

José Alfonso Aguilar Calderón

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

– PhD. Thesis –

Advisor: Irene Garrigos, Jose-Norberto Mazón López

Depto. Lenguajes y Sistemas Informáticos
Universidad de Alicante

Contents

1	Síntesis	1
1.1	Tesis Doctoral como Compendio de Artículos	1
1.1.1	Publicaciones Pertenecientes a la Tesis Doctoral	1
1.1.2	Artículos en Proceso de Revisión Pertenecientes a la Tesis Doctoral	3
1.1.3	Otras Publicaciones	4
1.2	Hipótesis Inicial y Objetivos de Investigación	5
1.3	Resumen del Contenido de la Tesis Doctoral	7
1.3.1	Una Aproximación Orientada a Objetivos para el Análisis de Requisitos en Ingeniería Web	7
1.3.2	Gestión de Requisitos en A-OOH	17
1.4	Hacia la Gestión de Requisitos en las <i>Rich Internet Applications</i>	26
1.5	Implementación	33
1.5.1	Editor Gráfico para la Especificación de Requisitos Web (<i>WebREd</i>)	34
1.5.2	Transformaciones Modelo a Modelo	34
1.5.3	Trazabilidad de Requisitos	37
1.5.4	Optimización de Pareto	37
1.5.5	Caso de estudio	37
1.6	Conclusiones	43
	References	44
	References	47

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 tres partes. La primera parte (Parte ??) consiste en una síntesis de la tesis. La Parte ?? presenta el conjunto de artículos publicados que forman el núcleo de la tesis. Finalmente, la Parte ?? 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) 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, 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 ??

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. Por último, 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 ??

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 tomando 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. Finalmente, la derivación se realiza por medio de un conjunto de transformaciones descritas de manera formal utilizando el lenguaje QVT³.

Capítulo ??

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

¹ <http://www.thomsonreuters.com/>

² <http://www.core.edu.au/>

³ *Query/View/Transformation*

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.

Capítulo ??

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, In press.

La idea de 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, ha sido objeto de investigación en el contexto del desarrollo dirigido por modelos (*Model-Driven Development*). Para ello, es necesario considerar a los requisitos funcionales así como a los no-funcionales debido a que los dos poseen la habilidad de satisfacer las necesidades de los *stakeholders* y por lo tanto, ambos tipos de requisitos afectan la calidad de un producto de *software*, según lo establecido en la ISO/IEC 9126-1:2001, en la sección de *Product Quality*, específicamente en la primera parte *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 los requisitos no-funcionales de la aplicación Web. Una configuración de requisitos óptima esta formada por el sub conjunto de requisitos funcionales que se deberán implementar en los modelos conceptuales de la aplicación. La idea 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.

1.1.2 Artículos en Proceso de Revisión Pertenecientes a la Tesis Doctoral

En este apartado, se presentan dos trabajos que forman parte de la tesis doctoral. Los trabajos están actualmente bajo proceso de revisión.

Apéndice ??

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

En este trabajo 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 ??, 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 ?? 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 a la obtención, 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 [24].

Apéndice ??

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 tecnologías de implementación de las aplicaciones Web evolucionan constantemente. Parte de la evolución son las aplicaciones RIAs (*Rich Internet Applications*), las cuales ofrecen, entre otras cosas, una mejor interactividad con el usuario, similar a la ofrecida por las aplicaciones *software* de escritorio. En este trabajo, se presenta la adaptación de la propuesta descrita en el Capítulo ?? para auxiliar al diseñador Web en la distribución de la funcionalidad de la aplicación 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 elegir la configuración que maximice a los requisitos no-funcionales. De esta forma, es posible optimizar los requisitos no-funcionales mediante la distribución de los requisitos funcionales entre el cliente y el servidor.

Apéndice ??

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).

Este apéndice es una extensión del Capítulo ?? acerca de la importancia de considerar el análisis de impacto 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 del perfil UML para adaptar el marco de modelado i^* en el dominio Web como un metamodelo, (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 del algoritmo para el análisis de impacto en el modelo de requisitos orientado a objetivos.

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 y/o internacionales. Cabe destacar que los trabajos no han sido incluidos 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, 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, uno de los factores de éxito más importantes en el desarrollo de *software* es la elicitación, gestión y análisis de requisitos. La gestión de requisitos es el proceso de comprender y controlar los cambios en los requisitos de la aplicación [58].

Sin embargo, en el desarrollo de *software* en ingeniería Web llevar a cabo una correcta gestión de los 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 la etapa de análisis y especificación de requisitos a razón de 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 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 (funcionales y no-funcionales) que el sistema debe cumplir para satisfacer las necesidades de los usuarios es una tarea que necesita una atención especial.

Actualmente, existen una notable cantidad de aproximaciones metodológicas para el desarrollo de aplicaciones Web (A-OOH [26], UWE [33], NTD [22], OOWS [48], etc.) que toman en cuenta la aplicación de distintas técnicas para llevar a cabo el 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 (casos de uso) [34]. 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 proceso de desarrollo en base a modelos. MDE guía el proceso en base a conceptos y reglas tomados directamente de una área de interés determinada, es decir, del dominio del problema [55]. Para esto, MDE incorpora conceptos de la ingeniería de software e ingeniería de requisitos, tales como los modelos de madurez [57], los marcos de trabajo conceptuales como las metodologías ágiles, trazabilidad de los productos de trabajo ⁴ derivados del proceso de desarrollo [57] y la evolución del software [58].

⁴ Cualquier artefacto producido por un proceso. Estos artefactos pueden incluir archivos, documentos, piezas de los productos, servicios, procesos y especificaciones [59].

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 [2] por medio del modelado del sistema *software* y su generación a partir de los modelos. 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) [44] surge como un estándar del OMG (*Object Management Group*) [47] 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, esto es, el código fuente de la aplicación [44].

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 algunos casos, 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 [35, 45]. Su supuesto básico 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.

En la actualidad, MDA ha sido aplicado para el desarrollo de aplicaciones Web por parte de ciertas metodologías tales como OOWS [53], NDT [23] y UWE [37]. Lamentablemente, a pesar que se ha resaltado la importancia de la etapa de análisis de requisitos en Web [5], algunas metodologías dan poca importancia a la especificación de requisitos (OOWS [48], OOHDM [56], WSDM [15] y HERA [12]) otras, por su parte, consideran la especificación de requisitos a nivel CIM de MDA utilizando técnicas como los casos de uso (UWE [33]). Sin embargo, por lo que al autor concierne, el trabajo presentado en la tesis doctoral es el primero que 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^* [31]) [4]. De esta forma, es posible la obtención, con alto grado de automatización, de la estructura de los modelos conceptuales a nivel PIM directamente de la especificación de requisitos. Una de las principales ventajas de la propuesta es que se asegura que los modelos conceptuales obtenidos a partir del nivel CIM de MDA sean correctos semánticamente. Asimismo, ofrece al diseñador soporte para la gestión de los requisitos, por ejemplo visualización de la trazabilidad, análisis de impacto así como la selección de alternativas de diseño que consideren el balance y maximización de los requisitos no-funcionales.

Cabe destacar que la **hipótesis de partida** de la investigación asociada a la tesis doctoral consiste en que sí es factible 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 la gestión de los requisitos (trazabilidad y análisis de impacto) y

(iii) 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 una metodología para el análisis y especificación de requisitos para aplicaciones Web, que considere:

- Una etapa de análisis de requisitos orientada a objetivos para representar las expectativas reales de los stakeholders de la aplicación Web.
- 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.
- 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 derivado a la evolución de la aplicación Web.
- 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.
- 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.

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. La finalidad de la primera etapa es 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, análisis de impacto y alternativas de diseño en base a la maximización de requisitos no-funcionales.

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*) [26, 4].

A-OOH [25], es la extensión del método OOH (*Object-Oriented Hypermedia*) [28] con soporte de personalización. El proceso de desarrollo de A-OOH esta basado en MDA [44], 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 Fig. 1.1, estos son:

- **Modelo de dominio.** El modelo de dominio de A-OOH se expresa como un diagrama de clases UML (*Unified Modeling Language*) [61]. Este 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. Este 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.** Este 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.** Este modelo es utilizado para la especificación de estrategias de personalización.
- **Modelo de usuario.** Este modelo 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. La estructura de la información necesaria para la personalización también se describe en este modelo.

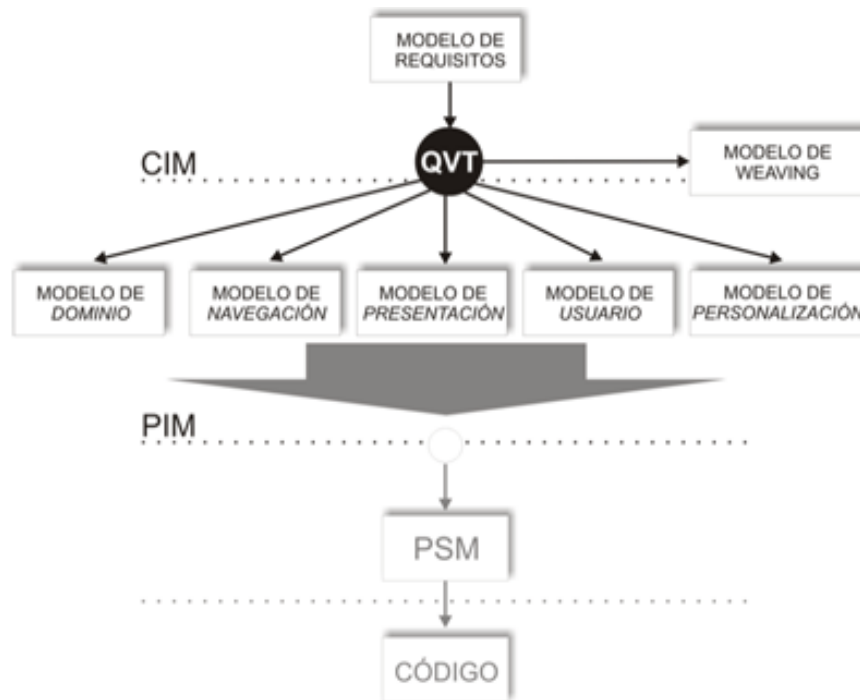


Fig. 1.1. La propuesta de la tesis doctoral integrada como CIM en el método Web A-OOH.

Especificación del Modelo de Requisitos a Nivel CIM








El primer paso de la propuesta presentada en la tesis doctoral es la especificación y el modelado de los requisitos de la aplicación Web. Para una explicación más amplia el lector puede referirse al Capítulo ?? 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^* [67, 68] es uno de los más

utilizados para analizar los objetivos de los *stakeholder's*⁵ (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 ?? y Apéndice ??) 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 como es que 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 importante como para influir 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 los elementos intencionales tales como *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 dentro de los actores. Las relaciones intencionales son enlaces del tipo *Meands-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 es que 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 [31].

Table 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. Lamentablemente no detalla como es que se satisfará el objetivo.
TASK		Representa una manera particular de hacer algo.
SOFTGOAL		Representan criterios de calidad. Son utilizadas cuando los objetivos del <i>stakeholder</i> no son precisos o sus criterios de éxito no están claramente definidos.
RESOURCE		Es una entidad que debe estar disponible para su uso.
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.

⁵ 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 [58].

Por otra parte, debido a 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 ?? y Apéndice ??) el marco de modelado i^* resulta insuficiente para representarlas por sí solo. Por tal motivo, debe adaptarse al dominio Web para poder ser utilizado en la especificación y análisis de requisitos. Esto permitirá modelar a los actores con sus objetivos y las relaciones existentes entre ellos. Para realizar la adaptación de i^* al dominio Web nuestra propuesta utiliza la taxonomía de requisitos Web presentada en [24], 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. 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 el sistema debe proveer a los usuarios. Por ejemplo: “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 son: “consultar productos por categoría”, “consultar el carrito de compras”, etc.
- **Requisitos de interfaz (*Layout*)**. Los requisitos también pueden definir la interfaz visual para los usuarios. Por ejemplo: “presentar un estilo diferente para los adolescentes”.
- **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 [20] (Fig. 1.2). 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*). De esta forma, los requisitos *Navigational*, *Service*, *Personalization* y *Layout* son instancias del elemento *Task* del marco de modelado i^* y el requisito *Content* del elemento *Resource*.

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^* .

11

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

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 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.

UML (*Unified Modeling Language*) [61] es un el lenguaje de modelado estándar utilizado para la derivación del modelo de dominio de A-OOH. El metamodelo 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 clase es representar a las clases dentro de un modelo. En una aplicación orientada a objetos, 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 [25] para representar las rutas de navegación que el usuario puede seguir durante su interacción con la aplicación Web. En la Figura 1.3 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*).

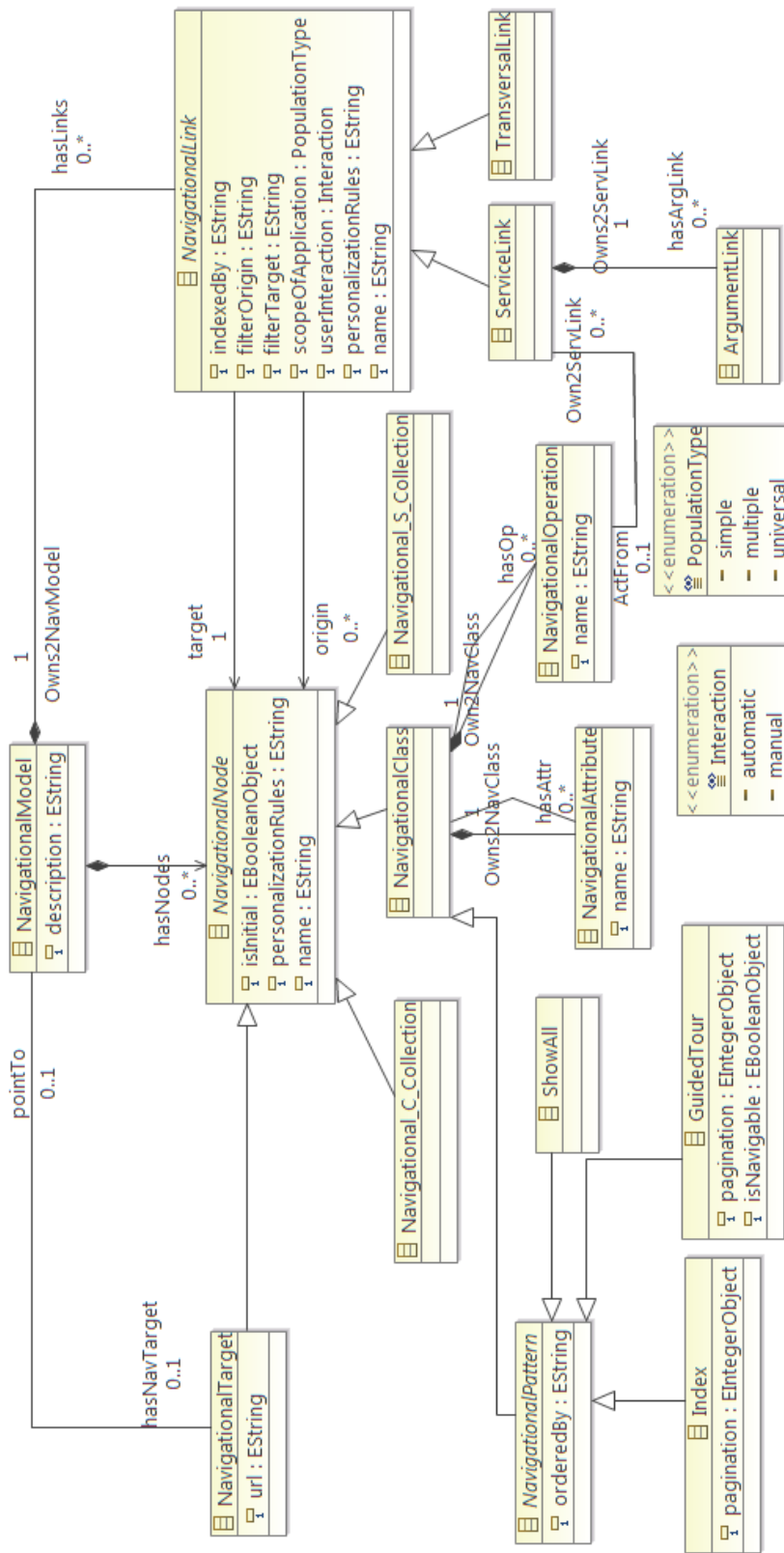


Fig. 1.3. Metamodelo de Navegación de A-OOH.

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 acceder a la información. La colección más común es *C-Collection* (*Classifier 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. Estos tipos de enlaces son representados por el estereotipo *TransversalLink*.
- **S-Link** (*Service Links*), con estos tipos de enlaces la navegación es realizada para activar 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 invoca.

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 [61]. 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 descritas formalmente en QVT [54] (ver Capítulo ?? y Apéndice ??).

- **Content2DomainClass**. El dominio origen de la relación está compuesto por un conjunto de elementos que representan un requisito del tipo *Content*. Cuando se detecta este patrón en el modelo de entrada se fuerza la creación de una clase tipo *Class UML* en el modelo destino (modelo de dominio). Por tanto, por cada requisito de contenido, se obtiene una clase en el modelo de dominio (Figura 1.4).
- **Service2Operation**. La regla detecta un conjunto de elementos en el modelo de entrada (modelo de requisitos) que corresponden con un requisito *Service* asociado a un requisito *Content*. Una vez detectado este patrón de elementos, se crea en el modelo de salida (modelo de dominio) una clase *Operation* en la clase del modelo de dominio correspondiente (Figura 1.5).
- **Navigation2Relationship**. Esta relación permite crear asociaciones entre clases en el 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 clase *Association* entre las clases del modelo de dominio que las representan (Figura 1.6).

Content2DomainClass

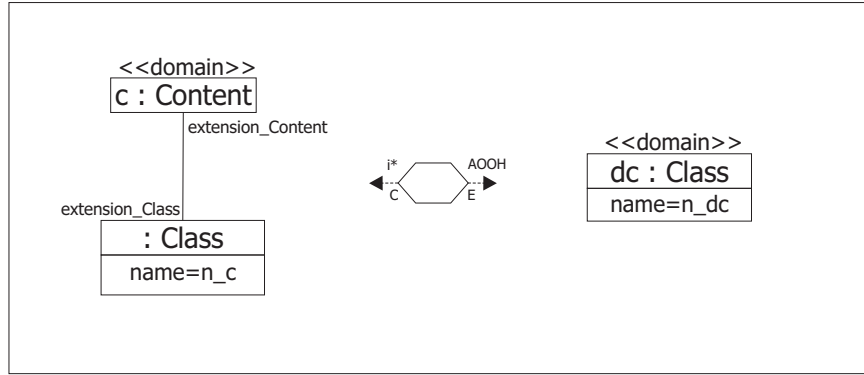


Fig. 1.4. Regla QVT para obtener las clases del modelo de dominio.

Service2Operation

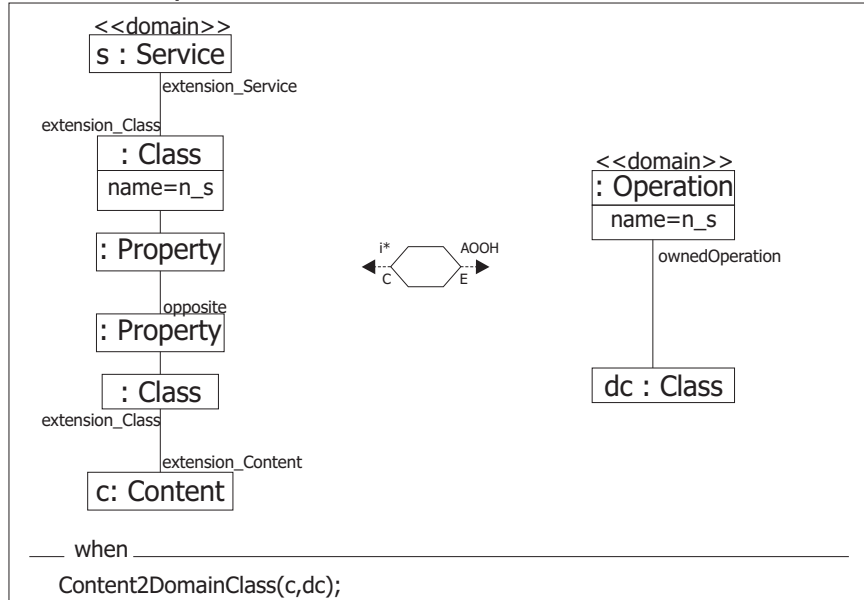


Fig. 1.5. Regla QVT para obtener las clases del modelo de dominio.

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, las reglas QVT para la obtención del modelo de navegación son las siguientes:

- **Navigation2NavClass.** Esta regla de transformación detecta cada requisito navegacional, derivando su correspondiente clase. En concreto, cuando se detecta en el modelo origen un requisito *Navigational* unido a un requisito *Content*, entonces se crea una clase estereotipada como *NavigationalClass* en el modelo 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 1.7).

Navigation2Relationship

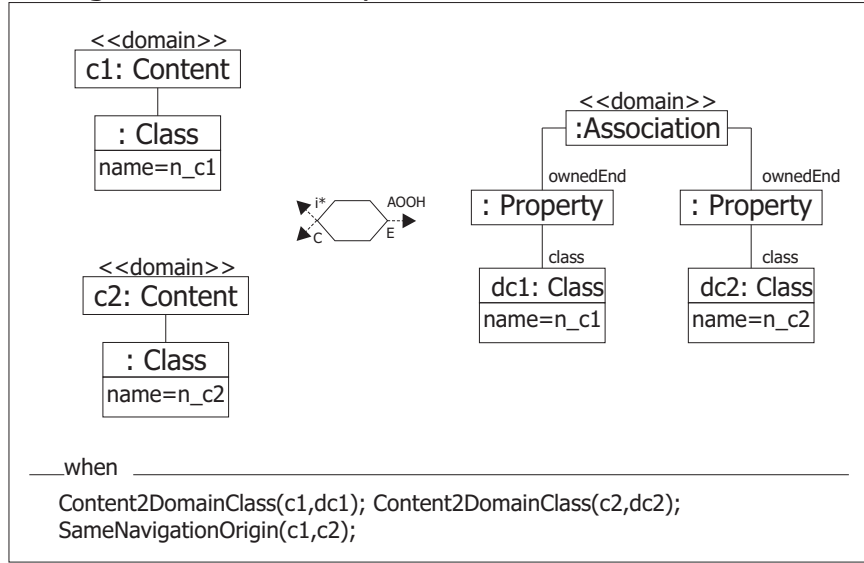


Fig. 1.6. Regla QVT para obtener las relaciones entre las clases del modelo de dominio A-OOH.

Navigation2Relationship

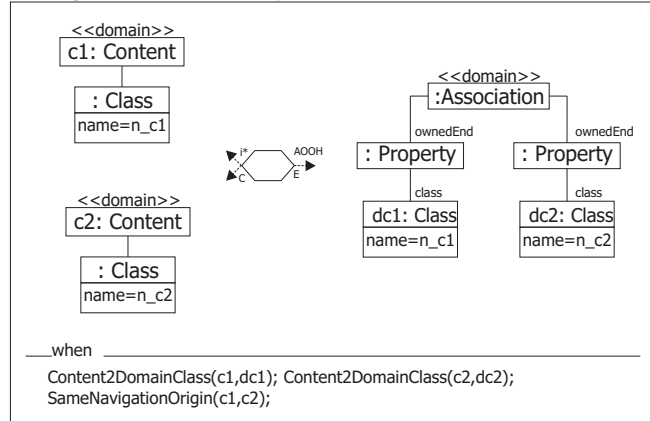


Fig. 1.7. Regla QVT para obtener las clases navegacionales.

- **Personalization2NavClass.** La regla es similar a la anterior pero detectando los requisitos de personalización que tienen un requisito de contenido asociado. Se derivan en el modelo de navegación los mismos elementos que en la regla anterior.
- **Navigation2TransversalLink.** Permite crear asociaciones entre clases navegacionales en el modelo de navegación. Existen dos modelos origen con el fin de detectar requisitos *Content*, 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.
- **Service2ServiceAndSLink.** La relación detecta un conjunto de elementos en el modelo origen que corresponden con un requisito *Service* asociado a un requisito *Content*. Una vez detectado este patrón de elementos, se crea en el modelo 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 *Personaliza tion2NavClass*, con el fin de crear en el modelo destino todas las posibles clases de navegación a partir de cada uno de los requisitos *Content* en el modelo origen.

Finalmente, es importante mencionar que los modelos conceptuales derivados son modelos iniciales que servirán como punto de partida para el diseñador Web, de esta forma es posible proveer un alto grado de automatización en el proceso de desarrollo de aplicaciones Web dirigido por modelos.

1.3.2 Gestión de Requisitos en A-OOH

Las metodologías para el desarrollo de aplicaciones Web aun no responden a las exigencias actuales de este tipo de *software* [5]. En ese aspecto, 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.

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. Actividades propias de esta área, como la especificación de requisitos o la gestión de requisitos del usuario, 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 [67] para la especificación de requisitos en ingeniería Web [9, 26] permite reflejar desde las etapas iniciales del proceso de desarrollo los objetivos, tareas y relaciones de los *stakeholder's*, 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 *stakeholder's* 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 en la tesis se ha extendido para proveer al diseñador con soporte para: (i) trazabilidad de requisitos (CIM-PIM) [2] (Apéndice ??), (ii) evaluar el impacto derivado de un cambio en los modelos conceptuales (análisis de impacto) [3] (Capítulo ??) y (iii) seleccionar alternativas de diseño considerando la maximización y/o balance de los requisitos no-funcionales [1] (Capítulo ??).

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 de navegación. Además, debido al desarrollo gradual de las necesidades de los usuarios de una aplicación Web, estos 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 [29]: (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 [7]. 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. A continuación, se presenta el metamodelo base para *weaving* [16] y una extensión para proveer a dicho metamodelo con elementos útiles para representar la trazabilidad entre modelos [7]. El metamodelo se muestra en la Figura 1.8:

- **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. Está compuesto por 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 el 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 *emphWLinkEnd*.
- **WModelRef**. Representa un identificador único de un modelo.

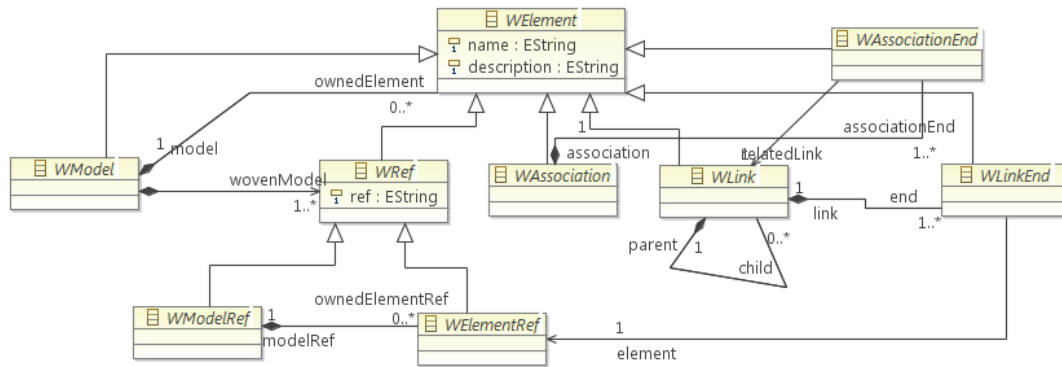


Fig. 1.8. Metamodelo de *weaving*.

Por otra parte, en la Figura 1.9, se ilustra la extensión del metamodelo para *weaving* que permite la representación de la trazabilidad. Esta extensión la forman los siguientes elementos:

- **TraceModel**. Es el elemento que representa al modelo de trazabilidad, esta 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.
- **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*).

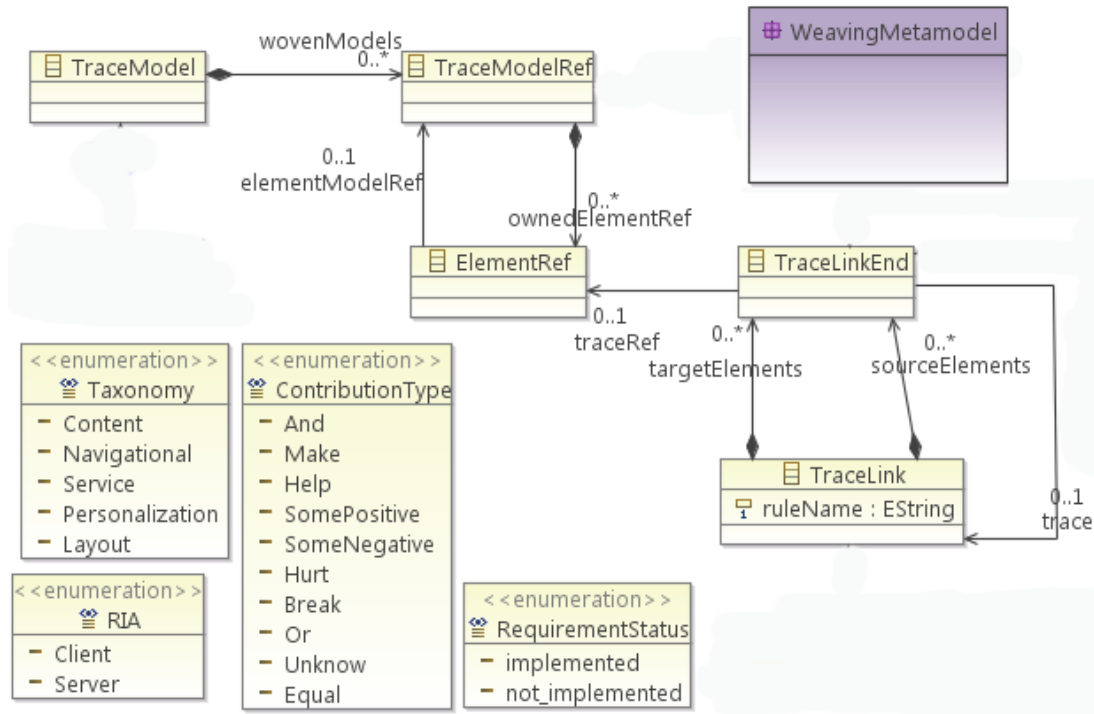


Fig. 1.9. Extensión del metamodelo de *weaving* para trazabilidad.

De esta forma se ha definido formalmente la regla de transformación *CreateModelTrace* para derivar enlaces *weaving* que almacene un conjunto de enlaces dirigidos a los elementos de los modelos conceptuales de A-OOH. La regla ha sido definida formalmente con el lenguaje estándar QVT.

- **CreateModelTrace.** Al ejecutarse por primera vez, la regla crea un modelo de *weaving* con referencias a los modelos origen y destino. De esta forma, cada vez que una regla QVT transforma un elemento del modelo de entrada (modelo de requisitos) en uno o varios elementos de los modelos conceptuales A-OOH (dominio y navegación) crea un nuevo enlace en el modelo de *weaving*.

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 [6], esto es 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. Esto quizá sea suficiente para aplicaciones Web no muy grandes, pero no es suficientes para aplicaciones Web sofisticadas. Asimismo, investigación empírica [41] demuestra que, incluso los desarrolladores más experimentados, 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 depen-

dencias entre los requisitos funcionales además de saber qué requisitos no-funcionales se ven afectados (ver Capítulo ??). De esta forma, a través del soporte para trazabilidad es posible conocer que 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, 26], el algoritmo puede mostrar al diseñador un camino alternativo en el cual se indique qué requisitos funcionales tienen que ser implementados para seguir cumpliendo 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 1.10).

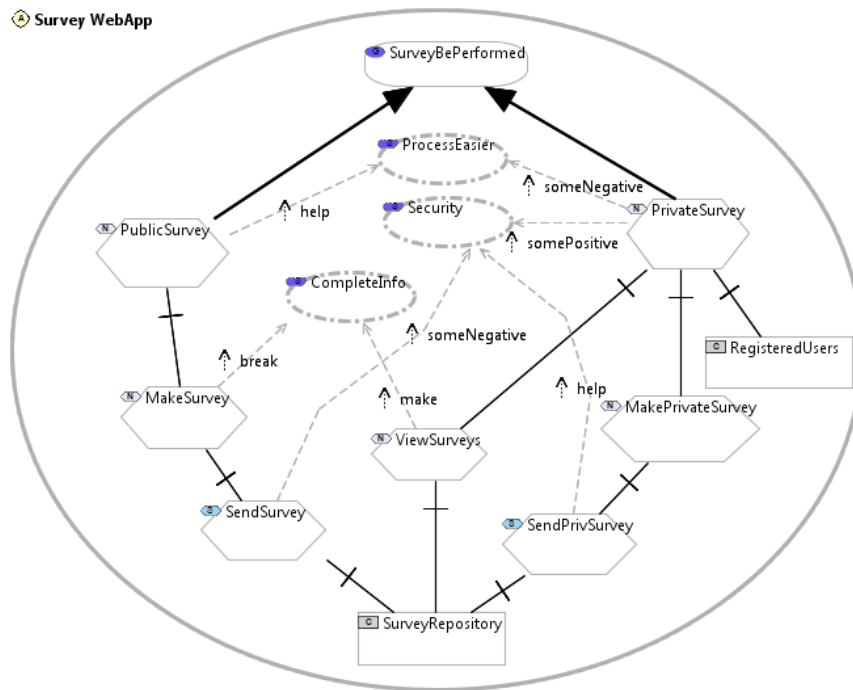


Fig. 1.10. El modelo de requisitos de la aplicación “Survey WebApp”.

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 positivamente 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 facilidad con que deberá realizarse el proceso de revisión utilizando la aplicación Web, por último, el requisito navegacional “View Surveys” afecta de forma positiva a la *Softgoal* “Complete Info”, es decir que si este 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 ya que son utilizadas para decidir qué requisitos funcionales será

necesario implementar. En este ejemplo, solo los requisitos funcionales relacionados con el requisito de navegación “*Public survey*” se encuentran implementados en los modelos conceptuales (Figura 1.11).

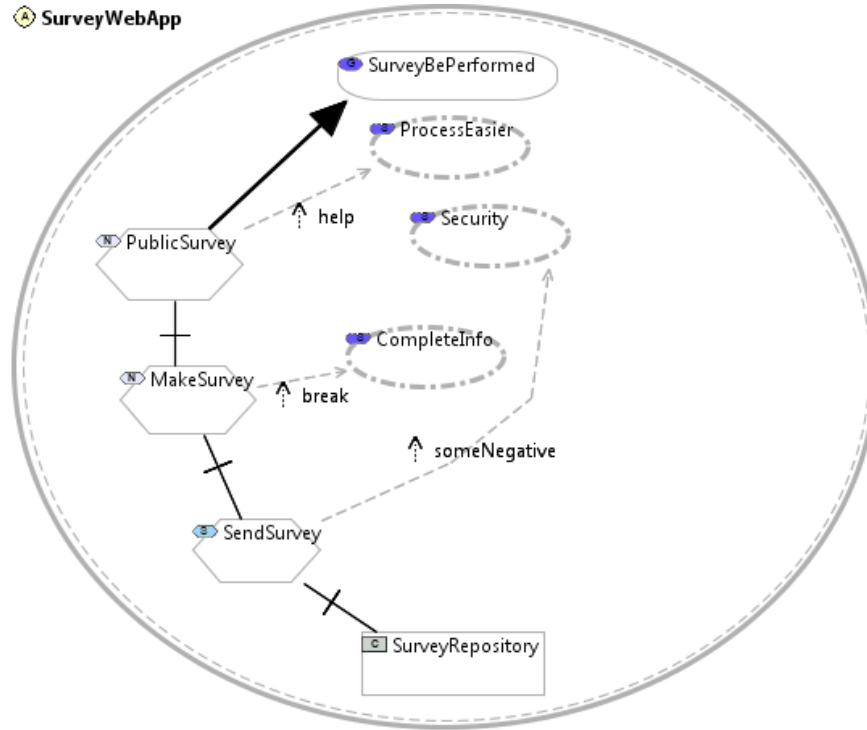


Fig. 1.11. Requisitos de la aplicación “*Survey WebApp*” implementados inicialmente.

Ahora bien, supongamos el siguiente escenario: se 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 1.11, 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 ???. 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*”.

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 a remover (Tabla 1.2) es necesario buscar un requisito funcional no implementado del cual su contribución compense la eliminación

Table 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

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 previamente en el Capítulo ???. 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 1.12 se muestran los requisitos funcionales a implementar.

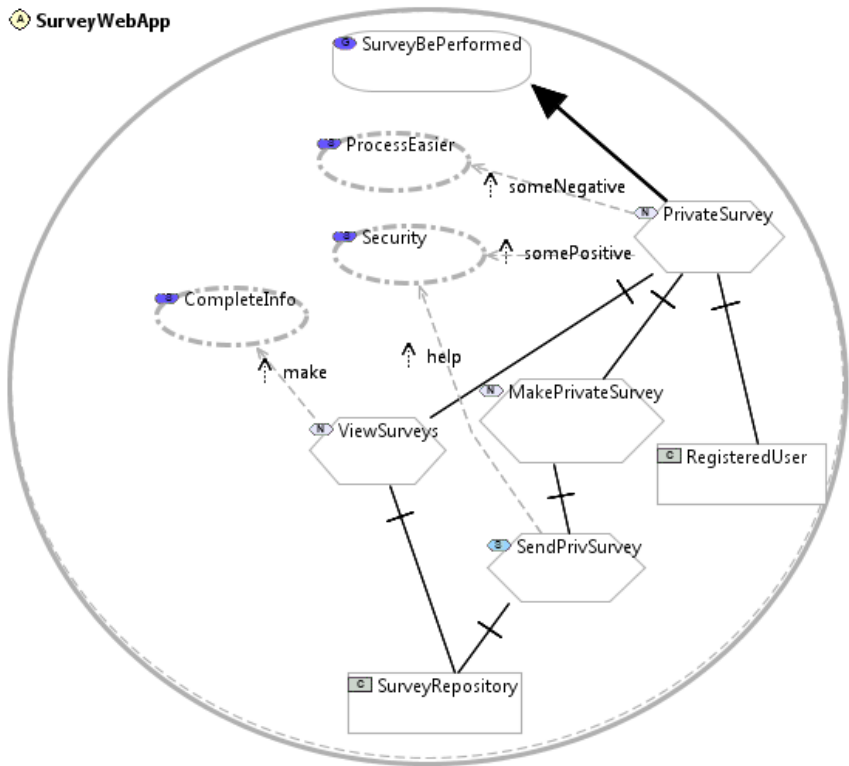


Fig. 1.12. Los nuevos requisitos a implementar en el modelo de dominio.

Optimización de Requisitos No-Funcionales en Aplicaciones Web

Como se ha motivado anteriormente, las aplicaciones Web tienen una audiencia amplia y heterogénea [2, 25], 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 para resolver la pregunta consiste en enriquecer a las metodologías de diseño para que sean capaces de asistir al diseñador en la etapa de requisitos y con ello 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 [60] para proveer al diseñador con un conjunto de alternativas de diseño basadas en la prioridad de los requisitos no-funcionales. 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 [38]. 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* [66], principalmente, en problemas en los que hay 2 o más objetivos, este tipo de problema es conocido como problema multi-objetivo (PMO) [52]. En un problema de optimización, se trata de encontrar una solución que represente el valor óptimo para una función objetivo [27].

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, para una explicación más detallada consultar el Capítulo ?? 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 ubicaciones alternativas y un conjunto de individuos, la ubicación “A” es una optimización sobre la ubicación “B” si solo si “A” puede al menos, optimizar a un individuo mejor que “B”, sin deteriorar a otro individuo* [14, 60].

En la Definición ??, las ubicaciones alternativas corresponden con el estado del requisito funcional, es decir, si el requisito está o no está implementado. Por otra parte, el conjunto de individuos comprende al conjunto de requisitos funcionales utilizados por el algoritmo. Mientras que, optimizar a un individuo mejor que “B”, significa maximizar a los requisitos no-funcionales. Finalmente, en la Definición ?? la frase “*sin deteriorar a otro individuo*” se refiere a no afectar negativamente a los requisitos no-funcionales.

Definición 2 Configuración Óptima de Pareto: *es a 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 esta formada por un conjunto de requisitos funcionales que pueden ser implementados en los modelos conceptuales de la aplicación Web final.*

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 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.

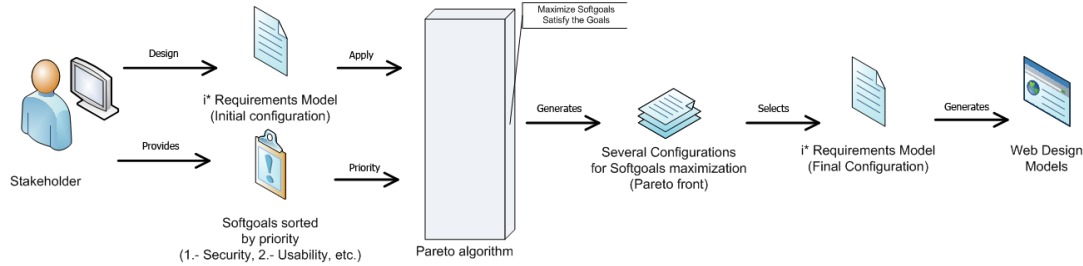


Fig. 1.13. Propuesta para la optimización de requisitos no-funcionales en aplicaciones Web.

La Figura 1.13 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, en último paso consiste en la selección de la configuración que mejor satisfaga la lista de requisitos no-funcionales a priorizar.

A continuación, se presenta un ejemplo conciso para ejemplificar paso a paso la aplicación de la propuesta para la optimización de requisitos no-funcionales 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 1.14, 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*”.

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*”. Es importante destacar que los requisitos no-funcionales son considerados *Softgoals* en la propuesta para la especificación de requisitos, como se definió en la Sección 1.3.1. 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 debido a que se llevará a cabo por medio de la aplicación Web, la *Softgoal* S3 llamada “*Accurate review process*” es

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

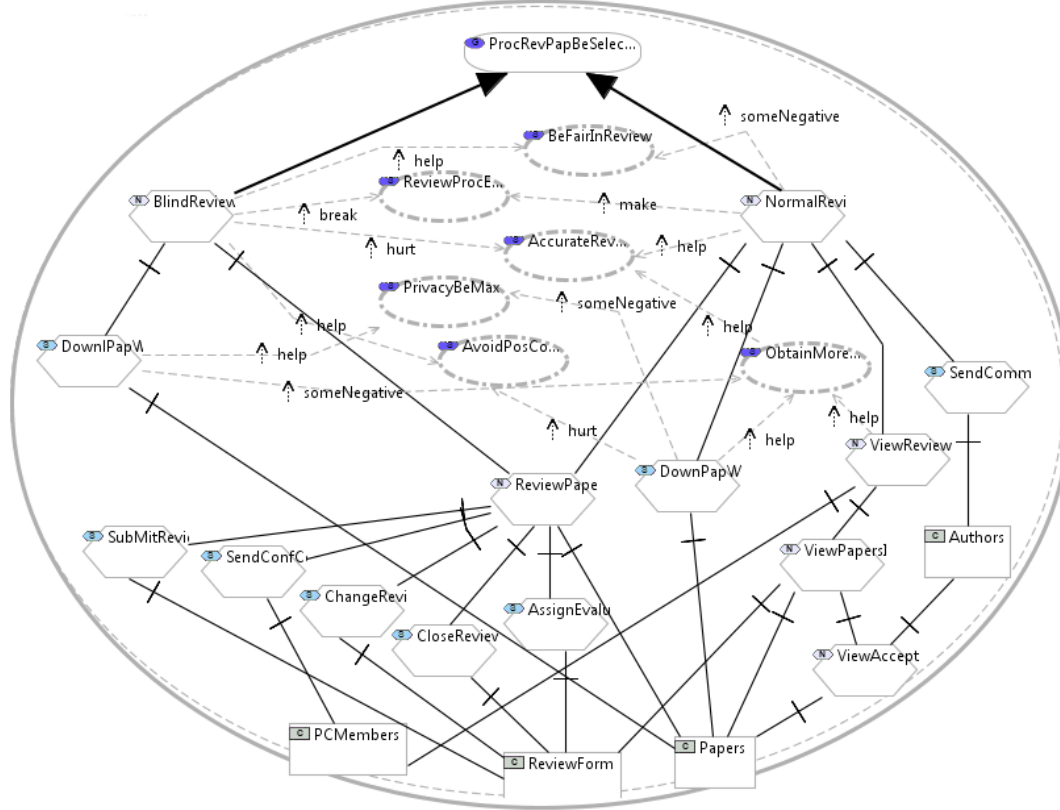


Fig. 1.14. Especificación de requisitos de la aplicación Web para la gestión de conferencias.

Table 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 <i>Softgoal</i>	Break
Positiva suficiente para satisfacer una <i>Softgoal</i>	Make

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 en particular, la *Softgoal* S5 nombrada “*Avoid possible conflicts of interest*”, permite representar la ética del usuario de la aplicación Web debido a que este 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*” la cual 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.

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

Table 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

“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”.

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

Con los pesos establecidos, el siguiente paso consiste en crear una matriz para calcular los valores numéricos 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 (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, 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 un individuo, es decir, maximiza a la *Softgoal* 5.

Finalmente, el último paso consiste en maximizar *Softgoals* y 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 1.6). 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 sta 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 1.6), “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).

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 HTML⁷ 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 [56]. 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 las metodologías de desarrollo (UWE [33], WebML [13], NDT [23], OOWS [48], A-OOH [25]).

Recientemente, las aplicaciones Web han experimentado cambios significativos relacionados con la tecnología de implementación, 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, tales como la inclusión de contenido multimedia bajo demanda, la

⁷ Hyper Text Markup Language

Table 1.5. 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

Table 1.6. 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"

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 de comunicación por medio del audio y video. La Figura 1.15⁸ muestra un diagrama de Venn en donde se representa la combinación de tecnologías que han dado origen a las RIAs y 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).

Sin embargo, la evolución de las aplicaciones Web a RIAs conlleva a 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

⁸ Imagen tomada de <http://www.simonwhatley.co.uk/rich-internet-applications-a-background>

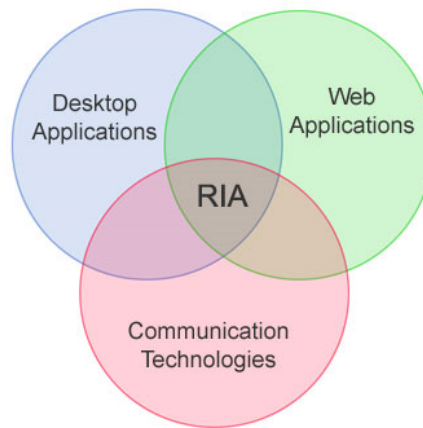


Fig. 1.15. Diagrama de Venn que ejemplifica las características de las RIAs.

(adaptaciones en algunos casos) y han surgido nuevas aproximaciones específicamente para el desarrollo de RIAs [51]. Dentro del grupo de aproximaciones que han sido extendidas para el soporte de RIAs encontramos a OOHDM [62], OOH con la extensión OOH4RIAs [43], UWE con las extensiones UWE-Patterns [36], UWE-R [42] y UWE-RUX [40], WebML [10, 11] y OOWS [63]. Por otra parte, dentro del grupo de aproximaciones creadas exclusivamente para el desarrollo de RIAs encontramos a RUX-Model [50], *Internet Application Modeling Language* [64] y ADRIA [18, 19], para información más detallada sobre las aproximaciones tradicionales y las de nueva creación el lector puede consultar el apéndice ???. 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 [40, 49] y en capacidades de interacción [43] 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 [65].

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 como es que 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 trabajo presentado en la sección 1.3.2 de la tesis (Capítulo ??) para asistir al diseñador al momento de decidir sobre la distribución de los 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 1.16. 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 existente, 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 sobre la adaptación de la propuesta presentada en el Capítulo ??, así como las definiciones y conceptos de esta sección consultar Apéndice ?? y Apéndice ??.

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 genoma 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 vulnerable 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. Por tal motivo, 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 efectos demostrativos, en el caso de estudio nos enfocaremos en los requisitos necesarios para el reporte de enfermedades.

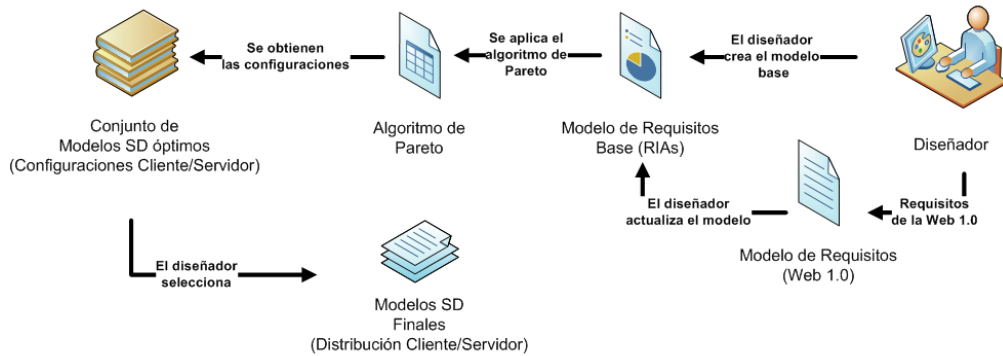


Fig. 1.16. Visión general de la propuesta para el análisis de requisitos RIA.

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 1.17).

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*”. 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 originadas en el requisito “Proporcionar interfaz gráfica” también son etiquetadas debido a que el requisito solo puede ser implementado del lado del cliente.

El segundo paso consiste en aplicar el algoritmo Óptimo de Pareto (Figura 1.16) 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” por que, como lo mencionamos antes, solo es posible implementarlo del lado del cliente. La Tabla 1.7 muestra los requisitos y *softgoals* a utilizar.

En seguida, 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. Posteriormente, 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

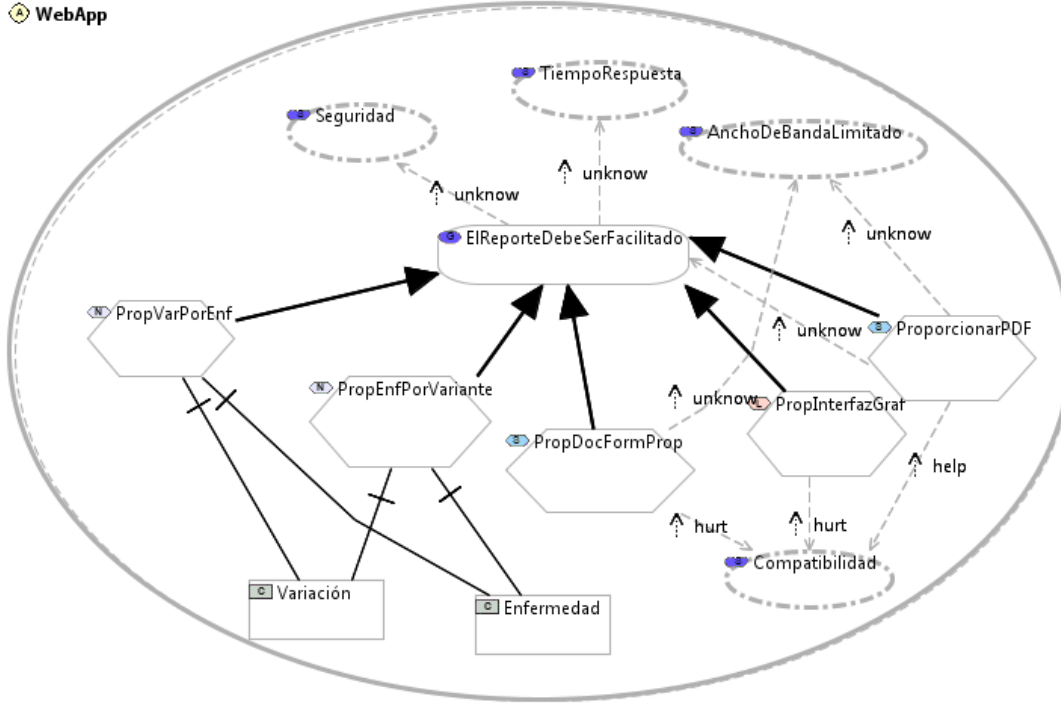


Fig. 1.17. Modelo de requisitos RIA base.

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

<i>Softgoals</i>	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

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 esta o no en la frontera de Pareto (última columna).

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

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

En la Tabla 1.8, las filas de color gris son la Frontera de Pareto. Finalmente, en el tercer paso (ver Figura 1.16), el diseñador puede seleccionar la frontera de Pareto que mejor satisfaga las necesidades del *stakeholder*, es decir, considerando 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 1.18), de acuerdo con Σ (-2), mejorar la seguridad afectará negativamente en algunas de las *softgoals*.

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 *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 *softgoals*, incluso, la *softgoal* “Tiempo de respuesta” será afectada negativamente.

Table 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

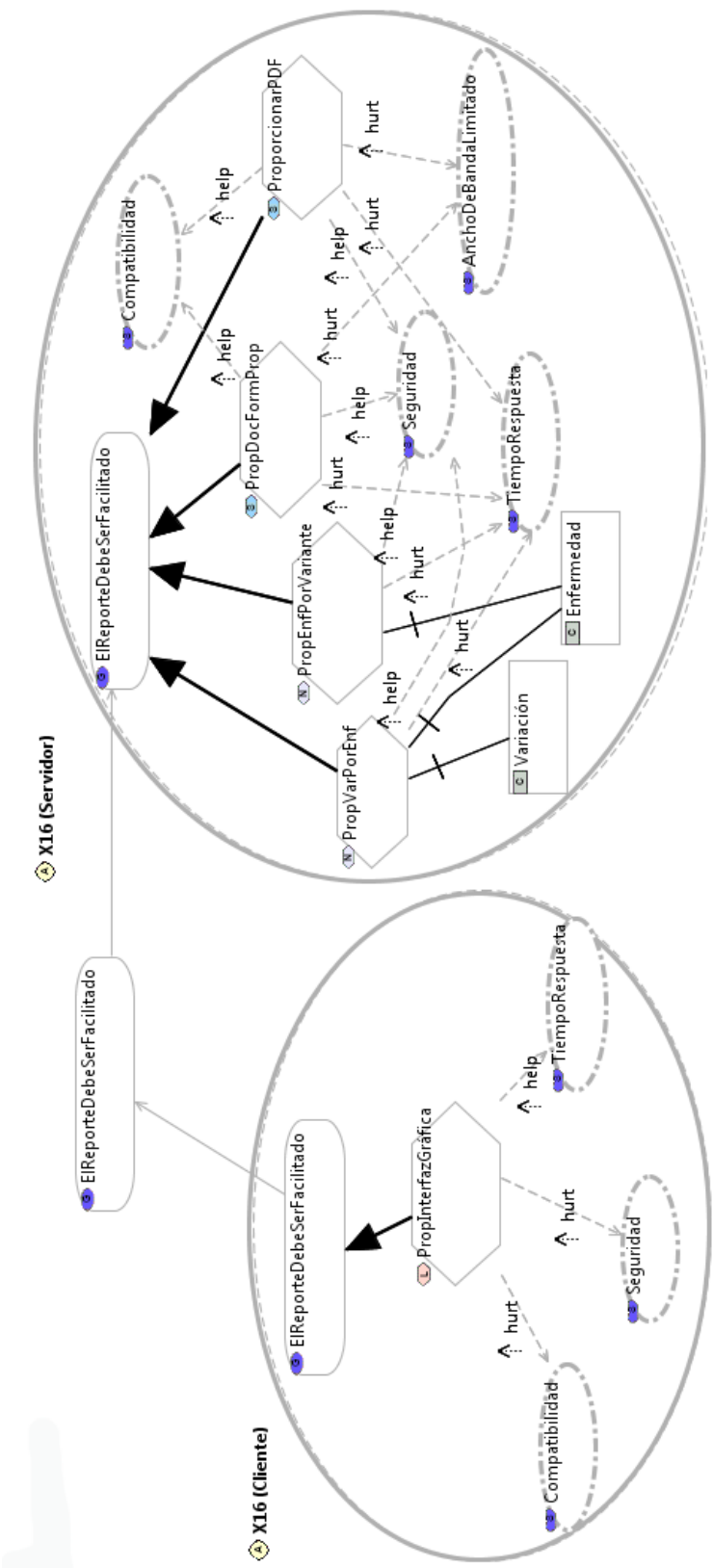


Fig. 1.18. Configuración X16: distribución de los requisitos entre el cliente y el servidor.

1.5 Implementación

Este apartado aborda los objetivos específicos planteados en la investigación asociada a la tesis doctoral. Los objetivos están descritos en la sección 1.2. A continuación se puntualiza cómo es que los objetivos han sido satisfechos a través de cada uno de los apartados establecidos en la sección de implementación de la tesis.

- 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^* . Con esta sección se cumplen los objetivos específicos 1 y 2 (Sección 1.2). Con el editor gráfico, se establecen mecanismos para la comprensión de los objetivos de negocio, y por medio del editor gráfico, 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 ??), 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 satisfacer el objetivo 5 de la tesis doctoral, debido 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 brindar soporte a la generación de código. Para más información consultar los artículos listados en la sección 1.1.3 y el Capítulo ?? de la tesis doctoral.
- Sección 1.5.3: “Trazabilidad de Requisitos”. La sección proporciona el soporte necesario para satisfacer el objetivo específico 3 de la sección 1.2. 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 ?? de la tesis doctoral.
- Sección 1.5.4: “Optimización de Pareto”. La implementación que complementa la tesis doctoral cumple con el objetivos 4 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 ?? y el Apéndice ?? de la tesis.

Por otra parte, la propuesta presentada en la tesis doctoral se ha seguido un proceso de desarrollo basado en MDA. Por lo tanto, para llevar a cabo la implementación ha sido necesaria combinación de un conjunto de tecnologías MDD entre las que destacan (i) la plataforma de desarrollo *Eclipse* [20], (ii) la tecnología EMF (*Eclipse Modeling Framework*) [21] y GMF (*Graphical Modeling Framework*) [30] y (iii) el lenguaje para transformaciones entre modelos ATL (*Atlas Transformation Language*) [39]. 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, fue desarrollado originalmente por IBM⁹. *Eclipse* es ahora desarrollado por la Fundación *Eclipse*, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto. La característica principal de este IDE es que es una plataforma de programación utilizada para crear entornos integrados de desarrollo puede extenderse por medio de *plugins* con el fin de añadir más características y nuevas funcionalidades.
- ***Eclipse Modeling Framework***. Es un marco de trabajo para modelado que ofrece facilidad de generación de código para construir herramientas y otras aplicaciones basadas en un modelo de datos estructurado. Desde una especificación del modelo descrita en XMI (XML

⁹ <http://www-01.ibm.com/software/rational/eclipse/>

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 anotación Java, documentos XML, o herramientas de modelado como *Rational Rose*, 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.** 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 [47] y OCL 2.0 [46]. 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.

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

Este apartado son implementados los conceptos (sintaxis abstracta) del marco de modelado *i** y la clasificación de requisitos presentada en [24] para proveer al diseñador con un editor gráfico para la especificación de requisitos Web. Concretamente, el metamodelo de requisitos implementado en la tesis doctoral ha sido extendido para incorporar los tipos de requisitos de *i** y la taxonomía presentada en la sección 1.3.1. La Figura 1.2 muestra una captura de pantalla donde puede apreciarse la implementación del metamodelo en *Eclipse*. La sintaxis concreta del metamodelo para requisitos Web se ha implementado para desarrollar un editor gráfico (Figura 1.19) por medio de la tecnología GMF.

El editor (Figura 1.19) *WebREd* (*Web Requirements Editor*) brinda al diseñador de la aplicación Web una interfaz gráfica para especificar en un diagrama los requisitos funcionales y no-funcionales *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 los elementos del marco de modelado *i** pueden ser modificadas seleccionando cada elemento en la vista de propiedades (Figura 1.20).

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 1.21, 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. Por último, 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).

1.5.2 Transformaciones Modelo a Modelo

Con el fin de automatizar el paso del CIM al PIM, se ha desarrollado una serie de reglas de transformación definidas formalmente en el lenguaje QVT (sección 1.3.1) con el fin de aprovechar cada una de las ventajas que ofrece la ingeniería dirigida por modelos. Con esta idea, las transformaciones deben de ser vistas desde una perspectiva de modelado como modelos

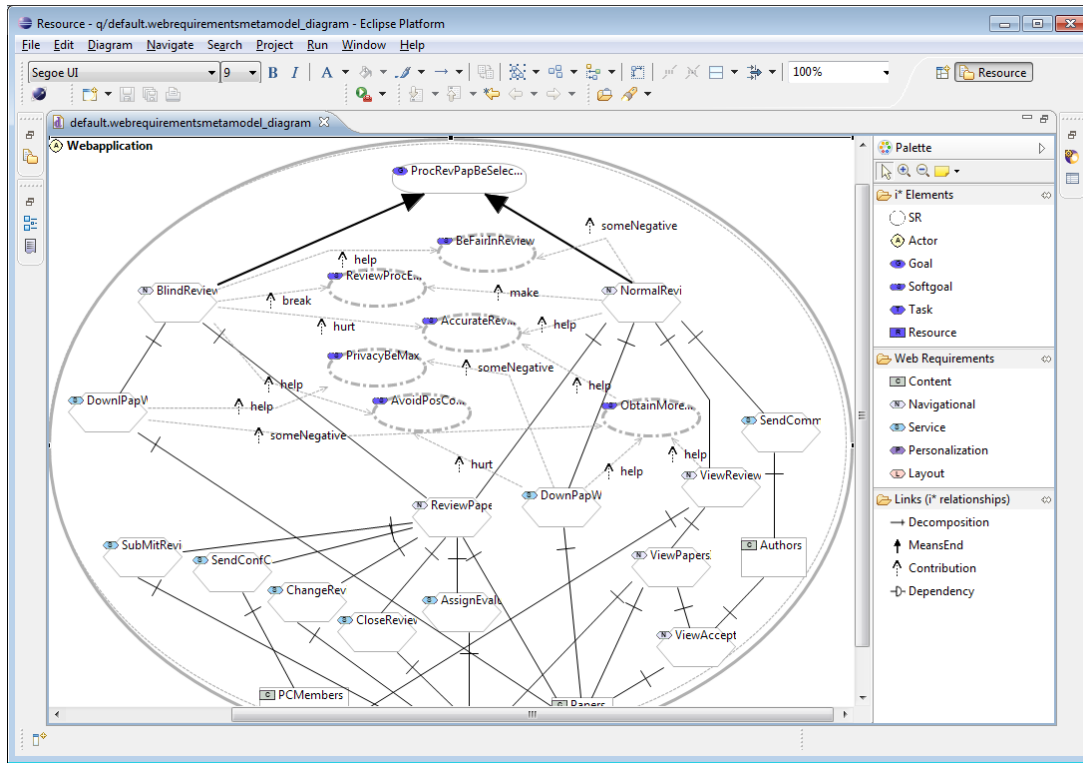


Fig. 1.19. Editor WebRED para la especificación de requisitos Web con i^* .

Navigational		
Core	Property	Value
Appearance	Criticality	open
	Dependency From	
	Dependency To	
	Evaluation	satisficed
	Implemented In	ServerSide
	Initial Status	implemented
	Name	PropEntPorVariante
	Solicited By	client

Fig. 1.20. Vista de propiedades del editor WebRED.

de transformaciones [8]. 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 [32].

La Figura 1.22 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.

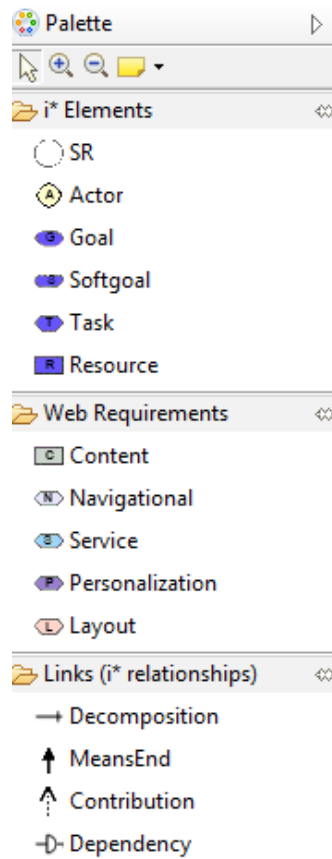


Fig. 1.21. Elementos para el diseño del modelo de requisitos Web con *i**.

```
--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.__vmodel, 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)
}
```

Fig. 1.22. Implementación de la transformación *Content2DomainClass* en el lenguaje ATL.

1.5.3 Trazabilidad de Requisitos

1.5.4 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 acerca de una aplicación Web para la gestión de conferencias (*Conference Management System*)¹⁰. El propósito de la aplicación Web es brindar soporte al proceso de envío, evaluación y selección de los artículos para una conferencia. Para efectos demostrativos, en este caso de estudio nos enfocaremos en la obtención del modelo de dominio, navegación y trazabilidad a partir de la especificación de requisitos.

El primer paso consiste en la especificación de los requisitos para la aplicación Web “*ContentManagementSystem*”. La Figura 1.23, muestra la especificación de requisitos de la aplicación Web (modelo de requisitos de A-OOH). 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*”.

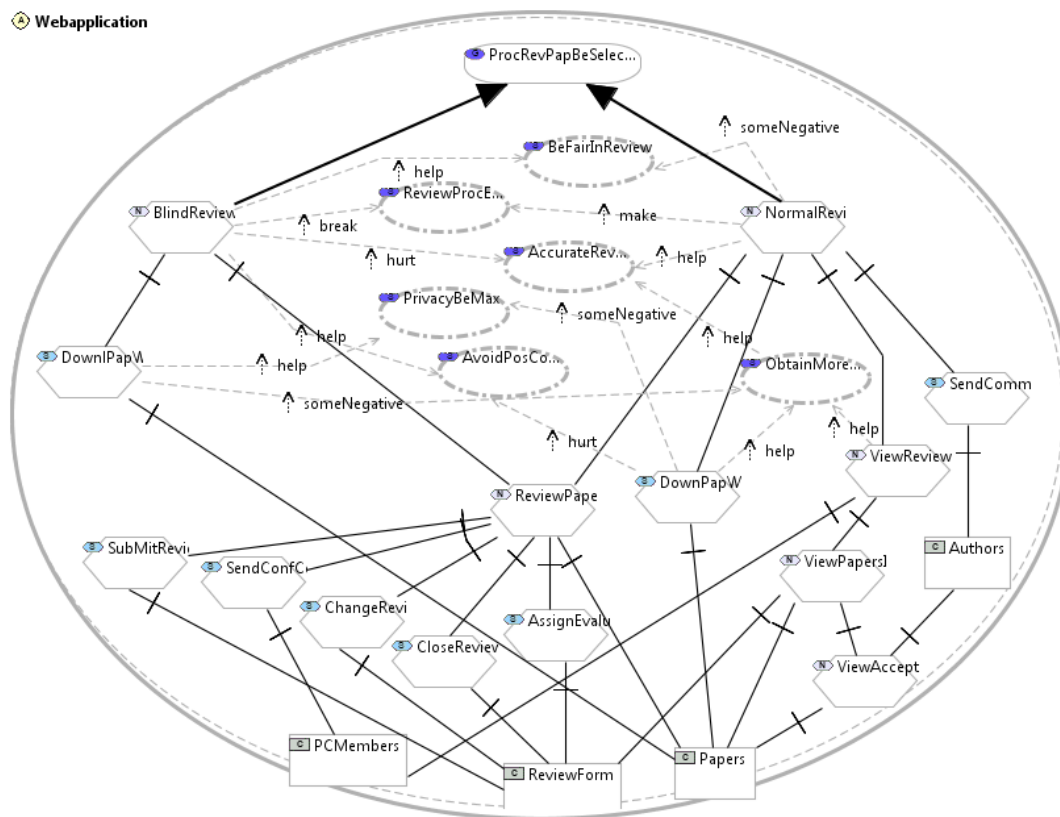


Fig. 1.23. Especificación de requisitos Web con i^* para el caso de estudio.

El siguiente paso consiste en ejecutar las reglas de transformación para derivar automáticamente el modelo de dominio de A-OOH. En la Figura 1.24 se muestra la regla de transformación *Service2ClassOperation* (definida en la sección 1.3.1) implementada en ATL.

¹⁰ La especificación completa del caso de estudio se encuentra en <http://users.dsic.upv.es/~west/iwwest01>

```

--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.__vmodel, 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;
  }
}

```

Fig. 1.24. Reglas ATL para obtener las operaciones de las clases del modelo de dominio A-OOH.

En este caso de estudio, el modelo de dominio de A-OOH se muestra en la Figura 1.25. 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.

Al mismo tiempo que ocurre la generación del modelo de dominio 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 1.26 muestra la parte de la transformación ATL donde se genera el modelo de trazabilidad.

El próximo paso consiste en visualizar el modelo de trazabilidad obtenido por medio de la herramienta AMW *Atlas Model Weaver* [17]. AMW es una herramienta que permite visualizar los enlaces contenidos en los modelos de *weaving*. En la Figura 1.27 se muestra el soporte para trazabilidad de A-OOH, en el lado izquierdo 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.

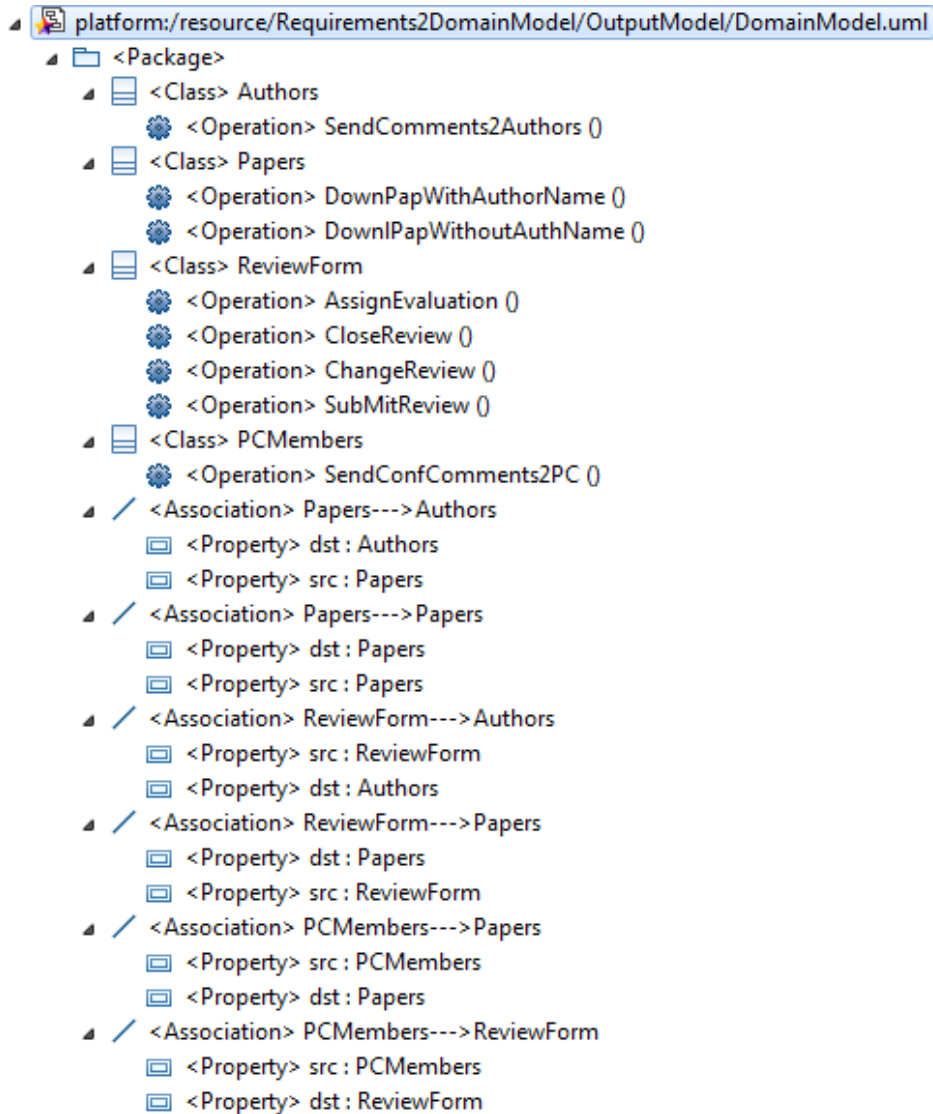


Fig. 1.25. Modelo de dominio A-OOH.

```

--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 <- 'A-OOH(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;
  }
}

```

Fig. 1.26. Extracto de la regla ATL para obtener el modelo de dominio en donde se crea el modelo de trazabilidad.

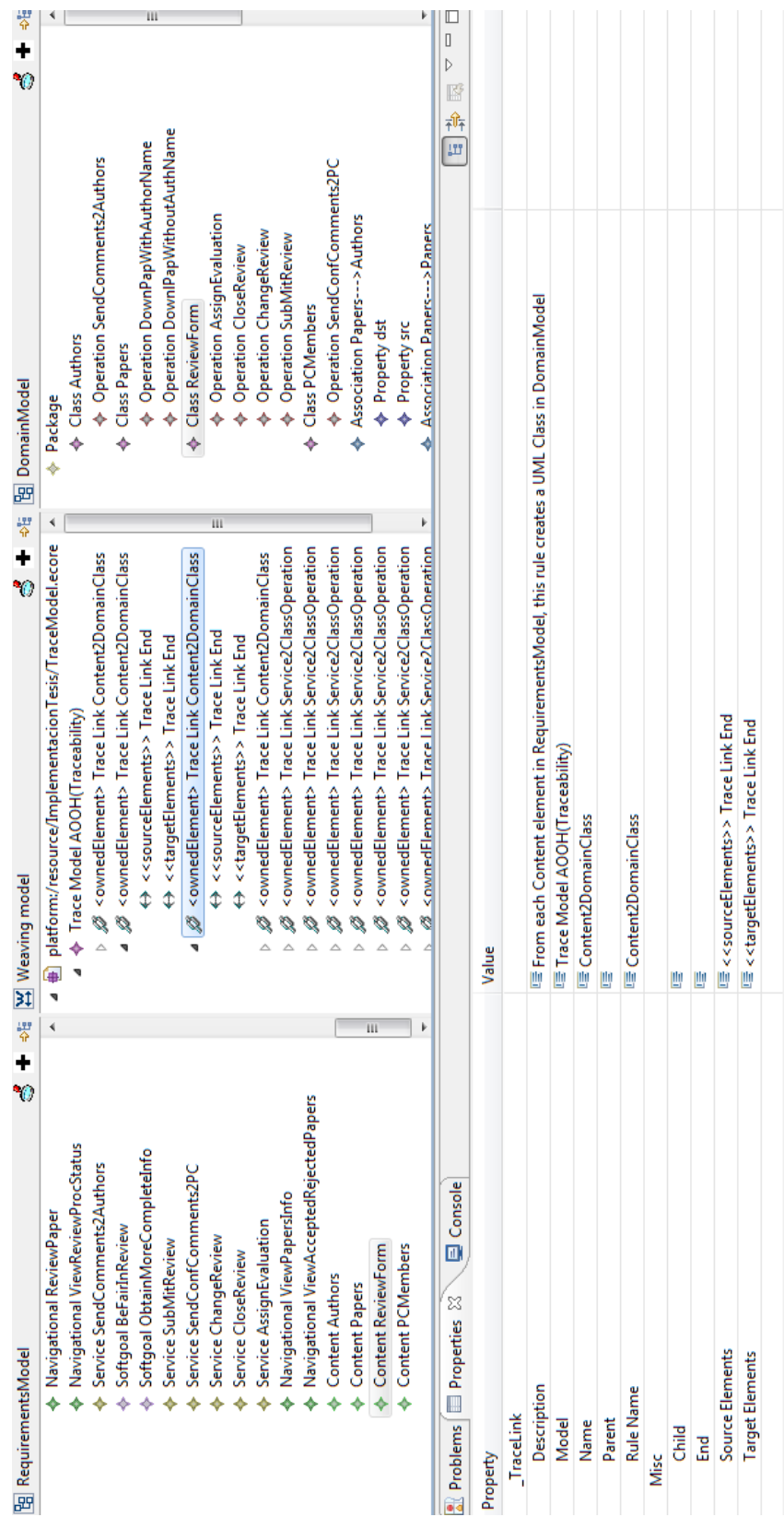


Fig. 1.27. Visualización del modelo de trazabilidad en la herramienta AMW.

El último paso es la obtención del modelo de navegación de A-OOH. En este caso de estudio existe una implementación en ATL (Figura 1.28) de las reglas de transformación presentadas en la sección 1.3.1 para derivar automáticamente el modelo de navegación. El proceso de derivación de un modelo de navegación permite obtener un modelo conceptual donde se especifiquen las rutas que el usuario de la aplicación Web podrá utilizar para desplazarse por el contenido. En la Figura 1.29 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 navegacionales, enlaces navegacionales y nodos navegacionales. Una vez que se tiene este modelo, queda concluido el caso de estudio.

```
-- 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')
  do
    {
      |
    }
  }
}
```

Fig. 1.28. Extracto de la regla ATL para obtener el modelo de navegación.

platform:/resource/Requirements2DomainModel/OutputModel/NavigationalModel.ecore

- ◆ Navigational Model Modelo de Navegación
 - ◆ Navigational Class Authors
 - ◆ Navigational Operation SendComments2Authors
 - ◆ Navigational Class Papers
 - ◆ Navigational Operation DownPapWithAuthorName
 - ◆ Navigational Operation DownIPapWithoutAuthName
 - ◆ Navigational Class ReviewForm
 - ◆ Navigational Operation AssignEvaluation
 - ◆ Navigational Operation CloseReview
 - ◆ Navigational Operation ChangeReview
 - ◆ Navigational Operation SubMitReview
 - ◆ Navigational Class PCMembers
 - ◆ Navigational Operation SendConfComments2PC
 - ◆ Navigational CCollection Home(Menu)
 - ◆ Transversal Link Home(Menu)-> Authors
 - ◆ Transversal Link Home(Menu)-> Papers
 - ◆ Transversal Link Home(Menu)-> ReviewForm
 - ◆ Transversal Link Home(Menu)-> PCMembers
 - ◆ Service Link (NavigationalOperation: SendComments2Authors -> NavigationalClass: SendComments2Authors)
 - ◆ Service Link (NavigationalOperation: DownPapWithAuthorName -> NavigationalClass: DownPapWithAuthorName)
 - ◆ Service Link (NavigationalOperation: DownIPapWithoutAuthName -> NavigationalClass: DownIPapWithoutAuthName)
 - ◆ Service Link (NavigationalOperation: AssignEvaluation -> NavigationalClass: AssignEvaluation)
 - ◆ Service Link (NavigationalOperation: CloseReview -> NavigationalClass: CloseReview)
 - ◆ Service Link (NavigationalOperation: ChangeReview -> NavigationalClass: ChangeReview)
 - ◆ Service Link (NavigationalOperation: SubMitReview -> NavigationalClass: SubMitReview)

Property	Value
Is Initial	false
Name	PCMembers
Personalization Rules	None

Fig. 1.29. Modelo de navegación del caso de estudio.

1.6 Conclusiones

La Web cambia constantemente debido a la evolución constante en las tecnologías de implementación. Por tanto, las metodologías ingenieriles necesitan adaptarse a los cambios con el fin de ofrecer al diseñador una aproximación sistemática e integral para el desarrollo de aplicaciones Web. Además, 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.

Para tratar con estas cuestiones, en esta tesis doctoral, se ha presentado una propuesta dirigida por modelos que permite (i) la especificación de un modelo de requisitos a nivel conceptual de manera integral, sistemática y bien estructurada y (ii) la derivación automática de los modelos conceptuales de dominio y navegación. Por lo tanto, la propuesta permite a los diseñadores disminuir el nivel de complejidad en el desarrollo de aplicaciones Web, ahorrando tiempo y esfuerzo con lo que el costo del proyecto disminuye. Posteriormente, se ha añadido a la propuesta dirigida por modelos un soporte al diseñador por medio de una gestión integral de requisitos con el fin de asegurar la trazabilidad de los requisitos de CIM a PIM, análisis de impacto y la implementación de requisitos funcionales en base a distintas alternativas de diseño considerando la maximización de los requisitos no-funcionales. También, se ha realizado una implementación en *Eclipse* para apoyar cada parte de esta propuesta.

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, a la vez permite la generación de los modelos conceptuales de la aplicación Web en un proceso con un alto grado de automatización.

References

1. J. A. Aguilar, I. Garrigós, S. Casteleyn, and J.-N. Mazón. Optimizing non-function requirements in web applications. In *WISM (In press) held in conjunction with ER 2011*, 2011.
2. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Modelos de weaving para trazabilidad de requisitos web en a-ooh. In *In DSDM: Actas del VII Taller sobre Desarrollo de Software Dirigido por Modelos, JISBD, Congreso Espanol de Informatica (CEDI)*, pages 146–155, Valencia, Espana, 2010. SISTEDES.
3. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Impact analysis of goal-oriented requirements in web engineering. In *ICCSA (5)*, pages 421–436, 2011.
4. J. A. Aguilar, I. Garrigós, J.-N. Mazón, and J. Trujillo. An mda approach for goal-oriented requirement analysis in web engineering. *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 *6th Web Information Systems and Technologies (WEBIST)*, volume 2, pages 187–190, Valencia, Spain, 2010. SciTePress Digital Library.
6. R. Arnold and S. Bohner. Impact analysis-towards a framework for comparison. In *Software Maintenance ,1993. CSM-93, Proceedings., Conference on*, pages 292 –301, sep 1993.
7. M. Barbero, M. Del Fabro, and J. Bézivin. Traceability and provenance issues in global model management. In *3rd ECMDA-Traceability Workshop*, 2007.
8. J. Bézivin, F. B. ”uttner, M. Gogolla, F. Jouault, I. Kurtev, and A. Lindow. Model transformations? transformation models! *Model Driven Engineering Languages and Systems*, pages 440–453, 2006.
9. D. Bolchini and J. Mylopoulos. From task-oriented to goal-oriented web requirements analysis. In *WISE ’03: Proceedings of the Fourth International Conference on Web Information Systems Engineering*, page 166, Washington, DC, USA, 2003. IEEE Computer Society.
10. A. Bozzon, S. Comai, and P. Fraternali. Current Research on the Design of Web 2.0 Applications Based on Model-Driven Approaches. *ICWE 2008 Workshops*, pages 25–31, 2008.
11. 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 ’06*, page 353, New York, New York, USA, 2006. ACM Press.
12. S. Casteleyn, W. V. Woensel, and G.-J. Houben. A semantics-based aspect-oriented approach to adaptation in web engineering. In *Hypertext*, pages 189–198, 2007.
13. S. Ceri, P. Fraternali, and A. Bongio. Web modeling language (webml): a modeling language for designing web sites. *International Journal of Computer and Telecommunications Networking*, 33(1-6):137–157, 2000.
14. Y. Collette and P. Siarry. *Multiobjective optimization: principles and case studies*. Springer Verlag, 2003.
15. 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.
16. M. Del Fabro, J. Bézivin, and P. Valduriez. Weaving models with the eclipse amw plugin. In *Eclipse Modeling Symposium, Eclipse Summit Europe*, volume 2006, 2006.
17. M. Del Fabro, J. Bézivin, and P. Valduriez. Weaving Models with the Eclipse AMW plugin. In *Eclipse Modeling Symposium, Eclipse Summit Europe*, volume 2006, 2006.
18. P. Dolog and J. Stage. ADRIA: A Method for Abstract Design of Rich Internet Applications for the Web 2.0.
19. P. Dolog and J. Stage. Designing interaction spaces for rich internet applications with uml. *Web Engineering*, pages 358–363, 2007.
20. Eclipse. <http://www.eclipse.org/>, 2010.
21. Eclipse Modeling Framework. <http://www.eclipse.org/modeling/emf/>, 2010.
22. M. Escalona, M. Mejías, and J. Torres. Developing systems with NDT & NDT-Tool. In *13th International conference on information systems development: methods and tools, theory and practice, Vilna, Lithuania*, pages 149–59, 2004.
23. M. J. Escalona and G. Aragón. Ndt. a model-driven approach for web requirements. *IEEE Transactions on Software Engineering*, 34(3):377–390, 2008.
24. M. J. Escalona and N. Koch. Requirements engineering for web applications - a comparative study. *J. Web Eng.*, 2(3):193–212, 2004.
25. I. Garrigós. *A-OOH: Extending Web Application Design with Dynamic Personalization*. PhD thesis, University of Alicante, Spain, 2008.
26. I. Garrigós, J.-N. Mazón, and J. Trujillo. A requirement analysis approach for using i* in web engineering. In *ICWE*, pages 151–165, 2009.

27. P. Godfrey, R. Shipley, and J. Gryz. Algorithms and analyses for maximal vector computation. *The VLDB Journal*, 16:5–28, January 2007.
28. J. Gómez, C. Cachero, and O. Pastor. Extending a conceptual modelling approach to web application design. In *CAiSE '00: Proceedings of the 12th International Conference on Advanced Information Systems Engineering*, pages 79–93, London, UK, 2000. Springer-Verlag.
29. O. Gotel and A. Finkelstein. An analysis of the requirements traceability problem. In *Proceedings of the First International Conference on Requirements Engineering*, pages 94–101, 1994.
30. Graphical Modeling Framework. <http://www.eclipse.org/modeling/gmp/>, 2010.
31. i* wiki. <http://istar.rwth-aachen.de>.
32. F. Jouault and I. Kurtev. On the architectural alignment of atl and qvt. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 1188–1195. ACM, 2006.
33. N. Koch. The expressive power of uml-based web engineering. In *International Workshop on Web-oriented Software Technology (IWWOST)*, pages 40–41, 2002.
34. 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.
35. N. Koch, S. Meliá, N. Moreno, V. Pelechano, F. Sánchez, and J. Vara. Model-driven web engineering. *Upgrade, The European Journal for the Informatics Professional. vIX i2*, pages 40–45.
36. 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 '9*, pages 283–291, Berlin, Heidelberg, 2009. Springer-Verlag.
37. N. Koch, G. Zhang, and M. J. E. Cuaresma. Model transformations from requirements to web system design. In *ICWE*, pages 281–288, 2006.
38. 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.
39. A. T. Language. <http://www.eclipse.org/m2m/at1/>.
40. M. Linaje, J. C. Preciado, and F. Sanchez-Figueroa. Engineering rich internet application user interfaces over legacy web models. *IEEE Internet Computing*, 11:53–59, 2007.
41. M. Lindvall and K. Sandahl. How well do experienced software developers predict software change? *J. Syst. Softw.*, 43:19–27, October 1998.
42. L. Machado, O. Filho, and J. a. Ribeiro. Uwe-r: an extension to a web engineering methodology for rich internet applications. *WSEAS Trans. Info. Sci. and App.*, 6:601–610, April 2009.
43. S. Meliá, J. Gómez, S. Pérez, and O. Díaz. A model-driven development for GWT-based Rich Internet Applications with OOH4RIA. In *Web Engineering, 2008. ICWE'08. Eighth International Conference on*, pages 13–23. IEEE, 2008.
44. Model Driven Architecture. <http://www.omg.org/mda/>.
45. 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.
46. Object Constraint Language, Version 2.0. <http://www.omg.org/spec/OCL/2.0/>.
47. Object Management Group. <http://www.omg.org>.
48. O. Pastor, S. M. Abrahão, and J. Fons. An object-oriented approach to automate web applications development. In *EC-Web 2001: Proceedings of the Second International Conference on Electronic Commerce and Web Technologies*, pages 16–28, London, UK, 2001. Springer-Verlag.
49. J. Preciado, M. Linaje, S. Comai, and F. Sanchez-Figueroa. Designing rich internet applications with web engineering methodologies. In *Web Site Evolution, 2007. WSE 2007. 9th IEEE International Workshop on*, pages 23–30. IEEE, Oct. 2007.
50. 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*, 14(13):2239–2254, 2008.
51. J. C. Preciado, M. Linaje, F. Sanchez, and S. Comai. Necessity of methodologies to model rich internet applications. In *Proceedings of the Seventh IEEE International Symposium on Web Site Evolution*, pages 7–13, Washington, DC, USA, 2005. IEEE Computer Society.
52. 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.
53. 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.
54. QVT Language. <http://www.omg.org/cgi-bin/doc?ptc/2005-11-01>.
55. D. Schmidt. Guest editor's introduction: Model-driven engineering. *Computer*, 39(2):25–31, 2006.

56. D. Schwabe and G. Rossi. The object-oriented hypermedia design model. *Communications of the ACM*, 38(8):45–46, 1995.
57. Software Engineering Institute. <http://www.sei.cmu.edu/>, 2010.
58. I. Sommerville. *Software Engineering*. Addison-Wesley, 6th edition, 2001.
59. I. Sommerville. *CMMI for Development®: Guidelines for Process Integration and Product Improvement*. Addison-Wesley Professional, 3rd edition, 2011.
60. F. Szidarovszky, M. Gershon, and L. Duckstein. *Techniques for multiobjective decision making in systems management*. Elsevier, 1986.
61. Unified Modeling Language. <http://www.uml.org>.
62. M. Urbietá, G. Rossi, J. Ginzburg, and D. Schwabe. Designing the interface of rich internet applications, 2007.
63. 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.
64. J. Wright. A Modelling Language for Interactive Web Applications. In *Automated Software Engineering, 2009. ASE'09. 24th IEEE/ACM International Conference on*, pages 689–692. IEEE, 2010.
65. 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.
66. 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.
67. E. Yu. *Modelling Strategic Relationships for Process Reengineering*. PhD thesis, University of Toronto, Canada, 1995.
68. E. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *RE*, pages 226–235, 1997.

References

1. J. A. Aguilar, I. Garrigós, S. Casteleyn, and J.-N. Mazón. Optimizing non-functional requirements in web applications. In *WISM (In press) held in conjunction with ER 2011*, 2011.
2. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Modelos de weaving para trazabilidad de requisitos web en a-ooh. In *In DSDM: Actas del VII Taller sobre Desarrollo de Software Dirigido por Modelos, JISBD, Congreso Espanol de Informatica (CEDI)*, pages 146–155, Valencia, Espana, 2010. SISTEDES.
3. J. A. Aguilar, I. Garrigós, and J.-N. Mazón. Impact analysis of goal-oriented requirements in web engineering. In *ICCSA (5)*, pages 421–436, 2011.
4. J. A. Aguilar, I. Garrigós, J.-N. Mazón, and J. Trujillo. An mda approach for goal-oriented requirement analysis in web engineering. *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 *6th Web Information Systems and Technologies (WEBIST)*, volume 2, pages 187–190, Valencia, Spain, 2010. SciTePress Digital Library.
6. R. Arnold and S. Bohner. Impact analysis-towards a framework for comparison. In *Software Maintenance ,1993. CSM-93, Proceedings., Conference on*, pages 292 –301, sep 1993.
7. M. Barbero, M. Del Fabro, and J. Bézivin. Traceability and provenance issues in global model management. In *3rd ECMDA-Traceability Workshop*, 2007.
8. J. Bézivin, F. B. "uttner, M. Gogolla, F. Jouault, I. Kurtev, and A. Lindow. Model transformations? transformation models! *Model Driven Engineering Languages and Systems*, pages 440–453, 2006.
9. D. Bolchini and J. Mylopoulos. From task-oriented to goal-oriented web requirements analysis. In *WISE '03: Proceedings of the Fourth International Conference on Web Information Systems Engineering*, page 166, Washington, DC, USA, 2003. IEEE Computer Society.
10. A. Bozzon, S. Comai, and P. Fraternali. Current Research on the Design of Web 2.0 Applications Based on Model-Driven Approaches. *ICWE 2008 Workshops*, pages 25–31, 2008.
11. 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 '06*, page 353, New York, New York, USA, 2006. ACM Press.
12. S. Casteleyn, W. V. Woensel, and G.-J. Houben. A semantics-based aspect-oriented approach to adaptation in web engineering. In *Hypertext*, pages 189–198, 2007.
13. S. Ceri, P. Fraternali, and A. Bongio. Web modeling language (webml): a modeling language for designing web sites. *International Journal of Computer and Telecommunications Networking*, 33(1-6):137–157, 2000.
14. Y. Collette and P. Siarry. *Multiobjective optimization: principles and case studies*. Springer Verlag, 2003.
15. 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.
16. M. Del Fabro, J. Bézivin, and P. Valduriez. Weaving models with the eclipse amw plugin. In *Eclipse Modeling Symposium, Eclipse Summit Europe*, volume 2006, 2006.
17. M. Del Fabro, J. Bézivin, and P. Valduriez. Weaving Models with the Eclipse AMW plugin. In *Eclipse Modeling Symposium, Eclipse Summit Europe*, volume 2006, 2006.
18. P. Dolog and J. Stage. ADRIA: A Method for Abstract Design of Rich Internet Applications for the Web 2.0.
19. P. Dolog and J. Stage. Designing interaction spaces for rich internet applications with uml. *Web Engineering*, pages 358–363, 2007.
20. Eclipse. <http://www.eclipse.org/>, 2010.
21. Eclipse Modeling Framework. <http://www.eclipse.org/modeling/emf/>, 2010.
22. M. Escalona, M. Mejías, and J. Torres. Developing systems with NDT & NDT-Tool. In *13th International conference on information systems development: methods and tools, theory and practice, Vilna, Lithuania*, pages 149–59, 2004.
23. M. J. Escalona and G. Aragón. Ndt. a model-driven approach for web requirements. *IEEE Transactions on Software Engineering*, 34(3):377–390, 2008.
24. M. J. Escalona and N. Koch. Requirements engineering for web applications - a comparative study. *J. Web Eng.*, 2(3):193–212, 2004.
25. I. Garrigós. *A-OOH: Extending Web Application Design with Dynamic Personalization*. PhD thesis, University of Alicante, Spain, 2008.
26. I. Garrigós, J.-N. Mazón, and J. Trujillo. A requirement analysis approach for using i* in web engineering. In *ICWE*, pages 151–165, 2009.

27. P. Godfrey, R. Shipley, and J. Gryz. Algorithms and analyses for maximal vector computation. *The VLDB Journal*, 16:5–28, January 2007.
28. J. Gómez, C. Cachero, and O. Pastor. Extending a conceptual modelling approach to web application design. In *CAiSE '00: Proceedings of the 12th International Conference on Advanced Information Systems Engineering*, pages 79–93, London, UK, 2000. Springer-Verlag.
29. O. Gotel and A. Finkelstein. An analysis of the requirements traceability problem. In *Proceedings of the First International Conference on Requirements Engineering*, pages 94–101, 1994.
30. Graphical Modeling Framework. <http://www.eclipse.org/modeling/gmp/>, 2010.
31. i* wiki. <http://istar.rwth-aachen.de>.
32. F. Jouault and I. Kurtev. On the architectural alignment of atl and qvt. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 1188–1195. ACM, 2006.
33. N. Koch. The expressive power of uml-based web engineering. In *International Workshop on Web-oriented Software Technology (IWWOST)*, pages 40–41, 2002.
34. 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.
35. N. Koch, S. Meliá, N. Moreno, V. Pelechano, F. Sánchez, and J. Vara. Model-driven web engineering. *Upgrade, The European Journal for the Informatics Professional. vIX i2*, pages 40–45.
36. 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 '9*, pages 283–291, Berlin, Heidelberg, 2009. Springer-Verlag.
37. N. Koch, G. Zhang, and M. J. E. Cuaresma. Model transformations from requirements to web system design. In *ICWE*, pages 281–288, 2006.
38. 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.
39. A. T. Language. <http://www.eclipse.org/m2m/at1/>.
40. M. Linaje, J. C. Preciado, and F. Sanchez-Figueroa. Engineering rich internet application user interfaces over legacy web models. *IEEE Internet Computing*, 11:53–59, 2007.
41. M. Lindvall and K. Sandahl. How well do experienced software developers predict software change? *J. Syst. Softw.*, 43:19–27, October 1998.
42. L. Machado, O. Filho, and J. a. Ribeiro. Uwe-r: an extension to a web engineering methodology for rich internet applications. *WSEAS Trans. Info. Sci. and App.*, 6:601–610, April 2009.
43. S. Meliá, J. Gómez, S. Pérez, and O. Díaz. A model-driven development for GWT-based Rich Internet Applications with OOH4RIA. In *Web Engineering, 2008. ICWE'08. Eighth International Conference on*, pages 13–23. IEEE, 2008.
44. Model Driven Architecture. <http://www.omg.org/mda/>.
45. 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.
46. Object Constraint Language, Version 2.0. <http://www.omg.org/spec/OCL/2.0/>.
47. Object Management Group. <http://www.omg.org>.
48. O. Pastor, S. M. Abrahão, and J. Fons. An object-oriented approach to automate web applications development. In *EC-Web 2001: Proceedings of the Second International Conference on Electronic Commerce and Web Technologies*, pages 16–28, London, UK, 2001. Springer-Verlag.
49. J. Preciado, M. Linaje, S. Comai, and F. Sanchez-Figueroa. Designing rich internet applications with web engineering methodologies. In *Web Site Evolution, 2007. WSE 2007. 9th IEEE International Workshop on*, pages 23–30. IEEE, Oct. 2007.
50. 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*, 14(13):2239–2254, 2008.
51. J. C. Preciado, M. Linaje, F. Sanchez, and S. Comai. Necessity of methodologies to model rich internet applications. In *Proceedings of the Seventh IEEE International Symposium on Web Site Evolution*, pages 7–13, Washington, DC, USA, 2005. IEEE Computer Society.
52. 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.
53. 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.
54. QVT Language. <http://www.omg.org/cgi-bin/doc?ptc/2005-11-01>.
55. D. Schmidt. Guest editor's introduction: Model-driven engineering. *Computer*, 39(2):25–31, 2006.

56. D. Schwabe and G. Rossi. The object-oriented hypermedia design model. *Communications of the ACM*, 38(8):45–46, 1995.
57. Software Engineering Institute. <http://www.sei.cmu.edu/>, 2010.
58. I. Sommerville. *Software Engineering*. Addison-Wesley, 6th edition, 2001.
59. I. Sommerville. *CMMI for Development®: Guidelines for Process Integration and Product Improvement*. Addison-Wesley Professional, 3rd edition, 2011.
60. F. Szidarovszky, M. Gershon, and L. Duckstein. *Techniques for multiobjective decision making in systems management*. Elsevier, 1986.
61. Unified Modeling Language. <http://www.uml.org>.
62. M. Urbiet, G. Rossi, J. Ginzburg, and D. Schwabe. Designing the interface of rich internet applications, 2007.
63. F. Valverde. *OOWS 2.0: Un Método de Ingeniería Web Dirigido por Modelos para la Producción de Aplicaciones Web 2.0*. PhD thesis, Universidad Politecnica de Valencia, 2010.
64. J. Wright. A Modelling Language for Interactive Web Applications. In *Automated Software Engineering, 2009. ASE'09. 24th IEEE/ACM International Conference on*, pages 689–692. IEEE, 2010.
65. 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.
66. 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.
67. E. Yu. *Modelling Strategic Relationships for Process Reengineering*. PhD thesis, University of Toronto, Canada, 1995.
68. E. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *RE*, pages 226–235, 1997.