource class. It is worth

noting that NFRs can be modeled by directly using the softgoal element. The EMF metamodel for Web requirements specification using the i* framework has been implemented in the Eclipse IDE (Integrated Development Enviroment).

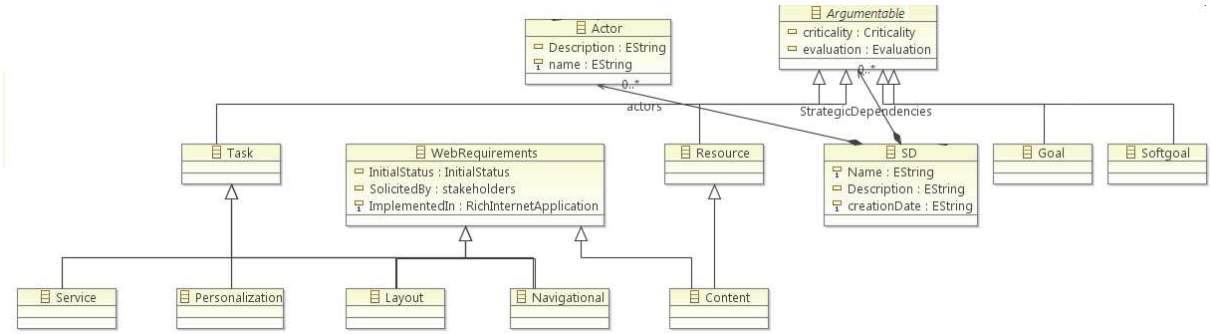


Figure 1. An extract of the i metamodel for Web requirements*

A sample application of the i* modeling framework for Web domain is shown in Figure 2, which represents the SR model of our running example for the Conference Management System (CMS). The purpose of the system is to support the process of submission, evaluation and selection of papers for a conference. The complete specification of the case study can be found at: http://users.dsic.upv.es/~west/iwwost01

A part of the CMS requirements specification (CMS Actor) is modeled in the Figure 2. The requirement specification is focused in the process of selecting the review procedure. Four actors participate in the CMS, for this example only the author, reviewer and system actors were considered. It is important to remark that each element from Figure 2 corresponds to a requirements type from the clasification previously mentioned (from Escalona and Koch), i.e., the content requirement (Content) from the taxonomy is displayed with the notation “Resource” from the i* modeling framework and the navigational (Navigational) and service (Service) requirements with the symbol “Task”, both of them with their respective associations called decomposition-links. A decomposition-link between two elements means that a requirement (“Task”) is decomposed in one or more sub-requirements (“Sub-Tasks”) to be able to perform its function. The Figure 2 depicts a correct scenario of the requirement decomposition by introducing the navigational requirement “Blind Review Process”, decomposed in two sub-requirements named “Download papers without authors’ name” (Service) and “Review Paper” (Navigational). Besides, the labels (☑) and (×) are used in the requirements specification to represent the requirements that are currently implemented in the Web application conceptual models.

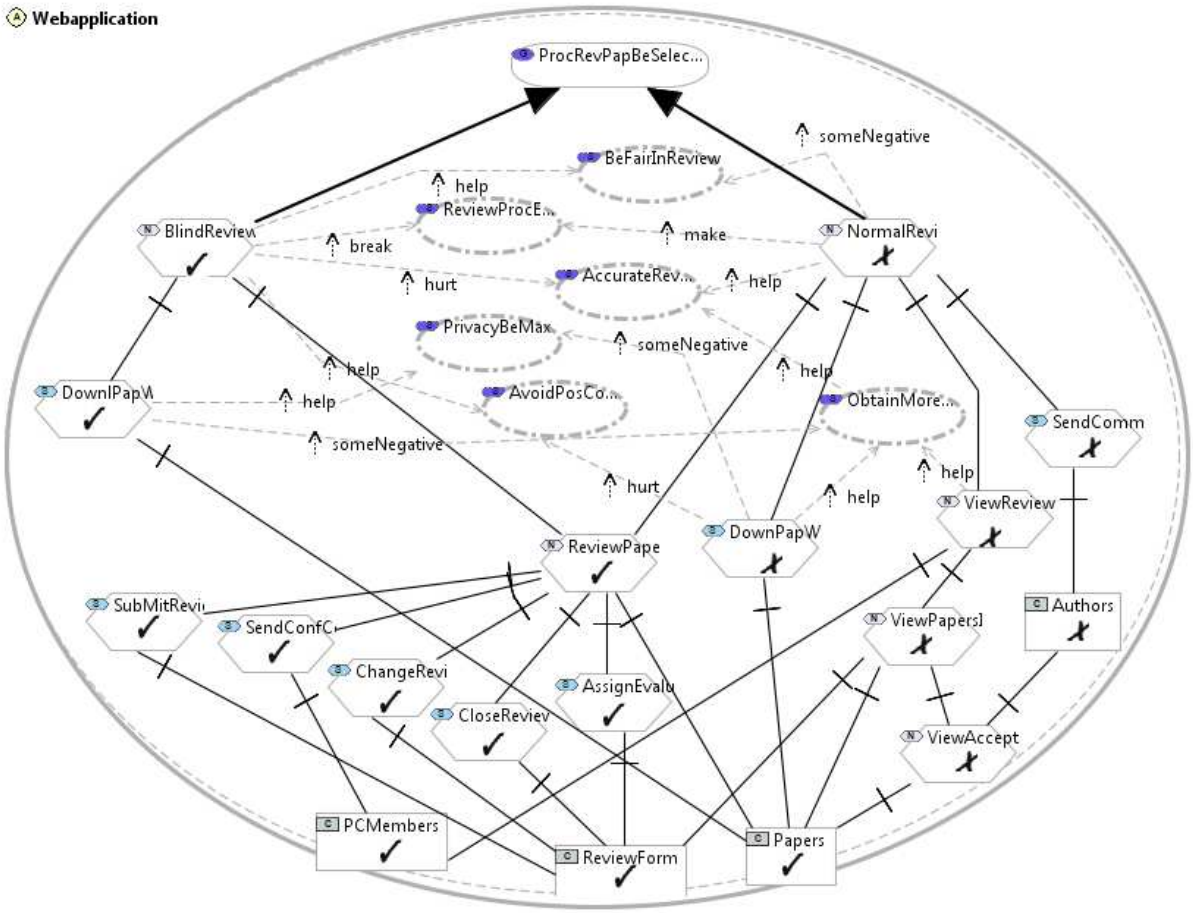


Figure 2. Part of the Conference Management System requirements expressed in a SR and SD Models

Applying the goal-oriented requirements analysis, we found that there are three actors detected which depends on each other to achieve their goal; these are namely “Reviewer”, “Author” and “Conference Management System”. The “Reviewer” actor needs to use the CMS to achieve its own goal which is “Review paper”. The “Author” actor depends on the CMS actor in order to “Paper be reviewed”. In Figure 2 are modeled these dependencies as well as the CMS actor by means of the SD and the SR models. The goal of the CMS actor is “Process of review of papers be selected”. To achieve the goal, the SR model specifies that one of the two navigational requirements: “Blind review process” or “Normal review process” should be performed. In this running example, the path to achieve the goal of the CMS actor is through the navigational requirement “Blind review process”, all the requirements needed to implement this path are labeled with (☑). Besides, we can observe in the SR model that some navigational and service requirements are decomposed in other requirements, some of them affects positively or negatively some NFRs, i.e., the service requirement “Download paper without authors’ name” needs the information provided by the content requirement “Papers”. Moreover, the service requirement affects positively the softgoal “Privacy be maximized” and in some negatively form the softgoal “Obtain more complete info”. This fact is very important to see how to satisfy the goal “Process of review of papers be selected” considering the Web application softgoals.

Accordingly, maximizing or minimizing the contribution from requirements to softgoals is a viable solution to find a path to fully-satisfy the goal.

From the computational independent model to the platform independent model

Once the requirements have been defined they can be used to derive the Web application conceptual models. For the sake of understandability, the process for obtaining the navigational conceptual model will not be described in detail, and our main focus will be on the derivation of the domain model. For a broaden explanation about this process, we refer reader to (J.A. Aguilar, et al., 2010).

The A-OOH domain model is expressed as a UML-compliant class diagram. It encapsulates the structure and functionality required of the relevant concepts of the application and reflects the static part of the system. The main modeling elements of a class diagram are the classes (with their attributes and operations) and their relationships. We have implemented a set of rules using the ATL (Atlas Transformation Language) (ATL, 2011) to derive the domain model from requirements specification (model). In this transformation rules, the source model correspond to our CIM for Web requirements specification while the domain model as a representation model for the Web application domain is the PIM level, both of them in a MDA context.

- Content2DomainClass. By using this transformation rule, each content requirement detected in the requirements model is derived into one class of the domain model (see Fig. 3).
- Service2Operation. Detects a service requirement with an attached content requirement in the SR model, each service requirement is transformed into one operation of the corresponding class (represented by the content requirement).
- Navigation2Relationship. To generate the associations in the domain model we have to detect a navigational root requirement (i.e. task) in the SR model which can contain one or more navigational requirements attached. Each of the navigational requirement can have attached a resource (i.e. content requirement) thus creating an association from one content requirement to other one by means of the navigational requirements associated.

```
--For each Content element in the i* Requirements Model, a UML Class will be created in the Domain Model.
rule Content2Class
{
    from
        vContent : MMiStar!Content
    to
        vClass : UML!Class (name<-vContent.name),
 __traceLink : traceability!TraceLink (
      name <- 'Content2DomainClass',
      sourceElements <- Sequence {__LinkEnd_vContent},
      targetElements <- Sequence {__LinkEnd_vClass},
      model <- thisModule.__wmodel, ruleName <-'Content2DomainClass', description<-'From each Content element
      in RequirementsModel, this rule creates a UML Class in DomainModel'),
    __LinkEnd_vContent : traceability!TraceLinkEnd (element <- __elementRef_vContent ),
    __elementRef_vContent : traceability!ElementRef (ref <- vContent.__xmiID__, modelRef <- thisModule.__model_IN),
    __LinkEnd_vClass : traceability!TraceLinkEnd (element <- __elementRef_vClass ),
    __elementRef_vClass : traceability!ElementRef (ref <- vClass.__xmiID__, modelRef <- thisModule.__model_OUT)

}
```

Figure 3. The transformation rule Content2Class implemented by means of ATL

Figure 4 shows the domain model obtained from requirements specification (model). The domain model is showed by means of the tree editor for easier viewing, all the classes, operations, associations with their respective properties are encapsulated in a UML-Package.

Once the model is derived the designer has only to refine them, avoiding the task of having to create them from scratch. The designer will have to specify the most relevant attributes of the classes, identify the cardinalities and define (if existing) the hierarchical relationships.

IMPACT ANALYSIS ALGORITHM FOR GOAL-ORIENTED REQUIREMENTS IN WEB ENGINEERING

In this section, we present a proposal which provides a form to analyze the impact of a change in the requirements specification (requirements model) within the A-OOH method. Therefore, the Web developer will be able to evaluate the effect of the change and select a sub-set of requirements to implement to fully satisfy the goal. To do this, the designer must perform (find) a balance between the contributions made by the FRs to the NFRs.

The algorithm is designed to be applied in the requirements specification (the i* requirements model) considering the type of contributions made by the intentional elements to the softgoals. This algorithm allows evaluating the impact in the requirements model resulting from removing any element of the A-OOH conceptual models (it is worth noting that, in this paper, our focus is on the previously defined domain model). In this way, it is determined which new requirements should be included in the A-OOH conceptual models for maximizing or balancing the softgoal satisfaction although some requirements have been removed. To this aim, some heuristics have been defined.

Heuristics

Some heuristics have been defined for determining the impact of the contribution links between intentional element and softgoals. Table 1 summarizes some terms for understanding the heuristics presented in this section. These terms correspond to the most common types of contribution links of the i* modeling framework.

Heuristics Terms	i* Contribution Type
Strongly-positive	Help
Weakly-positive	Some +
Strongly-negative	Hurt
Weakly-negative	Some -
Dependent-negative	Break
Dependent-positive	Make

Table 1. The i contributions types*

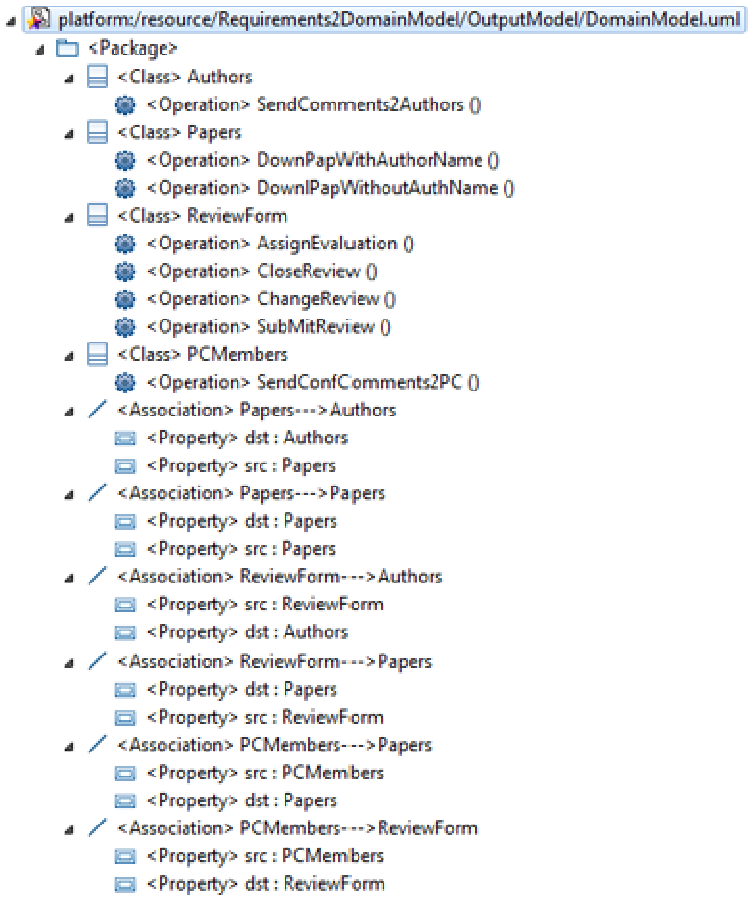


Figure 4. The the domain model derived from the requirements model

The “Help” contribution link is a partial positive contribution, not sufficient by itself to satisfy the softgoal. The contribution link named “Hurt” is partial negative contribution, not sufficient by itself to deny the softgoal. “Some +” is a positive contribution whose strength is unknown. “Some -” is the opposite contribution type to “Some +”, is a negative contribution whose strength is unknown. The “Break” contribution link refers to a negative contribution enough to deny a softgoal. Finally, the “Make” contribution link is a positive contribution strong-enough to satisfy a softgoal.

The heuristics defined are as follows:

- H1. If the contribution of the requirement to remove is strongly-positive, and the contribution of the requirement to implement is strongly-negative, do not implement the requirement.
- H2. If the contribution of the requirement to remove is more stronglypositive than the contribution of the requirement to implement, but the contribution to be implemented is weakly-negative, the requirement could be implemented.

- H3. If the contribution of the requirement to remove is strongly-negative or weakly-negative, and the contribution of the requirement to implement is strongly-positive or weakly-positive, the requirement should be implemented.
- H4. If the polarity of the contribution of the requirement to remove is as negative as the contribution of the requirement to implement, the requirement to implement should not be implemented to maximize the satisfaction of the softgoal.
- H5. If the polarity of the contribution of the requirement to remove is as positive as the contribution of the requirement to implement, the requirement to implement should be implemented to maximize the satisfaction of the softgoal.
- H6. If the contribution of the requirement to remove is Dependent-positive, then the developer should consider whether that requirement should be removed or not considering the need to implement this softgoal.
- H7. If the contribution of the requirement to remove is Dependent-negative, then the developer should consider whether that requirement should be removed or not considering the impact of not implementing this softgoal.

Preconditions and postconditions

The preconditions should hold before the algorithm could be executed and they affects to the elements specified in the requirements model described in the section "Goal-oriented requirements analysis in web engineering". Specifically, these preconditions permit the execution of the algorithm when:

1. The requirement to remove does not affect the goal by the "means-end" contribution type.
2. When there is more than one "means-end" contribution type, it means that the impact analysis will be possible by means of the softgoals tradeoff.
3. The requirement to remove affects other requirements and these requirements (not shared) are not in the possible paths to satisfy the goal.

Moreover, there is one postcondition that must hold when a requirement has been selected to be implemented in the requirements model as an alternative solution for the satisfaction of the goal: if the requirement to be implemented has associated requirements, these requirements must be implemented automatically.

Impact Analysis Algorithm

This algorithm considers the contributions made by the intentional elements from the requirements model to find a path to fully satisfy, where possible, the main goal. To do this, the designer must have to find tradeoffs between the softgoals. The algorithm is presented next.

1. *FUNCTION TradeOf fAlgorithm (RequirementsModel)*
2. *TR= task to remove; TI= task to implement;*
3. *SN= new sof t go a l ; IEList= int e n t ion a l elements l i s t ;*
4. *ASList= af f ected so f t g o a l s l i s t ;*
5. *TIList= l i s t of task to implement ; Value= false ;*
6. *P = PreConditions ( ) ;*
7. *IF (P=true) THEN*
8. *IELi st= CreateIntentionalElement sList (RequirementsModel) ;*
9. *IF(TR. Contributes2Softgoal s ( )) THEN*
10. *ASList=Cr eateAf fectedSoftgoa l sLi s t (TR) ;*
11. *FOREACH s FROM ASLi s t :*
12. *TI= SearchTaskToApply (IELi s t) ;*
13. *Value= Heur i s t i c s (ASList , IEList , TI) ;*
14. *TI . AddValue (Value) ;*
15. *TIList .Add(TI) ;*
16. *IF (TI . Contributes2Softgoal s (TI)) THEN*
17. *ASList . add (SN) ;*
18. *END IF*
19. *END FOREACH*
20. *FOREACH v FROM TILi s t :*
21. *CalculateAverage (v) ;*
22. *IF(CalculateAverage (v)) THEN*
23. *Implements (v) ;*
24. *END IF*
25. *END FOREACH*
26. *END IF*
27. *PostCondition ( ) ;*
28. *ELSE*
29. *ShowMessage (P.message ( )) ;*
30. *END IF*
31. *END PROGRAM*

1.1 Algorithm for impact analysis in goal-oriented Web engineering

A snippet of code of the algorithm for impact analysis in goal-oriented Web requirements specification is shown in 1.1. First, in lines 6 and 7 the pre-conditions are evaluated; these must be “true” to proceed with the execution of the algorithm. Next, from lines 8 to 19, the algorithm creates a list of intentional elements. In these lines, all types of requirements from the requirements model are stored in this list. The next step is to extract those softgoals that receive a contribution from the requirement to remove, for each softgoal from the list, finding a non-

implemented requirement and applying the heuristics introduced in the section “Impact Analysis Algorithm for Goal-oriented requirements in web engineering”. Each of these requirements must be stored in the list, and if it contributes to a softgoal, the softgoal must be stored in the list too. Then, lines 20 to 29 are used to evaluate each element from the list according to the weight of each element assigned by the heuristics to determine when a requirement must be implemented. Finally, the postcondition is executed and the alternative path to fully satisfy the goal from requirements model is obtained.

PERFORMING THE IMPACT ANALYSIS

For the sake of understandability, the following scenario is assumed along this section: the Web developer decides deleting from the A-OOH domain model the elements that were created from the requirement “Download papers without authors’ name”. It is necessary to know which other requirements are affected by this change. In addition, this action implies that the goal “Process of review of papers be selected” can not be satisfied. Thus, it is necessary to search for alternative paths in the i* requirements model (if there any) in order to fully-satisfy the goal “Process of review papers be selected”. To this aim, our algorithm is triggered. The execution of the impact analysis algorithm is detailed next.

The first step to execute our algorithm consists of applying the preconditions. For this running example, the preconditions result true, it means that there is any problem to the algorithm has been executed.

Next, it is necessary to develop a list of the requirements (implemented or not) that contribute to any softgoal in the i* requirements model (see Table 2). Also, if a softgoal contributes to other one, the softgoal must be added to the list too.

Requirements	S1	S2	S3	S4	S5	S6
“Blind review process”	Help	Break	Hurt	Help	-	-
“Download papers without authors’ name”	-	-	-	-	Help	Some-
“Normal Review Process”	Some-	Make	Help	-	-	-
“Download paper with authors’ name”	-	-	-	Hurt	Some-	Help
“View review process status”	-	-	-	-	-	Help
“Obtain more complete info”	-	-	Help	-	-	-

Table 2. The requirements contributions to the softgoals

Table 2 highlights in bold the requirement to be removed. This table shows a requirements list (FR and NFRs) and their type of contributions to the softgoals where S1 corresponds to softgoal “Be fair in review” from requirements model, S2 to “Review process easier”, S3 represents “Accurate review process”, S4 conforms to “Avoid possible conflicts of interest”, S5 it’s the “Privacy be maximized” softgoal and S6 refers to “Obtain more complete info”.

The next step is to identify the number of softgoals affected by the requirement to be removed. If necessary, a list of the softgoals that receive a contribution from the requirement to be removed is made. In this example, the requirement to be removed is “Download papers without authors’ name”, this one affects two softgoals: “Privacy be maximized” and “Obtain more complete info” S5 y S6 respectively (see Table 2).

For each softgoal that receives a contribution from the requirement to be removed, we search for a non-implemented requirement of which contribution compensates the possible elimination of the requirement to be removed. To do this, it is necessary applying the heuristics defined in the section “Impact Analysis Algorithm for Goal-oriented requirements in web engineering”.

For example, the softgoal “Privacy be maximized”, according to Table 2, receives a strongly-positive contribution (Help) from the requirement to be removed, thus being necessary searching for a non-implemented requirement to contribute to this softgoal. In this case, only the requirement “Download papers with authors’ name” contributes (negatively) to this softgoal (weakly-negative, i.e. Some -). Therefore, applying the heuristics described in the section “Impact Analysis Algorithm for Goal-oriented requirements in web engineering”, specifically the heuristic number 2 (H2), the requirement “Download papers with authors’ name” could be implemented.

Considering the softgoal “Obtain more complete info” according to Table 2, it receives a weakly-negative contribution (Some -) from the requirement to be removed, thus being necessary searching for a non-implemented requirement to contribute to this softgoal. In this case, two requirements (positively) contribute to this softgoal, “Download papers with authors’ name” and “View review process status” (strongly-negative, i.e. Help). Therefore, the heuristic H3 applies for this softgoal, thus, these requirements should be implemented.

After analyzing the softgoals contributions, the next step is searching for any softgoal in the requirements list that contributes to another softgoal. In this example, the softgoal “Obtain more complete info” makes a strongly-positive contribution (Help) to the softgoal “Accurate review process”, thus, the next step consists of searching for the requirement that makes a contribution to the softgoal and applying the heuristics. Therefore, the requirement that makes a contribution to the softgoal “Accurate review process” is “Normal review process”, this contribution is strongly-positive (see Figure 2), hence, according to H3 this requirement must be implemented.

After having performed the algorithm the requirements that could be implemented in order to fully-satisfy the goal “Process of review papers be selected” after having removed the requirement “Download papers without authors’ name”; these are (i) “Normal Review Process”, (ii) “Download paper with authors’ name” and (iii) “View review process status”. Next, it is necessary to evaluate the heuristics assigned to each requirement to know what could be implemented. To do this, it is necessary to evaluate the contribution type of each requirement, i.e., the navigational requirement “Download papers with authors’ name” negatively contributes (Some -) to the softgoal “Privacy be maximized”, thus hurting the softgoal “Avoid possible conflicts of interest” (Hurt) and helping the softgoal “Obtain more complete info” (Help). Therefore, by using the human judgment the navigational requirement “Download papers with authors’ name” can be implemented. For the navigational requirement “Normal review process”,

it is easier to determine whether it can be implemented because it only contributes to one softgoal, the “Accurate review process”, hence its contribution is Help, and therefore this requirement must be implemented. Finally, the navigational requirement “View review process status” positively contributes to the softgoal “Obtain more complete info”; consequently this requirement must be implemented.

The final step is to apply the postcondition from the section “Impact Analysis Algorithm for Goal-oriented requirements in web engineering”. In this running example, according to the postcondition, it is necessary to implement the navigational requirements “View papers info” and “View Accepted/Rejected papers” because these requirements are associated with the navigational requirement “View Review Process Status”. In addition, the content requirement “Authors” and the service requirement “Send Comments to Authors” must be implemented too in order to implement the alternative path to fully satisfy the goal “Process of review papers be selected”. Hence, the content requirement “Authors” is associated with the navigational requirement “View Accepted/Rejected papers” and the service requirement “Send Comments to Authors” is related with the navigational requirement “Normal review process”.

After finishing the execution of the algorithm, we obtain the requirements that are directly and indirectly affected by the deletion of the requirement “Download papers without authors’ name”. Moreover, the algorithm can find out which requirements must be implemented to continue satisfying the goal considering the contributions received from the softgoals. In this running example the requirements to implement are: “Download papers with authors’ name”, “Normal review process” and “View review process status”. Finally, according to the post-condition the requirements “View papers info”, “View Accepted/Rejected papers”, “Authors” and “Send Comments to Authors” must be implemented too. Figure 5 shows the final requirements model with the alternative path implemented to fully-satisfy the goal “Process of review papers be selected”.

OPEN SOURCE IMPLEMENTATION FRAMEWORK

In this section we describe in detail the implementation of the impact analysis algorithm within our approach for goal-oriented requirements analysis in Web engineering. To this aim, we have combined a set of technologies such as Eclipse (Eclipse, 2011), EMF (Eclipse Modeling Framework) (EMF, 2011), GMF (Graphical Modeling Framework) within the GMP (Graphical Modeling Project) (GMP, 2011) and Java (JAVA, 2011).

Eclipse is an open source IDE (Integrated Development Environment) used as a software platform to create integrated development environments; within Eclipse, the EMF (Eclipse Modeling Framework) project is a modeling framework and code generation facility for building tools and other applications based on a structured data model (abstract syntax). Also, the facilities for creating metamodels and models are provided by the metametamodel Ecore. Therefore, by using the facilities offered by EMF, it is possible to create a visual representation of the elements defined within the EMF metamodel by means of GMP (concrete syntax). The Eclipse Graphical Modeling Project (GMP) provides a set of generative components and runtime infrastructures for developing graphical editors based on EMF and GEF (Graphical Editing Framework). Both the Eclipse Modeling Framework (EMF) and the Graphical Modeling

Framework (GMF) are capable of generating editor plug-ins. Next, each one of the steps performed for the implementation framework is described.

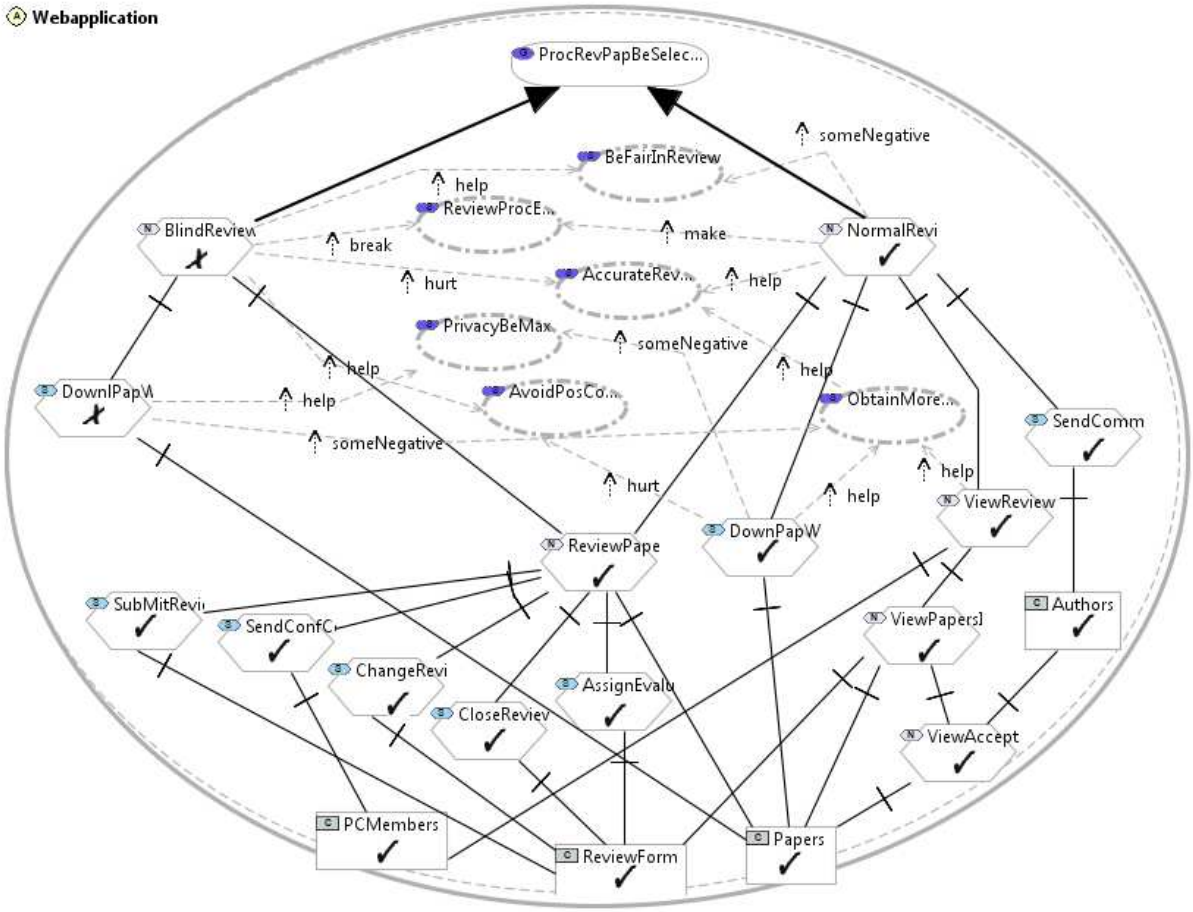


Figure 5. The Conference Management System requirements expressed in a SR and SD Models with the new requirements to implement (the alternative path "Normal Review Process")

The first step is the implementation of the Web requirements metamodel. The requirements metamodel was created using the EMF metamodel to incorporate a number of taxonomic features for the specification of Web requirements. With the implementation of this metamodel has been possible to adapt the i* modeling framework in the Web domain, with which is possible to model the needs and expectations of the stakeholders of the Web application. The clasification of Web requirements presented in (M.J. Escalona & Koch, 2004), and previously defined, have been incorporated as Ecore classes to represent each type of the requirements clasification. Figure 6 shows a screenshot where the reader can see the implementation of this metamodel in Eclipse IDE by means of the EMF.

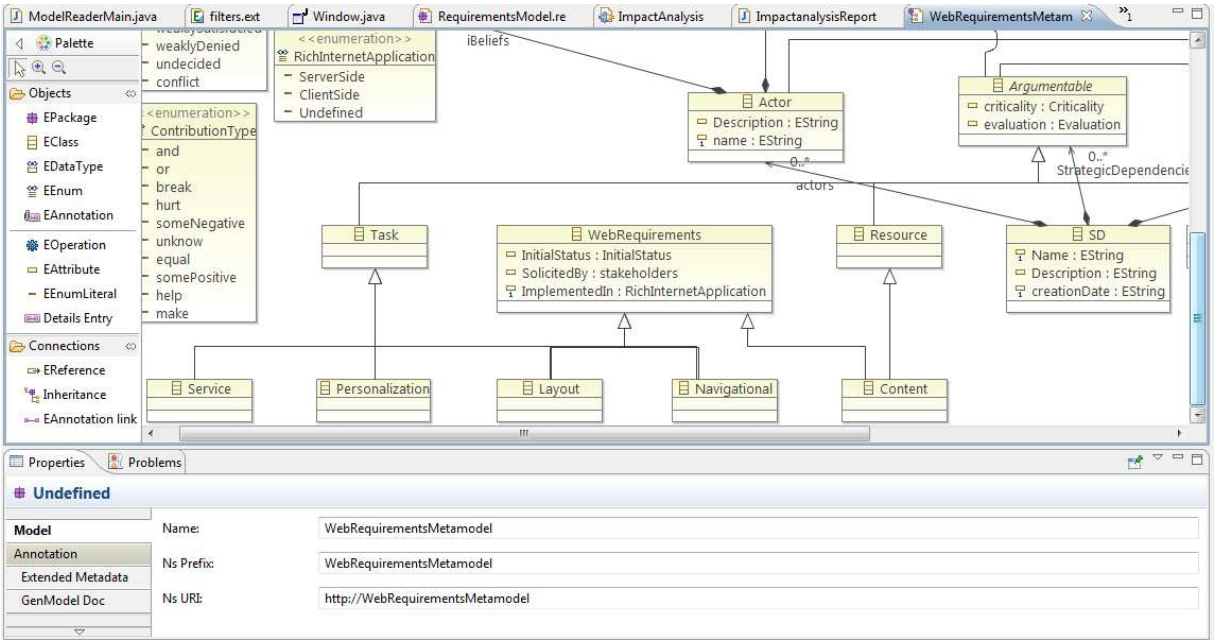


Figure 6. Screenshot of the Web requirements metamodel in Eclipse EMF

Once the metamodel has been implemented it is necessary to provide a graphical tool to assist the designer with the requirements specificaton. To do so, we have implemented a graphical editor using the GMF technology from the Eclipse Graphical Modeling Project.

GMF is a framework composed by three models: (i) Domain Model, (ii) Graphical Definition Model and (iii) Tool Definition Model. The Domain Model (ecore/genmodel files) is the starting point for the development of most of the Eclipse-based applications. The Domain Model represents the abstract syntax of the application domain, with this model the domain objects are represented as EMF classes. The Graphical Definition Model is a list of figures and shapes described in gmfgraph files, which will be used in the diagram to display classes from the domain model in a particular form defined by the designer. The Tool Definition Model (gmftool file) is the visual representation for the tools that will be avaliable in the final editor; basically, it defines what text you want to display on the tool palette and the button's tool tip. These three models are combined in the Mapping Model (*gmfmap* file) with which GMF knows what action take place when the designer selects a tool (Tool Definition Model), which classes need to be created (from the Domain Model), and what figures its necessary to render (Graphical Definition Model) when those classes are added to the diagram editor by the designer. Finally, once these models are combined in the Mapping Model, GMF transforms this model in the Diagram Generator Model. Then, it is generated a *gmfgen* file and the diagram application code.

In Figure 7, it is displayed a screenshot of the graphical editor implemented (called WebREd) (WebREd, 2011) by combining the Web requirements metamodel and the GMF technology. In the center of the figure is described the requirements specification for the the Conference Management System from the case study presented in the section “Goal-oriented requirements analysis in web engineering”.

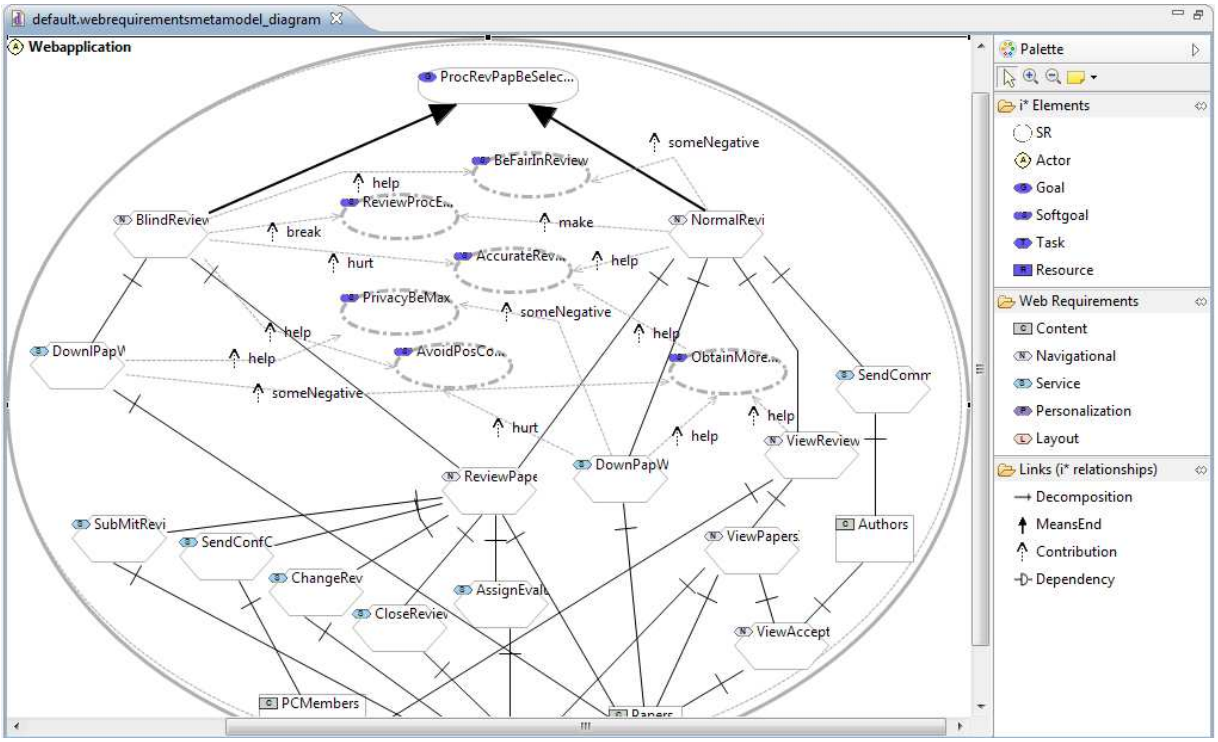


Figure 7. Screenshot of the Web requirements editor (WebREd)

As the reader can see, we have implemented each one of the elements of the Web requirements metamodel in the tool, thus the designer can model each requirement type from the clasification adopted (described in the section "Goal-oriented requirements analysis in Web engineering"). Also, WebREd provides the basic elements to model clasical goal-oriented models such as Task, Resource, Goal and Softgoal. The palette for drawing the different elements of the i* models for requirements specification can be seen on the right side of the Figure 7. At the top of the figure, there are those elements required to specify goal-oriented models according to the i* notation. Also, the requirements clasification adopted to specify requirements in the Web domain are in the right-center of the figure. Finally, the different types of relationships used in the i* modeling framework, with which the elements can be associated, are the right-botton of the figure.

With regard to the impact analysis support, this is performed in an automatic manner. When the designer specifies the requirements of the Web application by using the WebREd editor it is possible to know which requirements are affected due to a change in the Web application conceptual models. The impact of a change in the requirements can be consulted by the designer by means of a screen (window) and by means of a PDF (Portable Document File) report.

In Figure 8, it is shown a screenshoot of the impact analysis support offered by the WebREd editor. At the top of the figure, the main window for the impact analysis option is displayed. At the top we find the name of the element affected by a change originated in any of the Web application conceptual models described as "Element to remove". Also the information about the model currently selected is shown in the main window. In this particular case, this information is about the requirements model, thus including the name, description and creation date of the requirements model. Next, there is a tabbed pane with two options for the designer. The first one

shows a list of the requirements affected if and only if the requirement “Authors” is removed; also, is showed the type of each one of the affected requirements. Moreover, if the designer selects one of the requirements listed by double clicking on it, a new window (message dialog) is showed with a list of the softgoals affected by the requirement selected from the list (Figure 8, down). This softgoal's list shows the strength of the contribution made by the requirement affected to the softgoals. On the other hand, in the second tab, the requirements to be implemented (only when it is necessary) by the designer to still continue satisfying the goal are listed.

To conclude, the main window for the impact analysis option in the WebREd editor allows the designer to print a report in PDF (Portable Document File) format with which the designer can check the affected requirements.

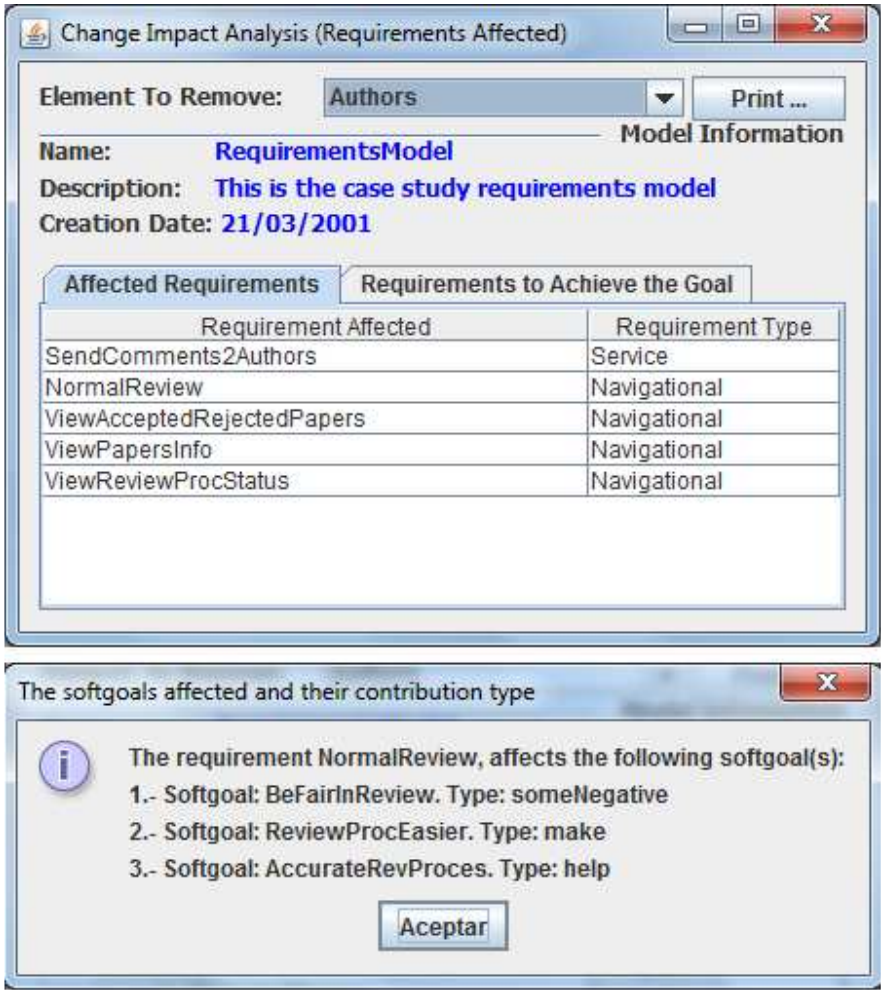


Figure 8. Screenshot of the impact analysis support offered by the Web requirements editor (WebREd)

CONCLUSION

Due to the dynamic idiosyncrasy of the Web and its heterogeneous audience Web applications should consider a requirement analysis phase in order to reflect, from the early stages of the Web application development process, specific needs, goals, interests and preferences of each user or user type, and due to the fast evolution of the Web, possible changes in those requirements should also be managed.

In this work, we have presented a methodology based on the i* modelling framework to specify Web requirements. Moreover, an algorithm to analyze the impact derived from a change done in the requirements model is presented.

Benefits of applying the algorithm described in this work include both the analysis of the impact derived from a change in a conceptual model and the ability to find an alternative path to fully-satisfy the goal by means of the softgoals tradeoff.

Nevertheless, according with the work presented in (Ameller, Franch, & Cabot, 2010), the softgoals are not considered with sufficient importance from the early stages of the development process. In this context, our proposal makes a contribution to the requirements analysis field considering the softgoals; hence it allows the designer to make decisions from the very beginning stages of the development process that would affect the structure of the envision website in order to satisfy users needs. Therefore, the designer can improve the quality of the requirements model analyzing the balance of the softgoals with the stakeholders.

Our short-term future work includes the definition of a metamodel to help to record the relationships among FR and NFRs, thus adapting the tradeoff algorithm presented in this work. Also, the impact analysis report will be completed with the adition of the requirements that the designer will need to implement to satisfy the goal. Besides, it is important to remark that the graphical editor is the basis for a prototype tool for the development of Web applications using the MDA (Model-Driven Architecture) paradigm.

Finally, note that this work has been done in the context of the A-OOH modeling method; however it can be applied to any Web modeling approach.

BIOGRAPHIES

José Alfonso Aguilar is a PhD candidate at the University of Alicante and is subventioned by the CONACYT (Consejo Nacional de Ciencia y Tecnología) and University of Sinaloa from Mexico. He has published papers about model-driven engineering and requirements engineering in national and international workshops and conferences, (such as WEBIST, ICCSA, WISM within ER, JISBD and so on) and in the Journal of Universal Computer Science (J.UCS). His Research interests are: Web engineering (WE), requirements engineering (RE), model-driven Web Engineering (MDWE). Other areas of interest: Content Management Systems (CMS), Geographic Information Systems (GIS), Web 2.0, Agile Software Development Methodologies, Process Improvement & CMMi.

Irene Garrigós is an assistant professor and post-doc researcher at the Department of Software and Computing Systems in the University of Alicante, (Spain), from which she holds a PhD and a Master in Computer Science. She has published several papers in national and international workshops, conferences and journals (such as ICWE, ER, WISE, APWEB, JISBD, information and software technologies, journal of Web engineering, and so on). Dr. Garrigós has served as a Program Committee member of several workshops and conferences such as ER, JISBD, WISM, MDA, FPUML, UWA and has served as assistant referee in several international conferences such as WWW and ICWE. She has done research stays in Belgium (Vrije Universiteit Brussel) and the Netherlands (Technische Universiteit Eindhoven). Her research interests are: Web engineering, personalization, model driven development, requirement engineering, Web and business intelligence, adaptive systems. She was involved in the organization of the 1st and 2nd edition of WeRE workshop (held in the RE and ICWE conferences, respectively) and the 1st and 2nd edition of the BEWEB workshop (held in the EDBT conference).

Jose-Norberto Mazón is an assistant professor at the Department of Software and Computing Systems in the University of Alicante (Spain). He obtained his Ph.D. in Computer Science from the University of Alicante (Spain). He has published several papers about data warehouses and requirement engineering in national and international workshops and conferences, (such as DAWAK, ER, DOLAP, BNCOD, JISBD and so on) and in several journals such as Decision Support Systems (DSS), SIGMOD Record or Data and Knowledge Engineering (DKE). He has also been co-organizer of the International Workshop on Business intelligencE and the WEB (BEWEB 2010 and 2011) and the International Workshop on The Web and Requirements Engineering (WeRE 2010 and 2011). His research interests are: requirement engineering, business intelligence, and model driven development.

REFERENCES

Aguilar, J. A., Garrigós, I., Mazón, J.-N., & Trujillo, J. (2010). *Web Engineering Approaches for Requirement Analysis - A Systematic Literature Review*. Paper presented at the 6th Web Information Systems an Technologies (WEBIST).

Aguilar, J. A., Garrigós, I., Mazón, J.-N. (2011). *Impact Analysis of Goal-Oriented Requirements in Web Engineering*. Paper presented at the 11th Computational Science and Its Applications (ICCSA).

Aguilar, J. A., Garrigós, I., Mazón, J. N., & Trujillo, J. (2010). An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering. *Journal of Universal Computer Science, 16*(17), 2475-2494.

Ameller, D., Franch, X., & Cabot, J. (2010). *Dealing with Non-Functional Requirements in Model-Driven Development*. Paper presented at the Proceedings of the 18th IEEE International Requirements Engineering Conference.

Arnold, R. S., & Bohner, S. A. (1993). *Impact Analysis - Towards a Framework for Comparison*. Paper presented at the Proceedings of the Conference on Software Maintenance.

ATL. (2011). The Atlas Transformation Language. 2011, from http://www.eclipse.org/atl

Boehm, B., & In, H. (1996). Identifying Quality-Requirement Conflicts. *IEEE Softw., 13*(2), 25-35.

Bolchini, D., & Mylopoulos, J. (2003). *From Task-Oriented to Goal-Oriented Web Requirements Analysis*. Paper presented at the Proceedings of the 4th International Conference on Web Information Systems Engineering.

Bolchini, D., & Paolini, P. (2004). Goal-driven requirements analysis for hypermedia-intensive Web applications. *Requir. Eng., 9*(2), 85-103.

Casteleyn, S., Woensel, W. V., & Houben, G.-J. (2007). *A semantics-based aspect-oriented approach to adaptation in web engineering*. Paper presented at the Proceedings of the 18th conference on Hypertext and hypermedia.

Ceri, S., Fraternali, P., & Bongio, A. (2000). Web Modeling Language (WebML): a modeling language for designing Web sites. *Comput. Netw., 33*(1-6), 137-157.

Del Fabro, M., Bézivin, J., & Valduriez, P. (2006). *Weaving Models with the Eclipse AMW plugin.* Paper presented at the Eclipse Modeling Symposium, Eclipse Summit Europe, Esslingen, Germany.

Eclipse. (2011). Eclipse Foundation Open Source Community. 2011, from http://www.eclipse.org/

Elahi, G., & Yu, E. (2009). Modeling and analysis of security trade-offs - A goal oriented approach. *Data Knowl. Eng., 68*(7), 579-598.

EMF. (2011). The Eclipse Modeling Framework Project. 2011, from http://www.eclipse.org/emf

Escalona, M. J., & Aragón, G. (2008). NDT. A Model-Driven Approach for Web Requirements. *IEEE Trans. Softw. Eng., 34*(3), 377-390.

Escalona, M. J., & Koch, N. (2004). Requirements engineering for web applications-a comparative study. *Journal of Web Engineering, 2*, 193-212.

Garrigós, I. (2008). *A-OOH: Extending Web Application Design with Dynamic Personalization.* Unpublished Ph. D, University of Alicante.

Garrigós, I., Mazón, J.-N., & Trujillo, J. (2009). *A Requirement Analysis Approach for Using i* in Web Engineering*. Paper presented at the Proceedings of the 9th International Conference on Web Engineering.

Ginige, A. (2002). *Web engineering: managing the complexity of web systems development*. Paper presented at the Proceedings of the 14th international conference on Software engineering and knowledge engineering.

GMP. (2011). The Graphical Modeling Project. 2011, from http://www.eclipse.org/emf

Gómez, J., Cachero, C., & Pastor, O. (2000). Extending a conceptual modelling approach to web application design. *Lecture Notes in Computer Science, 1789*, 79-93.

Gupta, C., Singh, Y., & Chauhan, D. S. (2010). Dependency based Process Model for Impact Analysis: A Requirement Engineering Perspective. *International Journal of Computer Applications IJCA, 6*(6), 28-33.

Horkoff, J., & Yu, E. (2009a). Evaluating Goal Achievement in Enterprise Modeling–An Interactive Procedure and Experiences. *The Practice of Enterprise Modeling*, 145-160.

Horkoff, J., & Yu, E. (2009b). *A Qualitative, Interactive Evaluation Procedure for Goal-and Agent-Oriented Models.* Paper presented at the 21st International Conference in Advanced Information Systems Engineering (CAiSE), Amsterdam, The Netherlands.

JAVA. (2011). JAVA Standar Edition (SE). 2011, from http://www.oracle.com/technetwork/java/javase/overview/index.html

Jouault, F. (2005). *Loosely coupled traceability for atl.* Paper presented at the European Conference on Model Driven Architecture (ECMDA) workshop on traceability, Nuremberg, Germany.

Koch, N., Knapp, A., Zhang, G., & Baumeister, H. (2008). Uml-Based Web Engineering. In G. Rossi, O. Pastor, D. Schwabe & L. Olsina (Eds.), *Web Engineering: Modelling and Implementing Web Applications* (pp. 157-191): Springer London.

Lindvall, M., & Sandahl, K. (1998). How well do experienced software developers predict software change? *J. Syst. Softw., 43*(1), 19-27.

Melia, S., Cachero, C., & Gomez, J. (2003). *Using MDA in web software architectures.* Paper presented at the Proceedings of the 2nd International Workshop on Generative Techniques in the Context of MDA., Anaheim, California, USA.

Molina, F., Pardillo, J., & Toval, A. (2008). *Modelling Web-Based Systems Requirements Using WRM*. Paper presented at the Proceedings of the 9th international workshops on Web Information Systems Engineering.

Molina, F., & Toval, A. (2009). Integrating usability requirements that can be evaluated in design time into Model Driven Engineering of Web Information Systems. *Advances in Engineering Software, 40*(12), 1306-1317.

Nuseibeh, B., & Easterbrook, S. (2000). *Requirements engineering: a roadmap*. Paper presented at the Proceedings of the Conference on The Future of Software Engineering.

Pastor, O., Fons, J., Pelechano, V., & Abrahão, S. (2006). Conceptual Modelling of Web Applications: The OOWS Approach. In E. Mendes & N. Mosley (Eds.), *Web Engineering* (pp. 277-302): Springer Berlin Heidelberg.

Schwabe, D., & Rossi, G. (1995). The object-oriented hypermedia design model. *Communication of the. ACM, 38*(8), 45-46.

Sommerville, I. (2005). *Software Engineering* (7th ed.): Addison Wesley.

Troyer, O. M. F. D., & Leune, C. J. (1998). WSDM: a user centered design method for Web sites. *Comput. Netw. ISDN Syst., 30*(1-7), 85-94.

WebREd. (2011). The Web Requirements Editor. 2011, from http://code.google.com/p/webred/

Yu, E. (1995). *Modelling strategic relationships for process reengineering.* Unpublished phD, University of Toronto Ontario, Canada.

Yu, E. (2002). *Towards modelling and reasoning support for early-phase requirements engineering*. Paper presented at the 10th Requirements Engineering (RE).

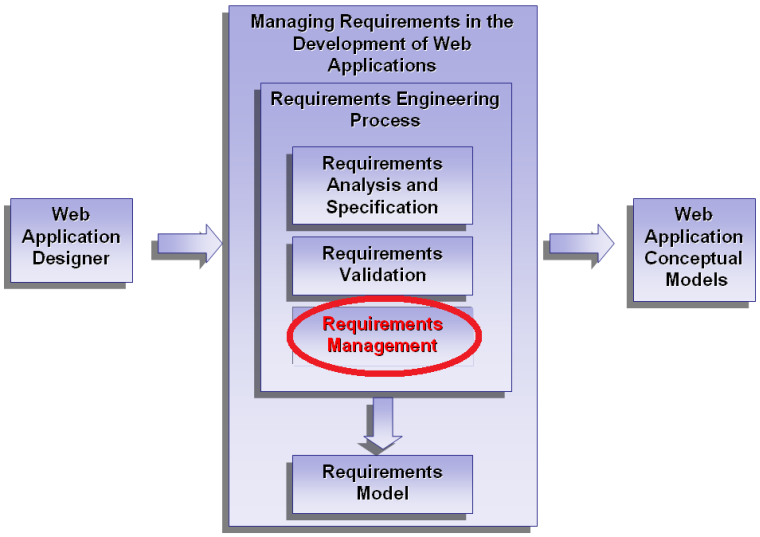
Zhang, W., Mei, H., & Zhao, H. (2005). *A Feature-Oriented Approach to Modeling Requirements Dependencies*. Paper presented at the Proceedings of the 13th IEEE International Conference on Requirements Engineering.

Zhao, J. (2002). *Change impact analysis for aspect-oriented software evolution*. Paper presented at the Proceedings of the 7th International Workshop on Principles of Software Evolution.

C

A Goal-Oriented Requirements Engineering Approach to Distribute Functionality in RIAs

El contenido de este apéndice ha sido enviado a 24th International Conference on Advanced Information Systems Engineering (CAiSE 2012)



En este trabajo, se presenta la adaptación de la propuesta descrita en el Capítulo 5 para auxiliar al diseñador Web en la distribución de la funcionalidad de la RIA entre el cliente y el servidor. Para lograrlo, se adaptó el algoritmo Optimización de Pareto para obtener un conjunto de soluciones óptimas, de entre las cuales, de acuerdo con la prioridad establecida por parte del *stakeholder* sobre los requisitos no-funcionales, el diseñador de la aplicación Web será capaz de optimizar los requisitos no-funcionales mediante la distribución de los requisitos funcionales entre el cliente y el servidor.

A Goal-Oriented Requirements Engineering Approach to Distribute Functionality in RIAs

José Alfonso Aguilar[1,3], Sven Casteleyn[2], Irene Garrigós[1], and Jose-Norberto Mazón[1]

[1]Department of Software and Computing Systems
University of Alicante, Spain
[2]Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València, Spain
[3]Computer Science Faculty
University of Sinaloa, Mexico
{ja.aguilar,igarrigos,jnmazon}@dlsi.ua.es
sven.casteleyn@upv.es

Abstract. The challenges and complexity involved in the design and development of Rich Internet Applications are well recognized, and most existing Web 1.0 design methodologies have been augmented to tackle these. However, the focus is mostly on technological and engineering challenges, while only providing limited support for the requirements analysis. This paper addresses this hiatus by extending an existing goal-oriented Web requirements analysis approach to support RIAs. Specifically, we present an approach (based on the Pareto efficiency) to evaluate and select desirable client/server distribution of functionality for RIAs, and guide the designer towards an optimal solution that provides a balanced maximization of the non-functional requirements. We illustrate our approach by using an excerpt of a real-world case study of a bioinformatics company.

Keywords: Web Engineering, Rich Internet Applications, Goal-Oriented Requirements Analysis

1 Introduction

Due to the Web's highly heterogeneous user base and its constant evolution, requirements specification is an important yet difficult task in the Web engineering field. Importantly, a key factor in a Web application's success is to ensure that the functionality required by the users is fulfilled, while at the time facilitating their browsing session [10]. This is tightly related to both (i) functional requirements (FRs) that describe the system services, behavior or functions, e.g., "download a sales report"; and (ii) non-functional requirements (NFRs) which specify a constraint over that functionality, e.g., "usability" [17]. Therefore, Web methodologies should be able to provide a requirements analysis phase in which functional and non-functional requirements are considered.

Unfortunately, both in industrial cases and in Web engineering methodologies, requirements are not sufficiently taken into account when developing a Web application [3]. For stakeholders[1], this results in a mismatch between their expectations and the resulting Web application, and for users in a Web application that poorly supports their requirements and does not fulfill their needs.

As a natural evolution of the Web, Rich Internet Applications (RIAs) enable client-side logic, offer better responsiveness and interactivity, and provide more elaborate and attractive presentation. Their main goal is to tackle the limitation imposed by traditional Web 1.0 technology [6], and provide the user with a richer experience, similar to that of desktop applications. While offering more possibilities to satisfy the user's expectations, RIAs are also considerably more complicated to design and develop. Existing Web design methodologies have been extended to support RIAs, but the proposed extensions mainly focus on presentation issues or interaction capabilities [13]. Requirements analysis is largely left to the discretion of the designer and no systematic, tailored support for RIAs is provided. Nevertheless, this is not a trivial task, as the designer needs to balance classical Web requirements, such as those functionalities (i.e., FR) related to navigation, and RIA specific NFR, such as "Responsiveness" or "Bandwidth reduction". Furthermore, the designer needs to take into account that the distribution of functionality between client and server influences the satisfaction of the NFRs; the choice where to place certain functionality is thus critical.

In our previous work [2, 11], we presented a GORE (Goal-Oriented Requirements Engineering) approach based on the i^* modeling framework to specify Web requirements. In this paper we extend that approach to support requirements engineering in RIAs. Specifically, we focus on studying the requirements distribution between client and server in order to provide a balanced maximization of the satisfaction of NFRs. For this purpose, we present a proposal that allows the designer to obtain and evaluate good client/server configurations (requirements distribution), i.e. those that maximize the NFRs. Our proposal is based on the Pareto efficiency, which is useful when there are multiple competing and conflicting objectives [5] that need to be balanced, as is the case here. In essence, a group of non-dominated optimal solutions (called Pareto front) is created, i.e., solutions that optimize the NFRs (corresponding to *softgoals* in our GORE framework) in such a way that no single NFR can still be improved, without negatively affecting another. From these balanced optimizations, the designer can select the final solution by taking into account the importance of the different NFRs. We present our results by using an excerpt of a real-world case study of a start-up bioinformatics company in order to effectively show how our approach can aid designers to obtain an adequate client/server distribution of functionality in RIAs.

This paper is structured as follows: Section 2 presents the related work. Section 3 describes our previous work on GORE for the Web 1.0. The contribution

[1] Individuals or organizations who affect or are affected directly or indirectly by the development project in a positive or negative form [17].

of this work is presented in Section 4. Section 5 describes a real world case study. Section 6 presents the ongoing work related to the implementation of our approach. Finally, the conclusion and future work are presented in Section 7.

2 Related Work

Research in Web engineering has shown that RIA development is a difficult challenge which requires extending the traditional Web engineering approaches [14], i.e., to improve the development process and adapt the conceptual models in order to support richer presentation and interactive user interfaces.

Several Web 1.0 Web engineering approaches have been extended to support RIAs, i.e., (i) **WebML** (Web Modeling Language) [4], (ii) **OOWS** (Object-Oriented Approach for Web Solutions Modeling) [19], (iii) **OOHDM** (Object-Oriented Hypermedia Design Method) [18], and (iv) **UWE** (UML-based Web Engineering), which has three extensions for RIA development (UWE-Patterns [9], UWE-R [12] and UWE-RUX [15]). For a complete analysis of the techniques used for requirements analysis by the approaches described in this paragraph we refer the reader to [3].

Some extension have been created in order to be combined with Web engineering approaches as the ones aforementioned, these are: (i) the **RUX-Model** [16], solely for the design of UIs for RIAs over existing HTML-based Web Applications in order to give them multimedia support, and (ii) the **OOH4RIA** [13], which defines a model-driven development process to obtain a complete RIA for the GWT (Google Web Toolkit) framework, and is supported by a CASE tool.

On the other hand, new methods have been created exclusively for the development of RIAs. **IAML** (Internet Application Modelling Language) [21] is a modelling language that directly uses RIA concepts as first-class modelling citizens [20]. IAML has a CASE tool for the development of RIAs based on EMF (Eclipse Modeling Framework) and GMF (Graphical Modeling Framework). The **ADRIA** (Abstract Design of Rich Internet Applications) [7] method is an UML-based approach, whose design activities depart from an object-oriented analysis, and focus on the design of events triggered by user interactions.

In some of these approaches, e.g. OOH4RIA, IAML, RUX, the requirements engineering phase is not considered at all. In others, classical software engineering techniques are used, such as use cases in UWE, WebML, OOWS, OOHDM and ADRIA, activity diagrams in UWE, WebML and OOHDM, or task models in OOWS. These techniques are inherited from their respected Web 1.0 methods, and are not well-adapted to take RIA's requirements into account, or deal with RIA specificities (i.e., client/server distribution). Our approach complements these techniques, is method-independent, and allows to focus on satisfying user goals and objectives by means of our goal-oriented approach, while taking specificities of RIAs, in particular the client/server distribution, into account.

Finally, some interesting goal-oriented techniques for requirements analysis have been proposed for non-Web engineering approaches, i.e., [8]. This work evaluates i^* models based upon an analysis question (what-if analysis) and the

human judgment. To this aim, this procedure uses a set of evaluation labels to represent the satisfaction or denial level of each element in the i^* model. Unfortunately, these general approaches have not been adapted to Web engineering.

3 Specifying Requirements in Web Engineering

In this section we summarize our requirements analysis approach for Web 1.0 applications based on the i^* modeling framework [1, 2, 11]. This approach is the starting point of our approach for analyzing the client/server distribution of functionality when RIA-specific requirements are considered.

The development of Web applications involves different kinds of stakeholders with different needs, which depend on each other to achieve their goals, perform several tasks or obtain some resources. For example, the Web administrator relies on new clients for obtaining data in order to create new accounts. The i^* framework [22] is a GORE framework used to explicitly analyze and model these relationships among multiple stakeholders (actors in the i^* notation), along with their goals, tasks and resources. Essentially, the i^* framework consists of two models: the strategic dependency (SD) model, to describe the dependency relationships (represented as ⊣◗⊢) among various actors in an organizational context, and the strategic rationale (SR) model, used to describe actor goals and interests and how they might be achieved. Therefore, the SR model (represented as a dashed circle ◌) provides a detailed way of modeling the intentions of each actor (represented as a circle ○), i.e., internal intentional elements and their relationships:

- A goal (elipse ⬭) represents an (intentional) desire of an actor. Interestingly, goals provide a rationale for requirements but they are not enough for describing how the goal will be satisfied. This can be described through means-end links (⊣▷) representing alternative ways for fulfilling goals.
- A task (hexagon ⬡) describes some work to be performed in a particular way. Decomposition links (—┼) are useful for representing the necessary intentional elements for a task to be performed.
- A resource (rectangle ▭) represents some physical or informational entity required for the actor.
- A softgoal (eight-shape ∞) is a goal whose satisfaction criteria is not clear-cut. How an intentional element contributes to the satisfaction or fulfillment of a softgoal is determined via contribution links ($\xrightarrow[\text{hurt}]{\text{help}}$). Possible labels for a contribution link are “make”, “some+”, “help”, “hurt”, “some-”, “break”, “unknown”, indicating the (positive, negative or unknown) strength of the contribution.

Our previous work [2] presented an approach for adapting the i^* framework for specifying requirements typically encountered in Web applications (this adaptation is supported by a metamodel described in Fig. 1):

- Service requirements refer to the internal functionality the system should provide to its users, e.g., “register a new client”.

- Navigational requirements aims to define the paths needed for the user to browse the Web applications, e.g., "consult products by category".
- Layout requirements for defining the visual interface for the users, e.g., "present certain text style for headers".
- Personalization requirements allow the designer to specify the desired adaptation functionality to be performed in the final Web application, e.g., "adapt font for visual impaired users".
- Content requirements define the content that the Web application presents to its users, e.g., "product information".
- Non-Functional requirements are related to quality criteria that the intended Web system should achieve and that can be affected by other requirements, e.g., "responsiveness".

An example of an i^* model for Web engineering can be seen in Fig. 3 with explanation in Sect. 5.

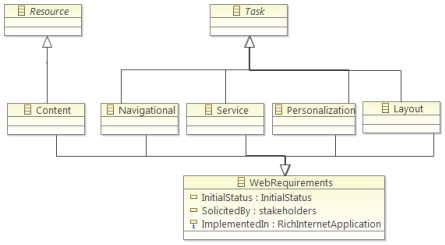


Fig. 1: An excerpt of the i^* requirements metamodel for Web domain.

Finally, it is worth noting that FRs (navigational, layout, service and personalization) are represented as tasks, while the requirements related to the data (content) are represented as resources. There is thus a clear separation between these two concerns; in this paper we focus on the (distribution of) FRs. Also, NFRs are represented as softgoals. For further details, we refer the reader to [2].

4 Optimizing RIA functionality distribution by using Pareto efficiency

A critical task in RIA requirements engineering is to determine a *good* distribution of functionality (client/server). In other words, for each FR the designer needs to decide where it will contribute best to the realization of the NFR or softgoals (according to our framework), i.e. on the client or on the server side. This is not a trivial decision, as each requirement influences different softgoals differently depending on whether it resides on the server or the client. In addition, maximizing one softgoal often leads to decreased support for another softgoal, so there is no notion of a single best client/server configuration and a tradeoff among softgoals should be made.

Pareto efficiency [5] is an ideal candidate to tackle this problem, as it is designed to balance competing concerns. It is a notion from economics widely applied to engineering, which is described as follows: given a set of alternative allocations (server or client) and a set of individuals (requirements), allocation A is an improvement over allocation B only if A can make at least one individual better (maximizing softgoals) than B, without making any other worse (weakening softgoals). Therefore, in our approach, a Pareto optimal (client/server) configuration is one so that no other configuration better satisfies one single softgoal, while satisfying the others equally. The set of Pareto optimal configurations are thus candidate *good* distributions. It includes the solutions that best satisfy each single softgoal, but also those that provide an optimal tradeoff between softgoals. As such, it forms a perfect basis to make a well-informed design decision for functionality client/server distribution.

More formally, finding the set of Pareto optimal configuration can be defined as the problem of finding a (decision) vector of decision variables X (i.e., a valid client/server distribution), which maximizes a vector of M objective functions $f_i(X)$, (i.e., the satisfaction of softgoal i in client/server distribution X, where $i \in \{1, ..., M\}$ (with M the amount of softgoals). To do so, the concept of domination between vectors is defined as follows: a decision vector X is said to dominate a decision vector Y ($X \succ Y$) if and only if their corresponding objective vectors of objective functions $f_i(X)$ and $f_j(X)$ satisfy: $\forall i \in \{1...M\} : f_i(X) \geq f_i(Y)$ and $\exists i \in \{1, ..., M\} : f_i(X) > f_i(Y)$. We then say that all decision vectors that are not dominated by any other decision vectors form the Pareto optimal set, while the corresponding objective vectors are said to form the Pareto front.

Following the Pareto efficiency, we defined the following steps to determine the Pareto optimal configuration for distributing FRs between client and server, while the softgoals are balanced and maximized (Figure 2). Our approach supports bith scenarios: (i) the evolution of an existing Web 1.0 specification to support RIAs, and (ii) from-scratch specification.

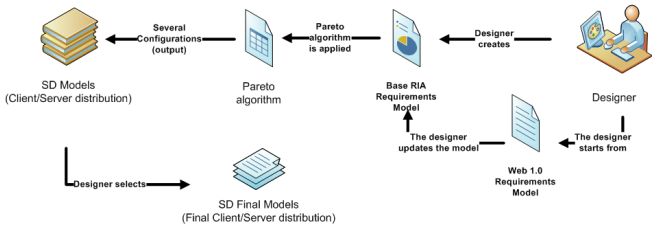


Fig. 2: An overview of our RIA requirement analysis approach.

Before defining each step of our approach for dealing with the distribution of functionality, it is worth noting that the distribution of data is also a relevant requirement when developing RIAs, and in fact, both are coupled. Indeed, when deciding to implement particular functionality on the client, the relevant data needed to realize this functionality needs to be duplicated on the client. A

straightforward solution is thus to duplicate in the client any resource (content requirement) that is related to a task (navigation, layout, personalization or service requirement) located in the client. We are currently applying this solution, allowing us to focus only on functionality. However, it needs to be mentioned that the duplication of data and its communication to the client causes some additional effects, which might influence, both positively and negatively, the maximization of softgoals. For example, duplication and communication of data to the client side might cause slower initialization, but on the other hand, reuse of this data for different tasks will increase responsiveness. We are therefore considering in our future work to also adopt content requirements in our Pareto algorithm to include their effect on the maximization of softgoals.

Step 1.- Create the base RIA requirements model. When no Web 1.0 artifacts are available, a Web 1.0 requirements model is created (see Sect. 3), but additionally including RIA specific requirements, softgoals and the contribution links they induce. The RIA specific requirements are those that are only realizable using RIA technology (e.g., "Provide a graphical view", "Capture webcam"). The RIA specific softgoals arise from the fact that RIA technologies are used; they would otherwise not be present, possible or relevant. Examples include "Responsiveness" or "Low bandwidth". At this point, we do not yet take into account the distribution of FRs among client and server. Hence, the label of the contribution links (help, hurt, some+, some-, make and break) cannot yet be determined, as this is dependent on where each requirement is implemented. For example, the requirement "Browse result set" will "help" the softgoal "Responsiveness" if implemented on the client, but "hurt" it when implemented on the server. Therefore, we leave the label of the contribution links as "unknown" at this moment; later, when we take into account server/client distribution, we will assign the correct label to each of the contribution links. In some cases, the label of the contribution can be directly assigned[2]: if the contribution involves a requirement that can only be realized on server or client side (i.e., "Provide graphical view" needs client-side technology and is thus only realizable on the client side) or for softgoals to whom a contribution is client or server independent (i.e., "Provide proprietary format" always "hurts" "Compatibility" as it concerns a non-standard format).

In case a Web 1.0 requirement model was already available, the model is evolved as follows: the RIA specific requirements, softgoals and contribution links (with "unknown" label) are added to the existing model. For contributions that were already present in the Web 1.0 requirement model, their label is replaced by "unknown" if it may be influenced by the client/server distribution of the requirements.

Step 2.- Obtain the Pareto optimal client/server distribution. We provide a set of sub-steps based on the Pareto efficiency, to obtain a set of valid client/server configurations that balance the maximization of softgoals, rather than maximizing one single softgoal.

[2] Note that leaving these "unknown" in this step would yield the same final result; assigning them now only simplifies the following steps.

A) Obtain a decision vector for requirements distribution. Each possible distribution of N FRs between client and server is stored in a decision vector X_v: $\forall v : 0 \preceq v < 2^N$, $\forall i \in \{1...N\} : X_{v_i} = R_i$, where X_{v_i} is the ith element of X_v, $R_i = S$ if the FR i is allocated on the server and $R_i = C$ if the FR i is allocated on the client.

B) Create a weight matrix. Depending on where the FR is implemented (either on the client or on the server), the contribution of each requirement to each softgoal must be quantified. To this aim, the designer creates a matrix by using the following weights to each kind of contribution: $w = 0$ if the requirement does not contribute to any softgoal, $w = +1$ if there is a "Help" contribution link, $w = -1$ if there is a "Hurt" contribution link, $w = +2$ if there is a "Some +" contribution link, $w = -2$ if there is a "Some -" contribution link, $w = +4$ if there is a "Make" contribution link and $w = -4$ if there is a "Break" contribution link.

Therefore, the matrix is defined, so that each entry $W_{i_j}^k$ corresponds to the contribution of the ith functional requirement to the jth softgoal on the k (client or server) side: $\forall i \in \{1...N\}$, $\forall j \in \{1...M\}$, $\forall k \in \{C, S\}$: $W_{i_j}^k = w$, where N is the number of functional requirements and M is the number of softgoals, and w is defined as previously described.

C) Calculate the objective function. For each softgoal j the corresponding objective function F_j with respect to a decision vector X_v is calculated by summing the contributions of all requirements to each softgoal j taking into account the client/server distribution defined in x_v: $\forall j \in \{1, ..., M\}$, $\forall v \in \{0, ..., 2^N - 1\}$: $F_j(X_v) = \Sigma_{j=1}^{M} W_{i_j}^k$, where N is the number of FRs, M is the number of softgoals.

D) Calculate the overall fitness. The sum of all objective functions with respect to a decision vector X_v is computed to obtain the overall fitness of the decision vector X_v: $\forall j \in \{1...N\}$, $\forall v \in \{0, ..., 2^N - 1\}$: $\Sigma_{j=1}^{M} F_j(X_v)$, where N is the number of requirements and M is the number of softgoals.

E) Calculate the Pareto front. To do this, it is necessary to obtain all decision vectors that are not dominated by any other decision vectors, these forms the Pareto optimal set, while the corresponding objective vectors are said to form the Pareto front.

Step 3- Select the final client/server distribution. From the set of Pareto optimal client/server distributions, the designer now elects the distribution of choice. When making this selection, the designer needs to take into account the priorities of the stakeholder, i.e., which softgoals are most important to realize, and which negative impact on other softgoals is an acceptable consequence. The merit of our approach is that it gives the designer a good overview of the good possible configurations. Furthermore, by using the GORE approach, he also gets a good insight in the impact, both positive and negative, his decisions have on the realization of all softgoals, i.e. on the user satisfaction. Finally, the final RIA requirements model is created using the client/server distribution selected by the designer.

5 Case Study

In this section, we illustrate our approach by presenting an excerpt of a real-world case study of a bioinformatics company. The main goal of this company is to provide online services based on human genome analysis. The company will allow clients to upload a gene, subsequently analyze the gene and provide the user with personalized reports corresponding to the desire of the user. The first and only service that initially will be available is the reporting of potential illnesses the user is vulnerable to, based on the detection of variations of the user's supplied gene compared to the company's gene database. The company is committed to offer their services using a RIA, as they expect a modern Web application will be more attractive for visitors, and they want to study the potential benefit of distributing their functionality between the client and server side. In our excerpt of the case study, we focus (solely) on the reporting requirement to illustrate our approach.

Step 1.- Create the base RIA requirements model. According to our proposal, the first step is to create the initial (base) RIA requirements model. As no Web 1.0 was available, it is necessary to specify the base RIA requirements model from scratch. To do this, the designer has to specify the company's goals and requirements.

In Figure 3 the initial requirements model for this case study is shown, focusing solely on the reporting requirement. The goal of the WebApp (actor) is "Report to be provided". To achieve this goal, some FRs are needed: the navigational requirements "Provide variations by disease" and "Provide disease by variant", the layout requirement "Provide graphical view" and, the service requirements "Provide proprietary format document" and "Provide pdf". Some of these requirements directly or indirectly affect some softgoals, i.e., the service requirement "Provide pdf" affects directly the softgoal "Compatibility" by means of a "Help" contribution and also affects the "Security" softgoal indirectly by means of the goal "Report to be provided".

As explained in Sect. 4, at this point, we do not yet decide which FR will be implemented on the server or on the client side, and thus the contributions to the softgoals are initially labeled as "unknown", as their contribution depends on where they are implemented (e.g. client or server). For the "Compatibility" softgoal however, we see that the contributions are already stated, as those values do not depend on where the requirement is implemented. Contributions originating in "Provide graphical view" are also stated, because this requirement is mandatory implemented on the client, as it can only be realized using client-side technology.

Step 2.- Obtain the Pareto optimal client/server distribution. The aim is to obtain a set of good client/server (FR) distributions. To do this, the following sub-steps are needed.

A) Obtain a decision vector for requirements distribution. To calculate this distributions, we only need to consider those FRs that contribute to the softgoals in the base requirements model, as can be seen in Table 1. Consequently, there are sixteen possible configurations (decision vectors) for the

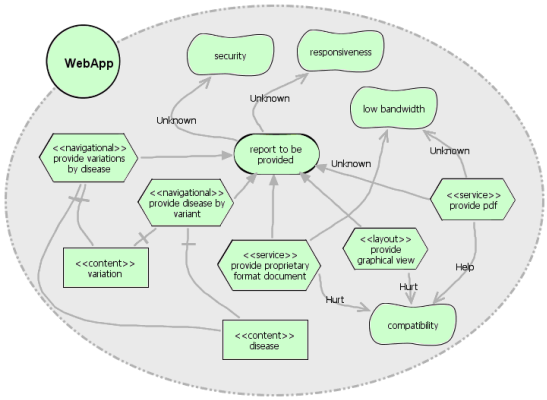


Fig. 3: Base RIA requirements model.

requirements distribution (using 2^4). Table 3 shows all possible decision vectors (column 2 to 5, all rows), in other words, all possible client/server configurations, where C represents Client and S represents Server. The requirement "Provide graphical view" is not considered as it is always allocated client side.

Table 1: *Softgoals* and FRs detected in the base RIA requirements model.

Softgoals	**Requirement**
S1.- security	R1.- provide variations by disease
S2.- responsiveness	R2.- provide disease by variant
S3.- compatibility	R3.- provide pdf
S4.- low bandwidth	R4.- provide proprietary format document

B) Create a weight matrix. Next, we create the weight matrices. For computing the objective functions in this case study, the following matrices are defined: Client matrix (Table 2a) and Server matrix (Table 2b). Each matrix contains the quantification of the contributions made by the FRs to each softgoal, both on the client and on the server side. In practice, the designer will take the base RIA requirements model, and concretize it twice: once filling in the contributions labels under the assumption that all FRs are located on the client, and once assuming they are all located on the server. The first model correspond to the Client matrix (Table 2a), the second to the Server matrix (Table 2b), using the weights defined in Section 4, 2B. As an example, row 3, (requirement "Provide pdf"), column 4 (softgoal "Responsiveness"), shows "+1" in the Client matrix, indicating a "help" contribution if the requirement is located on the client, and shows "-1" in the Server matrix, indicating a "hurt'" contribution if the requirement is located on the server side.

C) Calculate the objective function. The objective functions are calculated using the weight matrices from step 2.B, i.e., for the configuration X1 (Table 3): $F(S1) = -4$ is calculated summing the contributions made by the requirements to the softgoal S1 according to the client weight matrix (Table 2a) and $F(S2) = +4$ is calculated in the same form by using the server weight matrix

(Table 2b). For $F(S1) = -2$ from the configuration X2, the requirements R1, R2 and R3 are implemented in client side, so their contribution is taken from the client weight matrix (Table 2a): $\Sigma(R1, R2, R3) = -3$ and R4 is implemented in the server side, thus, $R4 = +1$ (taken from the server weight matrix (Table 2b), thus $F(S1) = -2$. For this case study, the results of the corresponding objective functions are shown in columns 6 to 9 from Table 3.

D) Calculate the overall fitness. All the objective functions are summed (Σ) with respect to each configuration (decision vector) to obtain the overall fitness (see column 10 from Table 3). For example, in the configuration X1 from Table 3, the overall fitness (Σ) is "+2", i.e. the sum of the softgoals $F(S1) + F(S2) + F(S3) + F(S4)$ for the decision vector X1.

Requirement	S1	S2	S3	S4
R1	-1	+1	0	0
R2	-1	+1	0	0
R3	-1	+1	+1	+1
R4	-1	+1	-1	+1

(a) Client weight matrix.

Requirement	S1	S2	S3	S4
R1	+1	-1	0	0
R2	+1	-1	0	0
R3	+1	-1	+1	-1
R4	+1	-1	-1	-1

(b) Server weight matrix.

Table 2: Weight matrices.

E) Calculate the Pareto front. The configurations (decision vectors) are Pareto front when they are not dominated by any other configurations. It is then said that these configurations form the Pareto optimal set, while the corresponding objective vectors (softgoals) are said to form the Pareto front. For example, the configuration X1 (decision vector X1) is Pareto front, because no other decision vector can be found that satisfies a particular softgoal better, without causing the satisfaction of other softgoals to decrease. Note that $F(S2)$, $F(S3)$ and $F(S4)$ are all maximized in this configuration. On the other hand, the configuration X2 is not Pareto front due to the configuration X5 that better satisfies $F(S4)$, and equally satisfies all other softgoals.

In Table 3, grey rows are the Pareto front from which we can select the final solution according to the priorities we establish over the softgoals. For example, the configuration X1, where every requirement is placed on the client side, is the best option if we do not mind that the satisfaction of softgoal S1 ("Security") is minimal (-4): all other softgoals are maximized, and the overall fitness is +2, indicating that overall, softgoals are well satisfied. On the other hand, if "Security" is the softgoal we want to prioritize, configuration X16, where every requirement is placed on the server side, is best suited. However, it comes with a high price with respect to other softgoals, and has an overall fitness of -2, indicated an overall poor satisfaction of softgoals. All the other solutions of the Pareto front are intermediate configurations that lead us to different tradeoffs such as X6 or X14. Both have an overall fitness of 0, indicating that overall, all softgoals are balanced.

Step 3.- Select the final client/server distribution. In this step, the final RIA requirements model is created using the client/server distribution selected by the designer. For the bioinformatics company, the configuration X14

Table 3: The possible functionality distribution among the server and the client by means of the softgoal tradeoff.

Configuration	R1	R2	R3	R4	F(S1)	F(S2)	F(S3)	F(S4)	Σ	Pareto Front
X1	C	C	C	C	-4	+4	0	+2	+2	Yes
X2	C	C	C	S	-2	+2	0	0	0	No because of X5
X3	C	C	S	C	-2	+2	0	0	0	No because of X5
X4	C	C	S	S	0	0	0	-2	-2	No because of X13
X5	C	S	C	C	-2	+2	0	+2	+2	Yes
X6	C	S	C	S	0	0	0	0	0	Yes
X7	C	S	S	C	0	0	0	0	0	Yes (as same as X6)
X8	C	S	S	S	+2	-2	0	-2	-2	No because of X14
X9	S	C	C	C	-2	+2	0	+2	+2	Yes (as same as X5)
X10	S	C	C	S	0	0	0	0	0	Yes (as same as X6)
X11	S	C	S	C	0	0	0	0	0	Yes (as same as X6)
X12	S	C	S	S	+2	-2	0	-2	-2	No because of X14
X13	S	S	C	C	0	0	0	+2	+2	Yes
X14	S	S	C	S	+2	-2	0	0	0	Yes
X15	S	S	S	C	+2	-2	0	0	0	Yes (as same as X14)
X16	S	S	S	S	+4	-4	0	-2	-2	Yes

(Figure 4) is selected as the final client/server distribution. In X14, the requirements R1, R2 and R4 are placed on the server and R2 on the client (see Table 1). The requirement "Provide graphical view" by default is always allocated in the client side (as mentioned in Step 2.A). This configuration was considered a good tradeoff, because security is deemed important, but other softgoals should not be totally neglected.

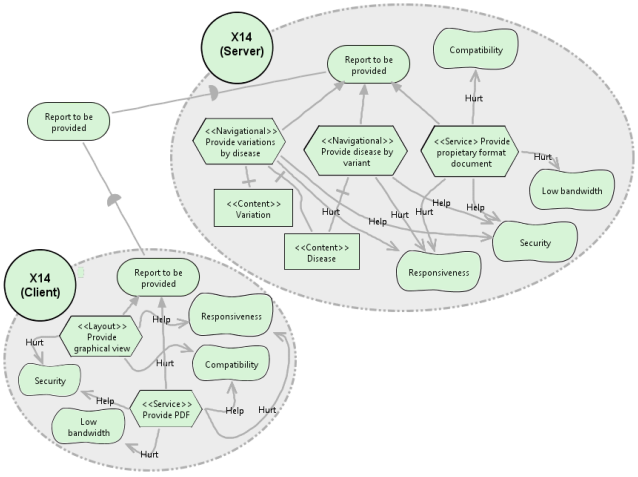


Fig. 4: X14 solution.

6 Implementation

Our approach for client/server distribution of RIA functionality has been implemented by using EMF (http://www.eclipse.org/emf/). EMF is a Java-based modeling framework and code generation facility for building tools and other

applications based on a structured data model. In our case, the left-hand side of Fig. 5 shows a representation of the i^* requirements model of our case study by using the tree editor of EMF. The right-hand side shows how our Pareto-based approach has been implemented by means of such a tree editor. Furthermore, several Java classes have been developed to obtain a tree representation of the Pareto efficiency to be tailored to the i^* model that conforms to our metamodel for specifying Web requirements (previously described in Sect. 3): a *ParetoModel* represents all the solutions of our approach from a specific input requirements model (contained in the *ParetoModelRef*). Both, the *ParetoLink* and *ParetoLinkEnd* reference to each of the elements of this input model in order to be useful for computing the Pareto efficiency.

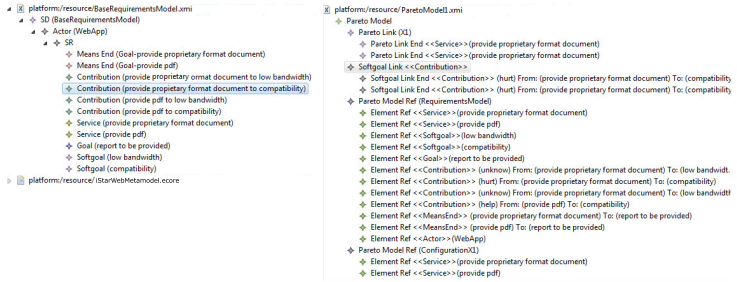


Fig. 5: The i^* base RIA requirements model (left) and the Pareto model (right).

7 Conclusions and Future Work

In this paper, we presented an extension to support RIAs to an existing i^*-based goal-oriented requirements analysis method for Web 1.0 applications. Our approach goes out from the user's goals and expectations, and allows the designer to evaluate and decide on good client/server distributions for the functionality. To do so, we devised a set of steps based on the Pareto optimal approach that is particularly suited to balance and maximize the conflicting softgoals. Furthermore, it facilitates the evaluation of the obtained (Pareto) optimal solutions and selection of the final solution taking into account the priorities of softgoals. We exemplified our approach using an excerpt of a real-world case study we performed for a bio-informatics company. As future work, we plan to study and incorporate the data distribution between client and server in our Pareto algorithm. We would also like to integrate our approach in a RIA engineering method, and study the possibility to automatically generate design artifacts.

References

1. Aguilar, J.A., Garrigós, I., Mazón, J.N.: Impact Analysis of Goal-Oriented Requirements in Web Engineering. In: ICCSA. pp. 421–436. Springer, Santander, Spain (2011)

2. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering. Journal of Universal Computer Science (J. UCS) 16(17), 2475–2494 (2010)
3. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: Web Engineering Approaches for Requirement Analysis - A Systematic Literature Review. In: WEBIST. pp. 187–190 (2010)
4. Bozzon, A., Comai, S., Fraternali, P.: Current Research on the Design of Web 2.0 Applications Based on Model-Driven Approaches. ICWE pp. 25–31 (2008)
5. Collette, Y., Siarry, P.: Multiobjective optimization: principles and case studies. Springer (2003)
6. Cormode, G., Krishnamurthy, B.: Key differences between Web 1.0 and Web 2.0. First Monday 13(6), 2 (2008)
7. Dolog, P., Stage, J.: Designing interaction spaces for Rich Internet Applications with UML. In: Baresi, L., Fraternali, P., Houben, G.J. (eds.) Web Engineering. LNCS, vol. 4607, pp. 358–363. Springer (2007)
8. Horkoff, J., Yu, E.: Evaluating Goal Achievement in Enterprise Modeling–An Interactive Procedure and Experiences. The Practice of Enterprise Modeling pp. 145–160 (2009)
9. Koch, N., Pigerl, M., Zhang, G., Morozova, T.: Patterns for the model-based development of rias. In: ICWE. pp. 283–291. Springer, Berlin (2009)
10. Lowe, D.: Web system requirements: an overview. Requirements Engineering 8, 102–113 (2003)
11. Luna, E.R., Garrigós, I., Mazón, J.N., Trujillo, J., Rossi, G.: An i*-based Approach for Modeling and Testing Web Requirements. Journal of Web Engineering 9(4), 302–326 (2010)
12. Machado, L., Filho, O., Ribeiro, J.a.: UWE-R: an extension to a web engineering methodology for rich internet applications. WSEAS Trans. Info. Sci. and App. 6, 601–610 (April 2009)
13. Meliá, S., Gómez, J., Pérez, S., Díaz, O.: A model-driven development for GWT-based Rich Internet Applications with OOH4RIA. In: ICWE. pp. 13–23. IEEE (2008)
14. Preciado, J.C., Linaje, M., Sanchez, F., Comai, S.: Necessity of methodologies to model rich internet applications. In: WSE. pp. 7–13. IEEE, Washington, DC, USA (2005)
15. Preciado, J., Linaje, M., Comai, S., Sanchez-Figueroa, F.: Designing rich internet applications with web engineering methodologies. In: WSE. pp. 23–30. IEEE (Oct 2007)
16. Preciado, J., Linaje, M., Sanchez-Figueroa, F.: Adapting Web 1.0 User Interfaces to Web 2.0 Multidevice User Interfaces using RUX-Method. Journal of Universal Computer Science (J. UCS) 14(13), 2239–2254 (2008)
17. Sommerville, I.: Software Engineering. Addison-Wesley, 6th edn. (2001)
18. Urbieta, M., Rossi, G., Ginzburg, J., Schwabe, D.: Designing the interface of rich internet applications. In: LA-Web. pp. 144–153. IEEE (2007)
19. Valverde, F., Pastor, O.: Facing the Technological Challenges of Web 2.0: A RIA Model-Driven Engineering Approach. WISE pp. 131–144 (2009)
20. Wright, J.M., Dietrich, J.B.: Requirements for rich internet application design methodologies. In: WISE. pp. 106–119. Springer, Berlin (2008)
21. Wright, J.: A Modelling Language for Interactive Web Applications. In: ASE. pp. 689–692. IEEE (2010)
22. Yu, E.: Modelling Strategic Relationships for Process Reenginering. Ph.D. thesis, University of Toronto, Canada (1995)

Bibliografía

1. J. A. Aguilar, I. Garrigós, S. Casteleyn, and J.-N. Mazón. Optimizing Non-Functioanl Requirements in Web Applications. In *Proceedings of the 8th International Workshop on Web Information Systems Modeling (WISM) (In press) held in conjunction with ER*, 2011.
2. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Modelos de weaving para trazabilidad de requisitos Web en A-OOH. In *DSDM: Actas del VII Taller sobre Desarrollo de Software Dirigido por Modelos, JISBD, Congreso Español de Informática (CEDI)*, pages 146–155, Valencia, España, 2010. SISTEDES.
3. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Impact Analysis of Goal-Oriented Requirements in Web Engineering. In B. Murgante, O. Gervasi, A. Iglesias, D. Taniar, and B. Apduhan, editors, *Proceedings of the 12th International Conference on Computational Science and Its Applications (ICCSA)*, volume 6786 of *Lecture Notes in Computer Science*, pages 421–436. Springer Berlin / Heidelberg, 2011.
4. J. A. Aguilar, I. Garrigós, J.-N. Mazón, and J. Trujillo. An MDA Approach for Goal-oriented Requirement Analysis in Web Engineering. *Journal of Universal Computer Science (J. UCS)*, 16(17):2475–2494, 2010.
5. J. A. Aguilar, I. Garrigós, J. N. Mazón, and J. Trujillo. Web Engineering approaches for requirement analysis- A Systematic Literature Review. In *Proceedings of the 6th Web Information Systems and Technologies (WEBIST)*, volume 2, pages 187–190, Valencia, Spain, 2010. SciTePress Digital Library.
6. S. Anwer and N. Ikram. Goal-Oriented Requirement Engineering: A Critical Study of Techniques. In *Proceeding of the 13th Asia Pacific Software Engineering Conference (APSEC)*, pages 121 –130, December 2006.
7. R. Arnold and S. Bohner. Impact analysis-Towards a framework for comparison. In *Proceeding of the Software Maintenance (CSM)*, pages 292 –301, sep 1993.
8. ATLAS Transformation Language. `http://www.eclipse.org/m2m/atl/`.
9. M. Barbero, M. Del Fabro, and J. Bézivin. Traceability and provenance issues in global model management. In *Proceeding of the 3rd ECMDA-Traceability Workshop*, 2007.
10. J. Bézivin. On the unification power of models. *Software and Systems Modeling*, 4(2):171–188, 2005.
11. J. Bézivin, F. Büttner, M. Gogolla, F. Jouault, I. Kurtev, and A. Lindow. Model transformations? transformation models! *Model Driven Engineering Languages and Systems*, pages 440–453, 2006.
12. D. Bolchini and J. Mylopoulos. From Task-Oriented to Goal-Oriented Web Requirements Analysis. In *Proceedings of the 4th International Conference on Web Information Systems Engineering (WISE)*, page 166, Washington, DC, USA, 2003. IEEE Computer Society.
13. A. Bozzon, S. Comai, and P. Fraternali. Current Research on the Design of Web 2.0 Applications Based on Model-Driven Approaches. In *Proceedings of the 8th International Conference on Web Engineering (ICWE)*, pages 25–31, 2008.
14. A. Bozzon, S. Comai, P. Fraternali, and G. T. Carughi. Conceptual modeling and code generation for Rich Internet Applications. In *Proceedings of the 6th International Conference on Web Engineering (ICWE)*, page 353, New York, New York, USA, 2006. ACM Press.
15. S. Casteleyn, W. V. Woensel, and G.-J. Houben. A semantics-based aspect-oriented approach to adaptation in Web engineering. In *Proceedings of the 18th Conference on Hypertext and Hypermedia (HT)*, pages 189–198, 2007.
16. S. Ceri, P. Fraternali, and A. Bongio. Web Modeling Language (WebML): a modeling language for designing Web sites. *Journal of Computer and Telecommunications Networking*, 33(1-6):137–157, 2000.
17. Y. Collette and P. Siarry. *Multiobjective optimization: principles and case studies*. Springer Verlag, 2003.
18. O. M. F. De Troyer and C. J. Leune. WSDM: a user centered design method for Web sites. *Comput. Netw. ISDN Syst.*, 30(1-7):85–94, 1998.
19. M. Del Fabro, J. Bézivin, and P. Valduriez. Weaving Models with the Eclipse AMW plugin. In *Proceedings of the Eclipse Modeling Symposium, Eclipse Summit Europe*, volume 2006, 2006.
20. P. Dolog and J. Stage. Designing interaction spaces for Rich Internet Applications with UML. *Journal of Web Engineering*, pages 358–363, 2007.
21. Eclipse. `http://www.eclipse.org/`, 2010.
22. Eclipse Modeling Framework. `http://www.eclipse.org/modeling/emf/`, 2010.
23. M. Escalona, M. Mejías, and J. Torres. Developing systems with NDT & NDT-Tool. In *Proceedings of the 13th International Conference on Information Systems Development: methods and tools, theory and practice*, pages 149–59, 2004.

24. M. J. Escalona and G. Aragón. NDT. A Model-Driven Approach for Web Requirements. *IEEE Transactions on Software Engineering*, 34(3):377–390, 2008.
25. M. J. Escalona and N. Koch. Requirements Engineering for Web Applications - A Comparative Study. *J. Web Eng.*, 2(3):193–212, 2004.
26. I. Garrigós. *A-OOH: Extending Web Application Design with Dynamic Personalization.* PhD thesis, University of Alicante, Spain, 2008.
27. I. Garrigós, J.-N. Mazón, and J. Trujillo. A Requirement Analysis Approach for Using i* in Web Engineering. In *Proceedings of the 9th International Conference on Web Engineering (ICWE)*, pages 151–165, 2009.
28. P. Godfrey, R. Shipley, and J. Gryz. Algorithms and analyses for maximal vector computation. *The VLDB Journal*, 16:5–28, January 2007.
29. R. Goeritzer. Using impact analysis in industry. In *Proceeding of the 33rd International Conference on Software Engineering (ICSE)*, pages 1155–1157. ACM, 2011.
30. J. Gómez, C. Cachero, and O. Pastor. Extending a Conceptual Modelling Approach to Web Application Design. In *Proceedings of the 12th International Conference on Advanced Information Systems Engineering (CAiSE)*, pages 79–93, London, UK, 2000. Springer-Verlag.
31. O. Gotel and A. Finkelstein. An analysis of the requirements traceability problem. In *Proceedings of the 1st International Conference on Requirements Engineering (RE)*, pages 94–101, 1994.
32. Graphical Modeling Framework. `http://www.eclipse.org/modeling/gmp//`, 2010.
33. i* wiki. `http://istar.rwth-aachen.de`.
34. S. E. Institute. *CMMI for Development®: Guidelines for Process Integration and Product Improvement.* Addison-Wesley Professional, 3rd edition, 2011.
35. Java Language. `http://www.java.com`, 2011.
36. F. Jouault and I. Kurtev. On the Architectural Alignment of ATL and QVT. In *Proceedings of the ACM Symposium on Applied Computing*, pages 1188–1195. ACM, 2006.
37. N. Koch. The Expressive Power of UML-based Web Engineering. In *Proceedings of the 1st International Workshop on Web-oriented Software Technology (IWWOST)*, pages 40–41, 2002.
38. N. Koch, A. Knapp, G. Zhang, and H. Baumeister. UML-Based Web Engineering. In *Web Engineering: Modelling and Implementing Web Applications*, Human-Computer Interaction Series, pages 157–191. Springer London, 2008.
39. N. Koch, S. Meliá, N. Moreno, V. Pelechano, F. Sánchez, and J. Vara. Model-driven Web Engineering. *The European Journal for the Informatics Professional*, pages 40–45, 2008.
40. N. Koch, M. Pigerl, G. Zhang, and T. Morozova. Patterns for the Model-Based Development of RIAs. In *Proceedings of the 9th International Conference on Web Engineering (ICWE)*, ICWE, pages 283–291, Berlin, Heidelberg, 2009. Springer-Verlag.
41. N. Koch, G. Zhang, and M. J. E. Cuaresma. Model transformations from requirements to Web system design. In *Proceedings of the International Conference of Web Engineering (ICWE)*, pages 281–288, 2006.
42. H. T. Kung, F. Luccio, and F. P. Preparata. On Finding the Maxima of a Set of Vectors. *J. ACM*, 22:469–476, October 1975.
43. M. Linaje, J. C. Preciado, and F. Sanchez-Figueroa. Engineering Rich Internet Application User Interfaces over Legacy Web Models. *Journal of IEEE Internet Computing*, 11:53–59, 2007.
44. M. Lindvall and K. Sandahl. How well do experienced software developers predict software change? *Journal. Systems and Software*, 43:19–27, October 1998.
45. L. Machado, O. Filho, and J. a. Ribeiro. UWE-R: an extension to a Web engineering methodology for Rich Internet Applications. *Journal of Trans. Info. Sci. and App.*, 6:601–610, April 2009.
46. S. Meliá, J. Gómez, S. Pérez, and O. Díaz. A model-driven development for GWT-based Rich Internet Applications with OOH4RIA. In *Proceeding of the 8th International Conference on Web Engineering (ICWE)*, pages 13–23. IEEE, 2008.
47. Model Driven Architecture. `http://www.omg.org/mda/`.
48. N. Moreno, J. Romero, and A. Vallecillo. An overview of Model-Driven Web Engineering and the MDA. *Web Engineering: Modelling and Implementing Web Applications*, pages 353–382, 2008.
49. B. Nuseibeh and S. M. Easterbrook. Requirements engineering: a roadmap. In *Proceedings of the Internationa Conference on Software Engineering (ICSE)*, pages 35–46, 2000.
50. Object Constraint Language, Version 2.0. `http://www.omg.org/spec/OCL/2.0/`.
51. Object Management Group. `http://www.omg.org`.
52. O. Pastor, S. M. Abrahão, and J. Fons. An Object-Oriented Approach to Automate Web Applications Development. In *Proceedings of the 2nd International Conference on Electronic Commerce and Web Technologies (EC-Web)*, pages 16–28, London, UK, 2001. Springer-Verlag.

53. J. Preciado, M. Linaje, S. Comai, and F. Sanchez-Figueroa. Designing Rich Internet Applications with Web Engineering Methodologies. In *Proceedings of the 9th IEEE International Workshop on Web Site Evolution (WSE)*, pages 23–30. IEEE, Oct. 2007.
54. J. Preciado, M. Linaje, and F. Sanchez-Figueroa. Adapting Web 1.0 User Interfaces to Web 2.0 Multidevice User Interfaces using RUX-Method. *Journal of Universal Computer Science (J.UCS)*, 14(13):2239–2254, jul 2008.
55. J. C. Preciado, M. Linaje, F. Sanchez, and S. Comai. Necessity of methodologies to model Rich Internet Applications. In *Proceedings of the 7th IEEE International Symposium on Web Site Evolution*, pages 7–13, Washington, DC, USA, 2005. IEEE Computer Society.
56. B. Qian, L. Wang, D. xian Huang, W. liang Wang, and X. Wang. An effective hybrid DE-based algorithm for multi-objective flow shop scheduling with limited buffers. *Computers and Operations Research*, 36(1):209 – 233, 2009.
57. R. Quintero, V. Pelechano, O. Pastor, and J. Fons. Aplicación de MDA al Desarrollo de Aplicaciones Web en OOWS. *Jornadas de Ingeniería de Software y Base de Datos (JISBD), VIII*, pages 84–668, 2003.
58. QVT Language. `http://www.omg.org/cgi-bin/doc?ptc/2005-11-01`.
59. D. Schmidt. Guest editor's introduction: Model-driven engineering. *Computer*, 39(2):25–31, 2006.
60. D. Schwabe and G. Rossi. The Object-Oriented Hypermedia Design Model. *Communications of the ACM*, 38(8):45–46, 1995.
61. Software Engineering Institute. `http://www.sei.cmu.edu/`, 2010.
62. I. Sommerville. *Software Engineering*. Addison-Wesley, 6th edition, 2001.
63. F. Szidarovszky, M. Gershon, and L. Duckstein. *Techniques for multiobjective decision making in systems management*. Elsevier, 1986.
64. Unified Modeling Language. `http://www.uml.org`.
65. M. Urbieta, G. Rossi, J. Ginzburg, and D. Schwabe. Designing the Interface of Rich Internet Applications. In V. A. F. Almeida and R. A. Baeza-Yates, editors, *5th Latin American Web Congress (LA-Web)*, pages 144–153. IEEE Computer Society, 2007.
66. P. Valderas and V. Pelechano. A Survey of Requirements Specification in Model-Driven Development of Web Applications. *ACM Trans. Web*, 5:10:1–10:51, May 2011.
67. F. Valverde. *OOWS 2.0: Un Método de Ingeniería Web Dirigido por Modelos para la Producción de Aplicaciones Web 2.0*. PhD thesis, Universidad Politécnica de Valencia, 2010.
68. A. Van Lamsweerde. Goal-Oriented Requirements Engineering: A Guided Tour. In *Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE)*, RE '01, pages 249–, Washington, DC, USA, 2001. IEEE Computer Society.
69. J. Wright. A Modelling Language for Interactive Web Applications. In *Proceedings of the 24th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 689–692. IEEE, 2010.
70. J. M. Wright and J. B. Dietrich. Requirements for rich internet application design methodologies. In *Proceedings of the 9th international conference on Web Information Systems Engineering*, WISE '08, pages 106–119, Berlin, Heidelberg, 2008. Springer-Verlag.
71. Xpand Language. `http://wiki.eclipse.org/Xpand`, 2011.
72. S. Yoo and M. Harman. Using hybrid algorithm for Pareto efficient multi-objective test suite minimisation. *Journal of Systems and Software*, 83(4):689–701, 2010.
73. E. Yu. *Modelling Strategic Relationships for Process Reenginering*. PhD thesis, University of Toronto, Canada, 1995.
74. E. Yu. Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering. In *Journal of Requirements Engineering*, pages 226–235, 1997.

Reunido el Tribunal que suscribe en el día de la fecha acordó otorgar, por a la Tesis Doctoral de Don/Dña. Jose Norberto Mazón López la calificación de .

Alicante de de

El Secretario,

El Presidente,

UNIVERSIDAD DE ALICANTE
Comisión de Doctorado

La presente Tesis de D. _____ ha sido registrada con el nº _____ del registro de entrada correspondiente.

Alicante ___ de _____ de _____

El Encargado del Registro,

Reunido el Tribunal que suscribe en el día de la fecha acordó otorgar, por a la Tesis Doctoral de Don/Dña. la calificación de .

Alicante de de

El Secretario,

El Presidente,

UNIVERSIDAD DE ALICANTE
CEDIP

La presente Tesis de D. _____ ha sido registrada con el nº _____ del registro de entrada correspondiente.

Alicante ___ de _____ de _____

El Encargado del Registro,