Metamodeling the Requirements of Web Systems

María José Escalona[1] and Nora Koch[2,3]

[1] Universidad de Sevilla
[2] Ludwig-Maximilians-Universität München
[3] FAST GmbH
mjescalona@us.es, kochn@pst.ifi.lmu.de

Abstract. A detailed requirements analysis is best practice in the development of traditional software. Conversely, the importance of requirements engineering for Web systems is still underestimated. Only few Web methodologies provide an approach for the elicitation of requirements and techniques for their specifi-cation. This paper focuses on specification through requirements models of Web systems. We present a metamodel, which contains the key concepts needed for the requirements specification of Web systems. The benefit of such a metamodel is twofold: (1) The key concepts are used for the definition of a common modeling language: a UML profile for Web requirements. (2) The elements of the metamodel are mapped to the modeling constructs of the differ-ent Web methodologies. In this way the prerequisite for model-to-model trans-formations is given, which allows to build different views of the requirements of a Web system using different Web methodologies.

Keywords: Web Engineering, Metamodel, UML Profile, Web Requirements.

1 Introduction

Web Engineering is a new area of Software Engineering, which focuses on the devel-opment of Web Systems (Kappel et al., 2003). In the last years, several approaches have been proposed for the Web environment. These methods provide specific model-ing elements for the analysis and design and most of them define a proprietary nota-tion used for the graphical representation of the elements. Almost all methods propose specific processes to support the systematic or semi-automatic development of Web applications. However, only few of the existing Web methodologies start the devel-opment cycle with a detailed requirements analysis (Escalona & Koch, 2004).

Conversely, the requirements analysis is considered by all software engineering approaches to be a key step in the development of successful software systems (Lowe & Ecklund, 2002). Empirical data demonstrate that efforts invested in a de-tailed requirements analysis considerably reduce drawbacks in later phases of the de-velopment (Sommerville & Ransom, 2005).

In this work we present an approach which aims to improve the development of Web applications reinforcing the requirements engineering aspects of the methods. We start with an analysis of the requirements of *requirements specification* of Web systems. We take into account both general characteristics of Web applications and

how Web engineering deals with requirements. We restrict the analysis to those methodologies that support requirements engineering by a process, a notation and/or tool support. The most relevant methods fulfilling these restrictions are NDT (Escalona, 2004), OOHDM (Rossi & Schwabe, 1998), UWE (Koch & Kraus, 2002) and W2000 (Baresi et al., 2003).

The key concepts related to the requirements engineering of Web systems and their relationships were identified through the analysis of these different Web engineering approaches and the review of literature. We have developed a common *metamodel* for the representation of concepts and relationships of Web requirements engineering (WebRE). The metamodel is visualized with a UML class diagram and constitutes the basis for the definition of a so called *UML profile for Web requirements* and tool support. Such a UML profile contains a set of modeling elements for which a specific graphical notation can be defined.

The advantage of the metamodel and its associated profile is twofold: On the one hand it offers a common modeling language of requirements engineering. This common modeling language provides NDT with a graphical notation and extends current methodologies as UWE and W2000 with additional modeling elements. And it provides OOHDM with a standard notation for User Interaction Diagrams (UIDs) as an alternative to its proprietary notation. On the other hand the mapping of methods to the metamodel is the basis for the definition of model transformations (PIM to PIM transformations) from models specified with one method, e.g. in NDT, to models of another method, e.g. UWE.

The vision is to integrate the requirements model in the model-driven process, more precisely, to start the model-driven process with a requirements model.

The remainder of this paper is structured as follows: Section 2 gives an overview of the state of the art of requirements analysis in Web engineering. Section 3 presents the metamodel that comprises the elements needed to model requirements of Web applications. Building on the metamodel a UML profile is defined in Section 4. Finally, in Section 5 a set of conclusions and future work are outlined.

2 Requirements in Web Engineering

The aim of a requirements engineering phase is always to obtain a stable set of requirements, which serves as basis for the further steps in the development process. Three activities are used to achieve this goal: elicitation, specification, and validation of requirements (Lowe & Hall, 1999).

The *elicitation of requirements* is the activity by means of which the functionalities of the system to be built are collected from any available source. The overall requirements elicitation objectives for software engineering remain unchanged when applied to Web systems. However, the specific objectives for Web systems become: (1) the identification of content requirements, (2) the identification of the functional requirements in terms of navigation needs and businesses processes, and (3) the definition of interaction scenarios for different groups of Web users.

Requirements specification consists of producing a description of the requirements. Different techniques can be used for the specification: from informal textual description to formal specification in languages like Z (Kappel et al., 2003; Escalona & Koch, 2004).

Finally, *requirements validation* consists of checking the requirements specification in order to establish whether the Web application user's needs are fulfilled.

This work focuses on requirements specification.

2.1 An Overview of Requirements Specification for Web Systems

Requirements specification can be focused on the description of the problems or the solutions (Wieringa, 2004). Problem description is goal-oriented; in contrast solution description is pattern-oriented. In both cases, it is important to write specifications or build models that are understandable for managers, provide sufficient information for developers, and allow validation of the models by final users. The development in the Web domain is influenced by a higher reliability of the user interface, volatility of user requirements and the business model, an unpredictable publishing environment and fine-grained evolution and maintenance.

Requirements specifications need to be described in documents in the degree of detail and formality that is appropriate for the corresponding project. The appropriateness of the specification technique is mainly established by the project risk and complexity of the Web application to be built. The techniques that can be used to produce the resulting description are natural language, templates, use cases, formal languages or prototypes. For a detailed analysis of such techniques for the Web development see Escalona & Koch (2004). Informal descriptions such as user stories, and semi-formal descriptions like templates and use cases, are particularly suited to describe how users intend to perceive their interaction with a Web system.

Use cases are further refined using for this purpose formatted specifications or workflows. Both representations usually include actors, pre- und post-conditions, workflow descriptions, exceptions and error situations, variations, information sources needed, produced results, references to other documents, and interdependencies with other models. In particular, in the development of Web systems the informational, navigational and process goals have to be gathered and specified. Informational goals indicate the need of content to be provided to the Web system user. Navigational goals point toward the kind of access to this content. Process goals specify the ability of the user to perform some tasks within the Web system (Pressman, 2005).

2.2 Comparing Current Approaches

Our preliminary survey (Escalona & Koch, 2004) gives an overview about techniques and notations for Web requirements provided by Web methodologies. This comparative study shows that NDT (Escalona, 2004), OOHDM (Rossi & Schwabe, 1998), UWE (Koch& Kraus, 2002) and W2000 (Baresi et al., 2003) are the Web methodologies that pay special attention to requirements. Other approaches analyzed in the survey either propose the use of classical techniques to deal with Web requirements or ignore this phase of the development process.

The selected approaches recognize the relevance of the separation of concerns in the early requirements phase. In order to illustrate the characteristics, similarities and differences of these methods, we model the requirements of the same example Web system with each of the four methodologies.

The running example is a simplified CD e-shop, whose functionality is restricted to (1) the registration of users at the CD e-shop, (2) login, (3) search of CDs, (4) add to the shopping cart, and (5) checkout for buying the CDs. The approaches NDT, OOHDM, UWE and W2000 start the modeling process by identifying actors and use cases, and build in the next step a use case model with them. Fig. 1 depicts the use case model for the simplified e-shop example.

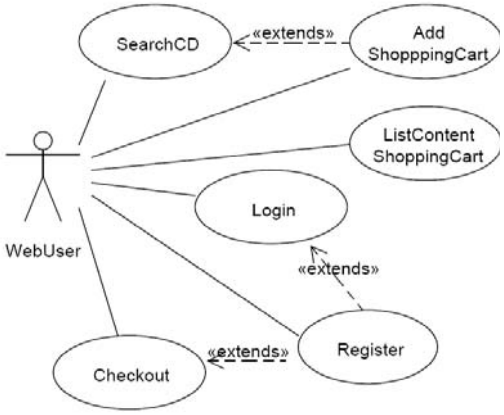


Fig. 1. UML use case diagram for the *CD e-shop*

Further modeling results produced in the requirements phase by these methodologies differ from each other and are shown in the following.

Navigational Development Techniques (NDT) is a methodology that mainly focuses on requirements and on the analysis phase. NDT (Escalona, 2004) uses several techniques to deal with requirements; basically, it proposes to use uses cases and provides formatted templates to describe requirements.

NDT classifies requirements in *storage information*, *actor*, *functional*, *interaction* and *non-functional requirements*. For each type, NDT defines a special template, i.e. a table with specific fields that are completed by the development team during the phase of requirements elicitation. Each template is assigned an identifier. The structured requirements specification performed by NDT allows the generation of the analysis models of the Web system from this specification. In this sense, NDT is a model-driven proposal. The complete life cycle of NDT is supported by its associated tool, named NDT-Tool (Escalona at al., 2003).

Concretely, for the CD e-shop example, NDT specifies several storage information requirements. A storage information requirement expresses all the information that has to be stored for a concrete application concept. For instance, a template for the registered user's information is identified by SR-01, another for the CD information by SR-02, etc. Table 1 shows the most relevant fields of the template for the requirements SR-01.

Each use case is also described by a functional template in NDT. Table 2 shows an example of such a template for the use case *Login*.

Table 1. Template for storage of information requirements (NDT)

SR-01	WebUser	
Description	The system manages information about users	
Specific data	Name & description	Nature
	name: contains *user's name*	String
	address: this field stores the user's postal address	String
	userID: is the user's identification to access the e-shop	String
	password: is the user's password to access the e-shop	String

The process starts when the system asks for the userID and the password, and for the "remember field". Remember has the value "true" if the application should remember the user identification and the password of the user, otherwise it has the value "false". In addition, NDT provides a template for actors, i.e. a template to describe the role the Web system user will play. Such a template is identified by e.g. identifier AC-01.

Finally, NDT also designs specific templates for interaction requirements. In this example, an interaction requirement for the CD information will be developed.

Table 2. Template for functional requirements (NDT)

FR-01	Login	
Description	Authentication to allow access to the checkout process	
Actors	**Use case actor**	
	AC-01. *WebUser*	
Normal sequence	**Step**	**Action**
	1	The system asks for the *userID* and *password* and the option to *remember* both *userID* and *password*
	2	The user puts the *userID* and the *password*
	3	The *userID* and the *password are checked*
	4	The *userID* and the *password* is stored if the field *remember* is true
	5	Access to checkout is allowed
Exceptions	**Step**	**Action**
	4	The user is not registered, so the user executes FR-02
	4	The *userID* or the *password* are not valid, continue with step 1

Object-Oriented Hypermedia Design Method (OOHDM) supports separation of concerns by developing separated conceptual, navigational and abstract interface models of Web systems. The navigation model is built with a variety of concepts, among others the powerful *navigation context*. The first versions of OOHDM (Schwabe & Rossi, 1998) did not cover the requirements phase focusing instead on design and implementation.

OOHDM was extended afterwards with use cases and a special technique to deal with user interaction in the requirements phase. The technique used is called *User Interaction Diagram* (UID) (Vilain et al., 2000). A UID is built for each special interaction of the Web user with the Web system.

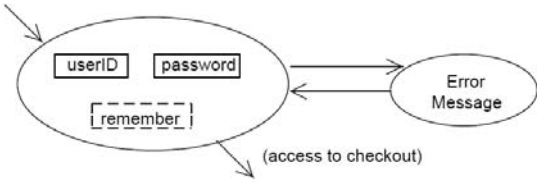


Fig. 2. UID for the *Login* use case (OOHDM)

A UID models interactions, information that require input from the user, and choices that allow changes between interactions. Each choice can be a single one or provoke the execution of a special operation. UIDs have a special notation, not based on standards. In Fig. 2, the UID for the use case *Login* is presented. The use case starts with the initial interaction where the userID and the password have to be entered by the user. In contrast Remember data is optional. After the user has entered the data, either the user will be able to checkout or (if userID or password is not correct) a new interaction will occur. In our example an error message is presented.

UML-based Web Engineering (UWE) is a model-driven software engineering approach for the Web domain. UWE provides a UML-based notation, a methodology and a tool environment for the systematic development of Web applications (Koch & Kraus, 2002). The systematic design follows the principle of separation of concerns, which is the intrinsic characteristic of the Web domain. Thus, UWE models a Web application from different points of view: the content, the navigation structure, the business processes, the presentation and the adaptive aspects. UWE provides semi-automatic transformations, for example from content to navigation structure models.

The UML compliance of UWE allows for the use of all CASE tools, which support the Unified Modeling Language. In addition, an open source plug-in – called ArgoUWE – for the open source tool ArgoUML (www.argouml.org) has been implemented supporting the systematic transformation techniques of UWE.

UWE models requirements with UML use case diagrams and UML activity diagrams. Use case diagrams are used to represent an overview of the functional requirements while activity diagrams provide a more detailed view. In UWE, the requirements process starts with the modeling of use cases using a stereotype for navigational use cases. After that, UWE recommends to develop an activity diagram for each process use case. In Fig. 3, the activity diagram for the *Login* use case is presented.

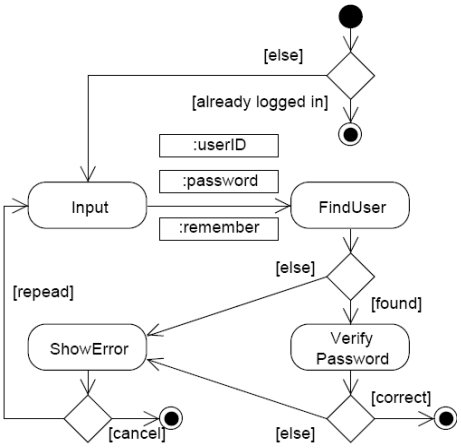


Fig. 3. Activity diagram for the *Login* use case (UWE)

Finally, **W2000** is an object-oriented approach derived from HDM (Baresi et al., 2001) that supports separation of concerns during the development process. W2000 extends the UML notation to model hypermedia applications. The requirements analysis in W2000 is divided into two sub-activities: functional and navigational requirements analysis. Every actor identified during the requirements elicitation phase has his own navigation and functional requirements model. W2000 thus proposes to develop two different types of use case diagrams. The first one includes the functional use cases. In our running example these use cases are Login, Register, AddToShoppingCart and Checkout. The second one, named the *navigation use case diagram*, represents the navigation possibilities of each actor. These are the use cases SearchCD and ListContentShoppingCart for the e-shop example.

3 Metamodel for Web Requirements

After consideration of the different proposals we concluded that they address many similar concepts, however not always using the same terminology. Each methodology has also its strengths and weaknesses. NDT proposes a detailed specification of requirements from the outset of a project but the templates are not easy to complete as they require intensive interviews. Conversely, visual representations like those proposed by UWE, W2000 or OOHDM are more intuitive for a first blueprint. But graphical notations are usually too abstract for the next phases (Insfrán et al., 2002). Modeling with UIDs faces the additional difficulty that CASE tools cannot be used due to the UIDs proprietary notation.

The modeling concepts we present for the Web requirements specification are defined based on the similarities of the methods that were analyzed. They are represented as UML metaclasses and constitute our metamodel for Web requirements

engineering (WebRE), which is depicted in Fig. 4. The metaclasses represent the concepts without any information about its representation. They are grouped in two packages, following the structure of the UML metamodel: the WebRE structure and the WebRE behavior package.

The **behavior** package consists of the metaclasses Navigation, WebProcess, WebUser, Browse, Search and UserTransaction. Functionality of a Web system is modeled by a set of instances of two kinds of specific use cases: navigation and process use cases and specific activities, such as browse, search and user transactions. A **Navigation** use case comprises a set of browse activities that the WebUser will perform to reach a target node. A browse activity is the action of following a link and is represented by the metaclass **Browse**. A browse activity can be enriched by search actions, which is represented by a **Search** metaclass. A Search has a set of parameters, which let define queries on the content. The results are shown in the target node.

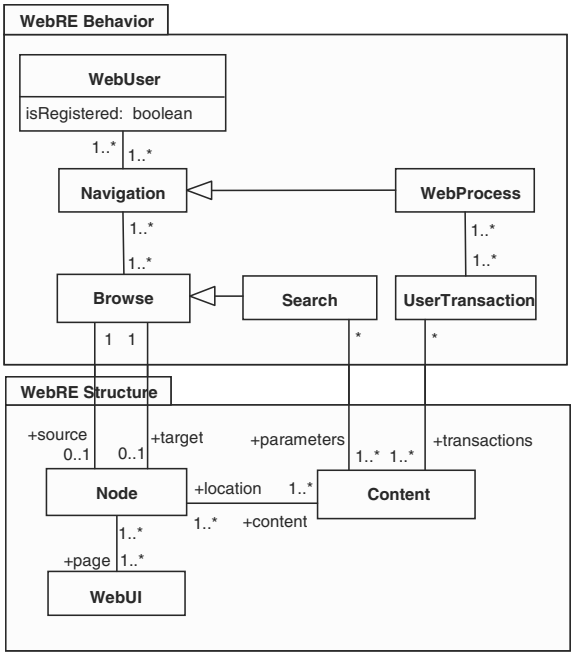


Fig. 4. Metamodel for Web Requirements Engineering (WebRE)

More complex activities are expressed in terms of transactions initiated by the user, like checkout in an e-shop or an online reservation. Such actions, which imply a transaction operation, are modeled by a metaclass **UserTransaction** in the behavior package. The second kind of use case is the **WebProcess,** which is refined by activities of type browse, search, and at least one user transaction.

A **WebUser** is any user who interacts with a Web System. Examples of instances of WebUser are RegisteredUser, Non-Registered User and System Administrator.

The second package of the metamodel is the **structure** package, which contains the metaclasses used to describe the structural elements of a Web system: content, node and Web user interface.

A **Node** is a point of the navigation where the user finds information. Each instance of a browse activity starts in a node (source) and finishes in another one (target). Nodes are presented to the user as pages. Note that a node can be associated to one or more pages, and a page may be associated to one or more nodes (e.g. asynchronous communication). The concept of page is represented by the **WebUI** metaclass. Besides, each node can show different pieces of information. Each piece of information of a Web system is represented as a metaclass **Content**.

The metamodel can be specified in more detail including invariants. For instance, a search activity has associated a node as source, which is the location of the parameters that will be used for the query of the search. Such an invariant can be formally expressed as an OCL constraint as follows:

```
context Browse
inv: self.oclIsKindOf(Search) implies
     self.parameters -> forAll
     (p | p.location -> includes(self.source))
```

Table 3 shows the mappings from the metaclasses of both packages WebRE Behavior and WebRE Structure to the modeling elements of the methods NDT, OOHDM, UWE and W2000. The shadowed cells express that the method of the corresponding row does not provide a modeling element that supports the metamodel concept of the first column.

Table 3. Mapping metamodel concepts to elements of Web methodologies

WebRE Concept		NDT	OOHDM	UWE	W2000
Behavior	WebUser	Actor	Actor	Actor	Actor
	Navitation	Visualization prototype	Use case	Navigation use case	Browse use case
	WebProcess	Use case	Use case	Use case	Use case
	Browse	Visualization prototype	Single choice	Activity	
	Search	Phrase	Optional data entry	Activity	
	UserTransaction	Functional requirement	Application processing	Activity	
Structure	Node	Visualization prototype			
	Content	Storage requirement	Data entry	Class	
	WebUI		Interaction		

In NDT, WebUsers are defined with the template AC used to define actors of a Web system. The concepts of Navigation, Browse and Node are modeled as interaction requirements in a template named visualization prototypes (VP). A Search action is modeled with phrases, which are written in BNL (bounded natural language) in order to select a set of content instances to be presented to a WebUser. WebProcesses are treated with use cases and the UserTransaction activities are modeled with the

functional requirements template (FR). Finally, the Content concept is described by the storage information requirement (SR). NDT does not contain any modeling element that covers WebUIs from the metamodel.

OOHDM uses use cases and actors to represent WebUser, Navigation and Web-Process. In addition, OOHDM provides UID elements to model in the requirements phase activities and structural elements with exception of Node. The Browse activities are represented in OOHDM with single choices, the Search activities with optional data entries and UserTransaction activities with application processing. Content is represented by data entries and WebUI with interactions.

UWE uses the UML behavioral elements use case and activity and the structural element class to model the concepts defined in the Web requirements metamodel. From the structural elements UWE only supports the content concept in requirements modeling. UWE extends the UML using the extension mechanism provided by the UML to define the modeling element Navigation use case, which is defined to represent the typical browsing interaction of Web users with Web systems. For the more detailed description of the Web user-Web system interactions activity diagrams are used without specific modeling elements that distinguish between Browse, Search and UserTransaction activities. Finally, classes in object flows associated to the activity diagrams model Content.

W2000 restricts the support to modeling elements actor and both types of use cases. In fact, it only provides elements for modeling in the large, i.e. building a UML use case model. The use case model contains actors, general use cases and specific browse use cases. W2000 recommends depicting two separated use case diagrams: one for general use cases and another for use cases of type browse, thus separating the navigation and process concerns.

4 Towards a Common Notation

A metamodel provides a basis for the definition of a notation and the development of tools.

The objective is to define on the one hand a notation for the concepts included in the metamodel for Web requirements that allow for intuitive and expressive specification of the requirements of Web applications.

On the other hand a domain specific modeling language requires tool support for their use in the development of Web systems. Limited impact can be achieved by proprietary notation and prototypes. Instead wide dissemination is achieved by providing plug-ins or extensions of already in use CASE tools, such as those for the UML. Therefore we define the modeling language for Web requirements as an extension of UML using the extensions mechanisms provided by the UML – a so-called UML profile.

The UML profile for Web requirements engineering specifies how the concepts of the WebRE metamodel relate to and are represented in standard UML using stereotypes and constraints.

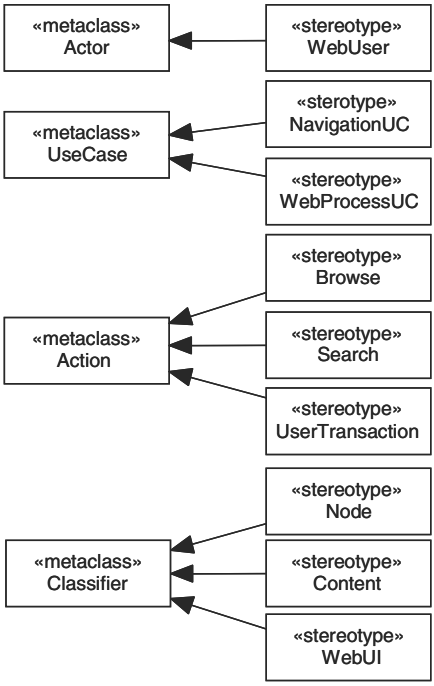
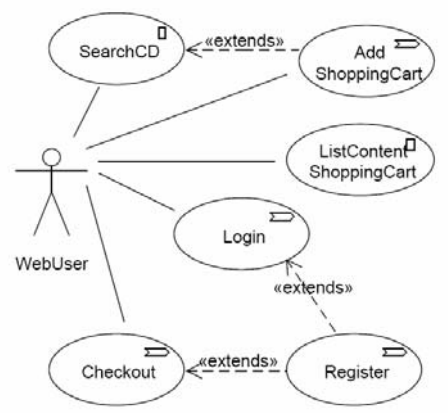


Fig. 5. Modeling elements of UML profile for Web requirements engineering (WebRE)

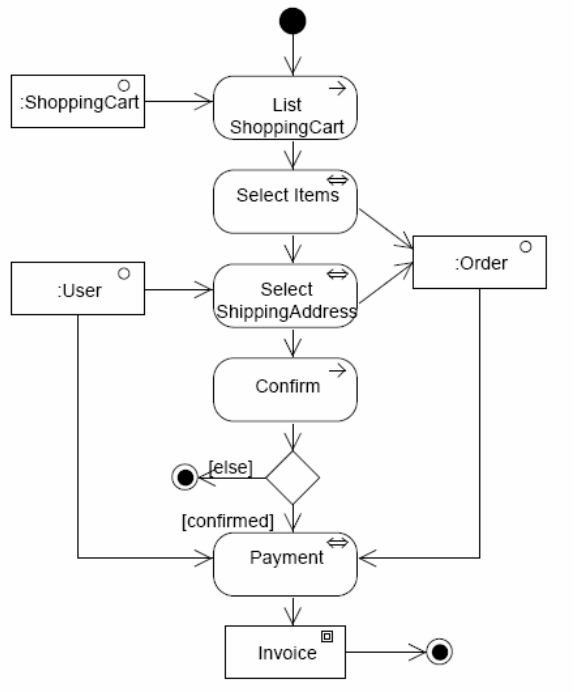
Table 4. Icons for stereotypes of the WebRE profile

WebRE Behavior	Navigation	□
	WebProcess	Σ
	Browse	⇒
	Search	?
	UserTransaction	⇔
WebRE Structure	Node	□
	Content	O
	WebUI	□

Fig. 5 shows the graphical representation of the UML profile showing how the stereotypes defined for each class of the metamodel extend a UML metaclass (OMG-UML 2.0, 2005). Table 4 shows the icons proposed for stereotypes of the WebRE profile. We use the common language provided by the profile to depict the use case diagram of the CD e-shop example presented in Sect. 2 (Fig. 6a). The model of the Checkout process is shown in Fig. 6b.



(a) Use case model



(b) UML activity diagram

Fig. 6. CD e-shop example using the UML Profile notation for Web requirements

5 Conclusions and Future Work

In order to reinforce requirements engineering in Web methodologies we present a metamodel for Web requirements (WebRE). The metamodel provides key concepts

for the requirements specification in the Web domain, such as specific use cases: navigation use case and Web process use case; specific activities such as browse, search and user transaction; and structural elements such as content, node and user interface of Web systems. We define a common modeling language – a so-called UML profile – to express these Web requirements concepts. A modeling language with Web specific constructs has the advantage of producing compact but semantically rich domain specific models. The additional advantage of a UML profile is the tool support given by UML generic CASE tools.

The disadvantage of such a common modeling language is the high probability that Web methodologies that already cover requirements engineering tasks will not replace the own notation and techniques in use by now. In contrast, methods that do not address requirements specification, can easily integrate the presented approach. However, we show that a mapping between elements of the metamodel and the modeling elements of the methodologies of the first group is possible.

A consensus would offer therefore the application of model transformations based on the model-driven development (MDD) principles. For example, the development of a Web system could be started using a graphical notation like activity diagrams proposed by UWE or UIDs of OOHDM, which are more intuitive to provide an overview of the Web system to be built. Afterwards, the visual models are transformed into a set of NDT formatted specifications, in order, for instance, to allow further modeling of details needed in next phase of the development process.

Subject to future work will be the specification of relations and transformations among the elements of the metamodel of Web requirements and the modeling elements of the different methodologies. For the specification we will use QVT (OMG-QVT, 2005), which is an OMG standard for model-to-model transformations.

For tool support, we plan to integrate transformation facilities among NDT and UWE or NDT and the modeling language defined in this paper for Web requirements (WebRE) into the NDT-Tool.

References

Baresi L., Garzotto F., Paolini P.: Extending UML for Modelling Web Applications. In: Annual Hawaii Int. Conf. on System Sciences, pp. 1285–1294. Miami, USA (2001)

Escalona, M.J., Torres, J., Mejías, M., Reina, A.M.: NDT-Tool: A CASE Tool to deal with Requirements in Web Information Systems. In: Lovelle, J.M.C., Rodríguez, B.M.G., Gayo, J.E.L., Ruiz, M.d.P.P., Aguilar, L.J. (eds.) ICWE 2003. LNCS, vol. 2722, pp. 212–213. Springer, Heidelberg (2003)

Escalona, M.J.: Modelos y Técnicas para la Especificación y el Análisis de la Navegación en Sistemas Software. Ph. Thesis University of Seville (2004)

Escalona, M.J., Koch, N.: Requirements Engineering for Web Applications: A Comparative Study. Journal on Web Engineering 2(3), 193–212 (2004)

Insfrán, E., Pastor, O., Wieringa, R.: Requirements Engineering-Based Conceptual Modelling. Requirements Engineering Journal 7(1) (2002)

Kappel, G., Pröll, B., Reich, S., Retschizegger, W.: Web Engineering, dpunkt Verlag (2003)

Koch, N., Kraus, A.: The expressive Power of UML-based Web Engineering. In: Second Int. Workshop on Web-oriented Software Technology (IWWOST02), pp. 105–119. Málaga, Spain (2002)

Lowe D., Eklund J. Client Needs and the Design Process in Web Projects. Journal on Web Engineering 1(1), 23–36 (2002)
Lowe, D., Hall, W.: Hypermedia and the Web. An Engineering Approach. John Wiley & Son, Chichester (1999)
OMG, MOF 2.0 Query/Views/ Transformations Final Adopted Specification, Object Management Group (2005), http://www.omg.org/cgi-bin/apps/doc?ad/05-11-01.pdf
Pressman, R.: Software Engineering: A Practitioner's Approach. McGraw Hill, New York (2005)
Schwabe, D., Rossi, G.: An Object Oriented Approach to Web-Based Application Design. In: Theory and Practice of Object Systems, vol. 4(4), Wiley and Sons, New York, USA (1998)
Sommerville, I., Ransom, J.: An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement. ACM TOSEM 14(1), 85–117 (2005)
Vilain, P., Schwabe, D., Sieckenius de Souza, C.: A diagrammatic Tool for Representing User Interaction. In: Evans, A., Kent, S., Selic, B. (eds.) UML 2000. LNCS, vol. 1939, pp. 133–147. Springer, Heidelberg (2000)
Wieringa, R.: Requirement Engineering: Problem Analysis and Solution Specification. In: Koch, N., Fraternali, P., Wirsing, M. (eds.) ICWE 2004. LNCS, vol. 3140, pp. 13–16. Springer, Heidelberg (2004)