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

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

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

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

Key Words: MDA, Web engineering, goal-oriented requirements, requirements analysis, model transformations
Category: D.2.1, D.2.2

1 Introduction

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

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

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

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

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

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

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

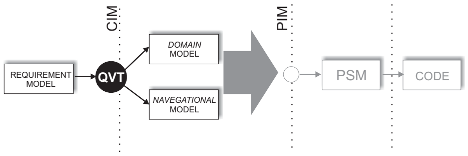


Figure 1: Our MDA approach for goal-oriented requirement analysis in Web engineering.

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

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

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

2 Related Work

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

UWE (*UML-based Web Engineering*) [Koch et al. 2006]. Regarding to the requirements specification, UWE initially used use case diagrams with textual descriptions, currently this phase is carried out by means of UML profiles in combination with a metamodel called WebRE (developed by the authors of NDT and UWE), more information can be consulted in [Escalona and Koch 2006]. Through the requirements model, UWE can generate content, navigation, presentation and process models through a series of QVT transformations. With regard to the implementation, a plugin called MagicUWE was developed to be used within the CASE tool MagicDraw (http://www.magicdraw.com) [Bush and Koch 2009], which allows UWE to provide a professional CASE tool, although it does not take into account the analysis of requirements.

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

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

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

OOHDM (*Object Oriented Hypermedia Design Model*) [Schwabe 1998]. This approach is based on the Object Oriented paradigm. The requirements specification is divided in (i) identification of roles, (ii) specification of scenarios, (iii) specification of Use Cases, (iv) specification of User Interaction Diagrams and (v) validation of User Interaction Diagrams and Use Cases. This approach defines guidelines to define conceptual and navigational schemas by means of rules described in natural language, this rules indicate how the conceptual and navigational schema can be defined from User Interaction Diagrams. OOHDM does not have a tool support for the RA phase. Furthermore, this approach does not support the definition of conceptual models from requirements specification.

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

Table 1: Summary of methodological approaches.

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

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

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

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

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

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

3 MDA artifacts for modeling requirements in Web engineering

In this section, we present a proposal which provides a way of specifying requirements by using *i** and A-OOH *(Adaptive Object Oriented Hypermedia method)* [Garrigós 2008] in an MDA context. A-OOH is the extension of the OO-H modeling method [Cachero and Gómez 2002], which includes the definition of adaptation strategies. This approach has also been extended with UML-profiles so all the conceptual models are UML-compliant (see Sect. 3.1). It is worth noting that we use A-OOH for demonstration purposes but the proposal could be applied to any Web modeling method. The automatic generation of both the Domain and Navegational conceptual models from the specified requirements model is also supported (see Sect. 3.2) by means of QVT rules. Designers only have to focus on specifying the requirements, and refining the generated conceptual models if needed.

3.1 Web requirements specification as a CIM

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

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

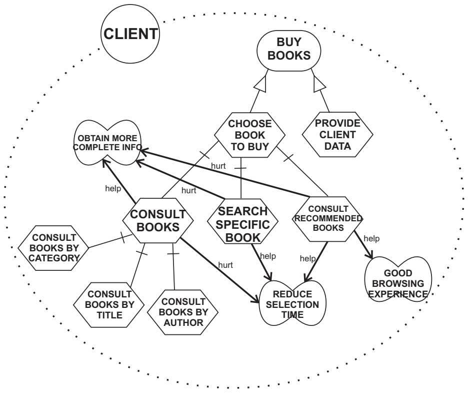


Figure 2: Modeling the client in an SR model.

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

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

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

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

3.2 Definition of conceptual models as PIM

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

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

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

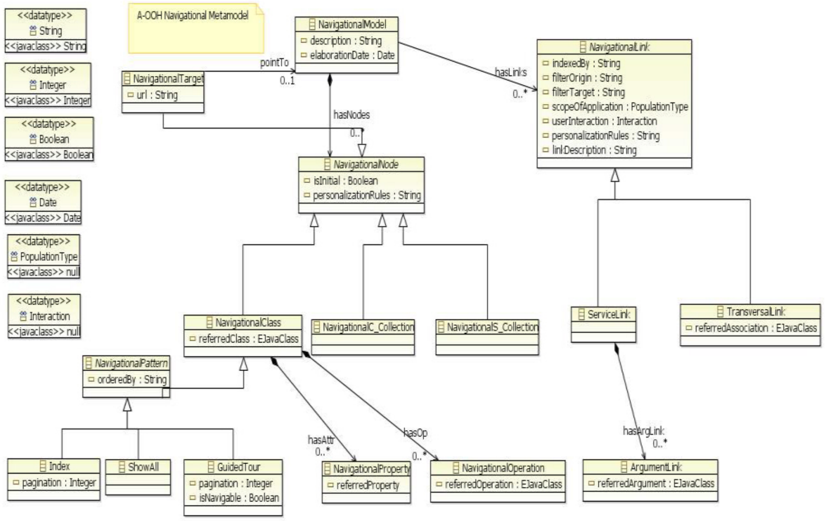
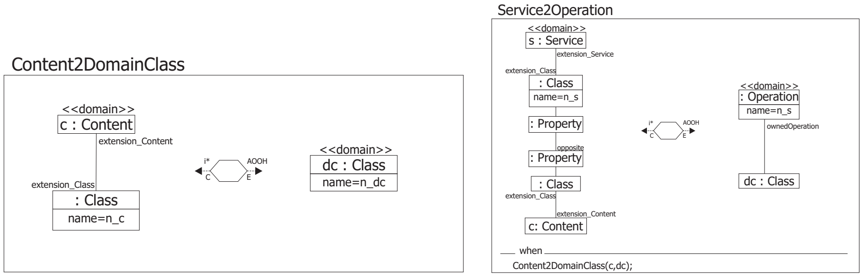


Figure 3: Excerpt of the A-OOH Navigational metamodel.

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

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

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



(a) How to derive the DM classes.

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

Figure 4: QVT rules for the Domain Model.

3.3 Deriving the Domain model

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

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

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

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

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

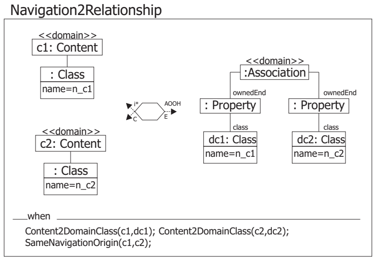


Figure 5: QVT rule for associations between DM classes.

3.4 Deriving the Navigational model

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

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

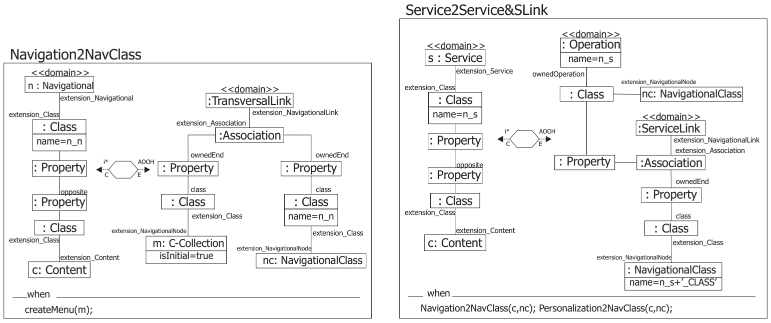
Navigation2TLink. This rule checks the navigational requirements associated with one or more content requirements in the SR model , if it is detected, then a transversal link is added from the NC that represents the root navigational requirement to each of the NCs representing the associated navigational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

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

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

4 Sample Application of our Approach

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



(a) How to obtain navigational classes for NM.

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

Figure 6: QVT rules for the Navigational Model.

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

4.1 Requirements specification

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

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

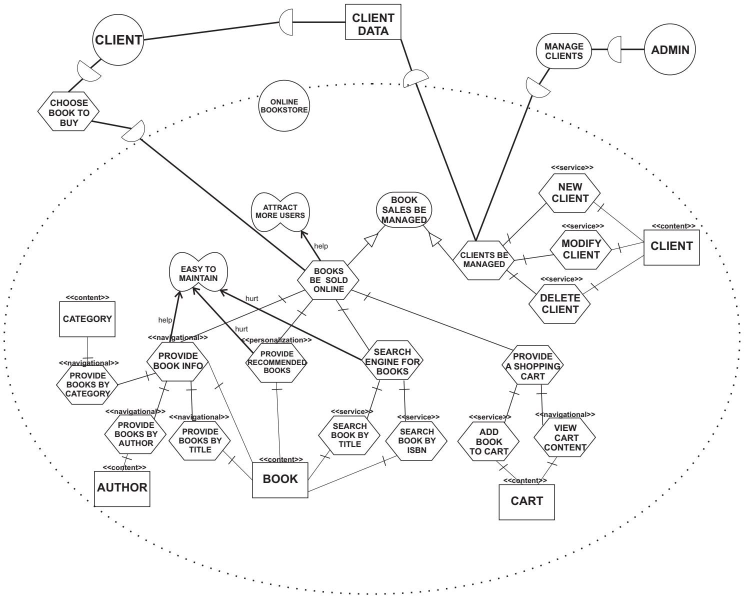
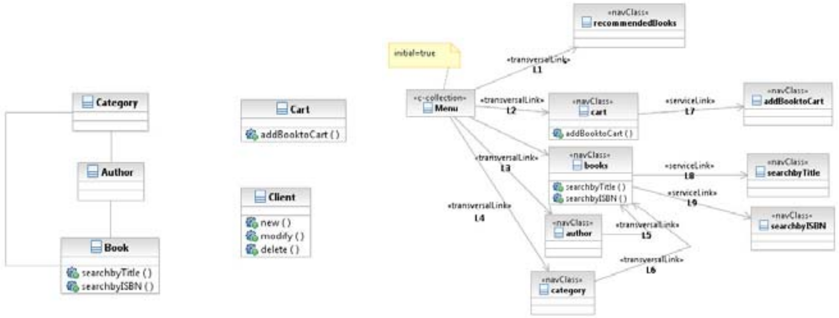


Figure 7: Modeling the online bookstore in an SR model and the SD model.

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

4.2 Domain Model

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



(a) The Domain Model.

(b) The Navigational Model.

Figure 8: The Domain and Navigational Models.

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

4.3 Navigational Model

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

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

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

5 Implementation Framework

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

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

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

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

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

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

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

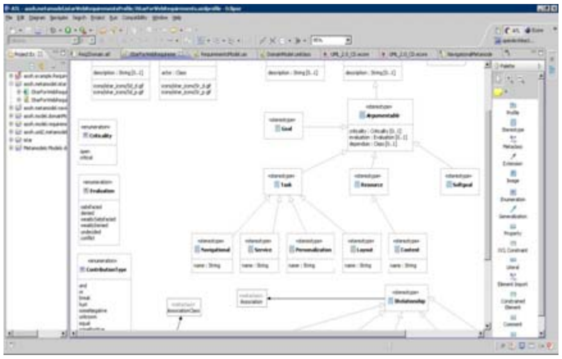


Figure 9: UML profile for Web requirements implemented in Eclipse.

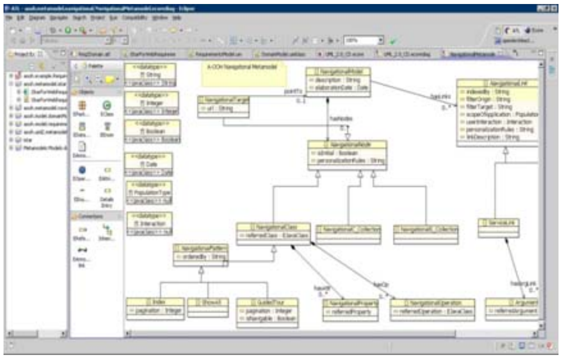


Figure 10: A-OOH Navigational metamodel implemented in Eclipse.

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

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

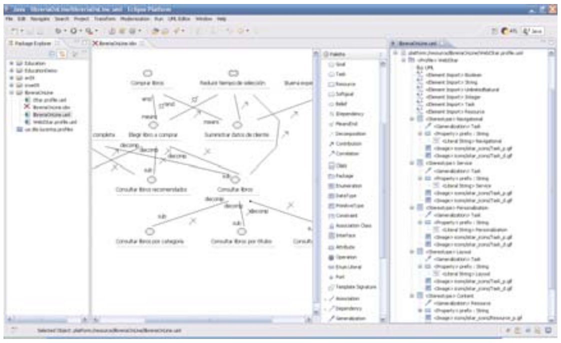


Figure 11: Screen capture of our prototype.

6 Conclusions and Future Work

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

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

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

Acknowledgements

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

References

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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