# A Survey of Requirements Specification in Model-Driven Development of Web Applications

PEDRO VALDERAS and VICENTE PELECHANO, Universitat Politècnica de València

Model-driven development has become more and more important in the last few years. In the context of web application development, many web Engineering methods that propose model-driven development processes have appeared. However, earlier stages of these processes are seldom considered and few of these methods rigorously face the problems of specifying web application requirements and translating them into the proper conceptual model. However, it is widely recognized that requirements engineering activities are essential to obtain quality software products.

This article surveys Model-driven web engineering methods in a comparative study and analyzes the techniques proposed for specifying functional, data and navigational requirements as well as the mechanisms provided for automatically translating these requirements into conceptual models. Our main goal is to provide a critical view of the support that is provided by these methods for handling web application requirements in order to show their current limitations and strengths.

Categories and Subject Descriptors: D.2.1 [**Software Engineering**]: Requirements/Specifications—*Elicitation methods; methodologies; tools*

General Terms: Design, Management

Additional Key Words and Phrases: Requirements engineering, model-driven development, web engineering, web applications, survey

**ACM Reference Format:**
Valderas, P., and Pelechano, V. 2011. A survey of requirements specification in model-driven development of web applications. ACM Trans. Web 5, 2, Article 10 (May 2011), 51 pages.
DOI = 10.1145/1961659.1961664 http://doi.acm.org/10.1145/1961659.1961664

## 1. INTRODUCTION

Model-Driven Development (MDD) [Mellor et al. 2003] claims that software systems must be developed through the use of models. MDD processes usually start to develop a software system with a requirements phase in which a requirements model is defined to describe the user's needs in a computation-independent way. Then, this model is refined into one or more conceptual models that describe the system without considering technological aspects. These conceptual models are focused to be used in analysis phases. Finally, these models are (1) either refined into design models that describe the system by using concepts of a specific technology and are then translated into code, or (2) directly derived to code if they contain enough information to implement the software product in a precise and complete way.

This work has been developed with the support of MICINN under the project everyWhere TIN2010-18011 and cofinanced with ERDF.
Authors' address: P. Valderas and V. Pelechano, Research Center on Software Production Methods, Universitat Politècnica de València, 46022, Valencia, Spain; email: {pvalderas, pele}@pros.upv.es.

DOI 10.1145/1961659.1961664 http://doi.acm.org/10.1145/1961659.1961664

Currently, different factors allow us can consider MDD to be a reality. On the one hand, a myriad of methods have appeared in the recent years that propose either models based on the Unified Modeling Language (UML) or Domain-Specific Modeling Languages (DSL) to construct software from models. In addition, the emergence of MDD methodologies is encouraged by the Model-Driven Architecture (MDA) from the Object Management Group [OMG 2005]. New technologies for supporting MDD are also being developed. There are technologies such as Ecore, the Eclipse Modeling Framework (EMF) and Graphical Modeling Framework (GMF), both associated to the Eclipse Modeling Project,[1] or the DSL tools, which are integrated into MS Visual Studio, that allow us to create CASE MDD tools. There are also technologies such as ATL (ATLAS Transformation Language), XPAND, QVT (Query/View/Transformation), or MOFScript, which allow us to specify and execute model transformations. All these technologies underscore the fact that MDD can be put into practice.

From the perspective of web application development, current web engineering methods are not oblivious to these MDD trends. There are methods such as OOHDM [Schwabe et al. 1996], UWE [Koch 2000], WSDM [De Troyer and Leune 1998], OOWS [Fons et al. 2003a] or WebML [Ceri et al. 2003] that propose developing web applications by means of models. All these methods introduce conceptual models that allow us to describe web applications in a technology- independent way. Additionally, most of these methods propose tool-supported strategies that allow us to automatically or systematically generate code that is equivalent to the abstractions created at the conceptual level. These methods have been successfully applied in the development of several web applications [Acerbis et al. 2007; Fons et al. 2003b; Schwabe and De Almeida Pontes 1998], which is the proof that the implementation of web applications through the construction of conceptual models and their later refinement into code is possible. However, the specification of web application requirements is faced with less attention. In particular, little support is provided (1) to construct a requirements model to properly capture the user needs that must be supported by web applications and (2) to derive the conceptual model that satisfies the specified requirements.

This work presents a detailed study about the existing support to capture requirements in the context of model-driven web engineering methods [MDWE 2007]. Requirements are usually classified into early requirements, which refer to organizational aspects [Preciado et al. 2005; Schwinger and Koch 2006] and late requirements, which refer to the expected functions of a software system. In this work, we focus on late requirements. Within late requirements, we can find requirements related to different aspects such as functionality, data, navigation, adaptability, interface, and so on. In this work, we focus on analyzing functionality, data and navigation requirements since those are the ones that are mainly considered in the context of MDWE. Although also important, the other types of requirements are planned to be analyzed in further studies.

Thus, we analyze nine MDWE methods with regard to two main aspects: (1) techniques proposed to specify functional, data and navigational requirements and (2) mechanisms provided to support the treatment of these requirements within a MDD context. We also demonstrate the applicability of the proposed RE techniques with respect to a given web application example. The main objective of this analysis is to provide a critical view of the support that is provided by MDWE methods for handling web application requirements in order to throw some light on their current limitations and strengths.

The rest of this article is organized as follows. Section 2 presents a chronological overview of the MDWE methods that support the development of web applications.

[1]http://www.eclipse.org/proposals/modeling/

This section also indicates which methods have been selected to be studied in detail and the reasons for this selection. Section 3 introduces the criteria used to evaluate each method as well as a running example. Section 4 presents the study about how MDWE methods support requirements specification activities. Section 5 presents a report on lessons learned, summarizing the strengths and limitations of the methods. Section 6 compares this survey with other existing ones. Finally, conclusions are commented on in Section 7.

## 2. MDWE METHODS: A CHRONOLOGICAL OVERVIEW

Several MDWE methods for developing web applications have been presented throughout the published literature. Some of these methods are focused on the definition of new techniques for modeling and designing web applications and, although some of them consider requirements in an indirect way, they do not introduce an explicit requirements phase in their development processes. They are the following: HDM (Hypermedia Design Method) [Garzotto et al. 1993], The MacWeb Hypermedia Development Environment [Nanard and Nanard 1995], EORM (Enhanced Object-Relationship Model) [Lange 1996], the Araneus Project [Mecca et al. 1999], WebComposition [Gellersen and Gaedke 1999] and WUML [Kappel et al. 2001].

There are other methods that acknowledge the need of requirements activities and include a requirements stage in their development processes. However, different support is provided to specify web requirements within the proposed requirements activities. On the one hand, there are some approaches that either propose the use of existing requirements specification techniques defined for traditional software (non-Web) development or do not propose any concrete technique. These approaches are the following: RMM (Relationship Management Methodology) [Isakowitz et al. 1995], the OO/Pattern Approach [Thomson et al. 1998], HFPM (Hypermedia Flexible Process Modeling) [Olsina 1998], RNA (Relationship Navigation Analysis) [Yoo and Bieber 2001], Hera [Frasincar et al. 2002], Conallen's approach [Conallen 1999], the Design-Driven Requirements Elicitation approach [Lowe and Eklund 2002] and the GX web Engineering Method [Souer et al. 2007]. On the other hand, there are other approaches that propose new proprietary techniques or extend traditional ones to specify web applications requirements. They are: OOHDM, WSDM, UWE, SOHDM, WebML, OOH, OOWS, W2000 and NDT. This survey is focused on the analysis of these methods since they propose new techniques specifically created for specifying web requirements. The goal of this analysis is to provide a precise view of the efforts that are currently being done in order to improve the specification of web requirements. Before presenting this analysis, we introduce the criteria used to study them.

Figure 1 shows an overview of the methods introduced in this article. They are chronologically ordered by the year of the first publication that we know.

## 3. ANALYSIS SET UP

In this section, we explain that aspects have been considered to analyze each method and why. We also introduce the running example that has been used to put into practice the requirements techniques proposed by each method.

### 3.1 Evaluation Criteria

The objective of this survey is to provide a critical analysis of how requirements specification activities are supported in the context of MDD of web applications. Then, we evaluate the requirements models that have been proposed by ten web engineering methods from two perspectives: the requirements specification perspective and

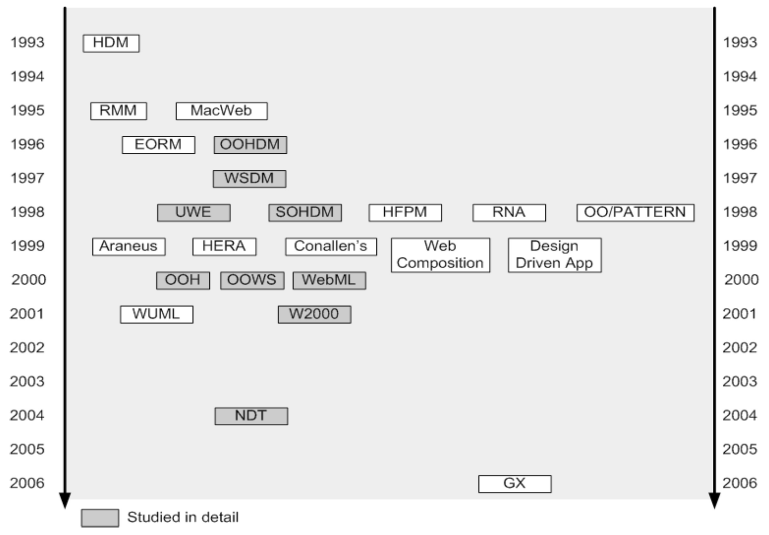


Fig. 1. Chronological overview of MDWE methods.

the model-driven perspective. The evaluation criteria used for each perspective are explained in the following sections.

*3.1.1 Evaluation of the Requirements Specification Perspective.* In the evaluation of the requirements specification perspective, we study how the functional, data and navigational requirements are specified by each method. In order to choose the types of requirements to be studied, we performed an initial review of the published literature. From this review, we checked that most of the published papers about web requirements in MDWE methods focus on the capture of these kinds of requirements and other types of requirements such as adaptation, scalability or performance are little considered. Thus, we consider the capture of functional, data and navigational requirements to be one of the most important challenges that MDWE methods are currently facing, and then we focus this survey on these types of requirements.

A definition of each type of requirement and the criteria to evaluate its specification are presented in the following.

(1) *Functional requirements (FR)* describe fundamental actions that must take place in the software [IEEE 1998]. The evaluation criteria used for analyzing the specification of functional requirements are defined from the different aspects proposed by the IEEE standard 830-1998 [IEEE 1998] to be included in a functional requirements specification. We evaluate if the functional requirements specification includes:
    - validity checks on the inputs (FR1)
    - exact sequence of operations (FR2)
    - responses to abnormal situations (FR3)
    - effect of parameters (FR4)
    - relationship of outputs to inputs (FR5)

(2) *Data requirements (DR)* describe the data that must be specified within a system [Ashworth 1989]. If these data are stored in a database, these requirements are also known as logical database requirements [IEEE 1998]. The evaluation criteria used for analyzing the specification of data requirements are defined from the properties that the IEEE Standard 830-1998 proposes to include in a data requirements

specification [IEEE 1998]. We evaluate if the data requirements specification includes:
— types of information (e.g., text, images, numerical data, etc.) to be stored (DR1)
— frequency of use (DR2)
— accessing capabilities of users and system operations (DR3)
— data entities and their relationships (DR4)
— integrity constraints (DR5)
— data retention requirements (how long should data be stored) (DR6)

(3) *Navigational requirements (NR)* describe the users' navigation needs through the hyperspace [Escalona et al. 2004]. These requirements have recently gained importance in web application development because navigation, which is an aspect that has been little considered in traditional software development, has become critical for web applications [Burdman 1999; England and Finney 1999; Lowe 2003; Overmyer 2000].
Navigation is a critical aspect in web applications because the focus of web applications is extensively based on publishing information [Greenspun 1999]. In accordance with the hypertext paradigm [Conklin 1987], this information is organized in a structure made up of nodes, links, and anchors, which allows users to navigate throughout the information in a nonlinear way. Furthermore, this navigational structure is complemented with mechanisms that allow users to filter the accessed information according to specific criteria. However, not only information is provided by web applications. Functionality can also be provided to users. Access to this functionality also is published throughout the navigational structure. Therefore, the evaluation criteria used for analyzing the specification of navigational requirements consist in analyzing the mechanisms proposed to describe:
— published information (NR1)
— different possibilities of navigating this information (NR2)
— mechanisms to filter this information (NR3)
— functionality published together with the information (NR4)

We evaluate the specification of each type of requirement by analyzing the following two aspects.

— *Explicit Support and Techniques*. First, we analyze if a type of requirements is explicitly supported by the method. We consider a type of requirement to be explicitly supported if its specification has been considered by authors in some published work. If a type of requirement is explicitly supported, we indicate the techniques proposed by the authors to specify them.
— *Support Provided to Each Criterion*. We evaluate how the criteria associated to each type of requirement are supported. One of the following two evaluations is done.
  (1) If a type of requirement is explicitly considered by the method, we analyze whether or not the techniques proposed by the authors include mechanisms to support the criteria introduced above. Each criterion is marked as fully supported, partially supported, or not supported according to the support provided by the techniques.
  (2) If a type of requirement is *not* explicitly considered, we present our analysis about how we think that criteria could be supported (if they can) by the techniques proposed by the authors, although they were not defined to support this type of requirement. It is worth noting that this analysis has been tested only with the running example. We cannot guarantee that it could be satisfied in other applications. Thus, the criteria are be marked only as partially supported

or not supported in order to indicate whether or not some support could be provided.

*3.1.2 Evaluation of the Model-Driven Perspective.* MDD proposes the development of software by using models. To make this vision a reality a key premise to be achieved is automation [Mellor et al. 2003]. If models cannot be automatically transformed, they end up being used as merely documentation, and then they are of limited value [Selic 2003].

In order to transform models, we must define mapping functions [Mellor et al. 2003]. These mapping functions should take a specific model as source, analyze its defined elements and transform them into code or elements of another model. To do this, it is crucial that the meaning of each model element be precisely defined. This meaning is obtained by constructing a precise metamodel [Atkinson and Kuhne 2003], which must define the meaning of every element that can be created in a model.

On the other hand, Atkinson and Kuhne [2003] discuss that visual modeling is needed for defining a proper model-driven architecture. This aspect helps people to understand models and improve problems related to either the interpretation of information described by personnel that have left the organization or the validation of requirements by customers.

Finally, it is clear that automation in MDD cannot be achieved if the proper tools are not developed. Thus, we need to provide two types of tools: (1) tools that allow us to create and properly store models, and (2) tools that take these models and transform them.

The evaluation criteria used for analyzing the support provided for handling requirements in MDD environments consist in studying if

— the requirements model proposed by each MDD web engineering method fits a precise metamodel (MR1);
— methods define mapping functions to transform requirements models into conceptual models that satisfy the specified requirements (MR2). In terms of the MDA, this is known as CIM-to-PIM transformations;
— tools for automatically applying mappings are implemented (MR3);
— a concrete visual syntax is defined to specify requirements (MR4);
— Tools for creating the proposed requirements model are developed (MR5).

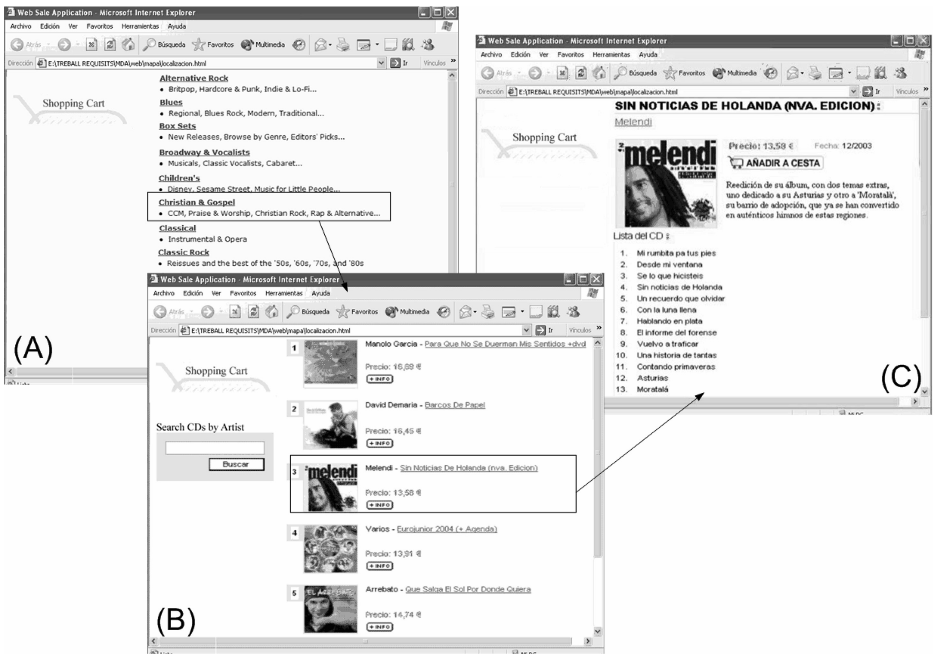
In this perspective, each criterion is evaluated as fully supported, partially supported, or not supported.

## 3.2 The Running Example

In order to exemplify the requirements specification techniques proposed by each method we propose an E-commerce application as running example. We propose an example based on an E-commerce application because applications of this type are focused on both providing a great amount of information to users and providing them with functionality in order to allow the purchase of products. These aspects allow us to properly show how the three types of requirements considered in this survey (functional, data and navigational) are supported. Note that this example has been chosen to analyze strengths and weaknesses of MDWE methods in relation to the specification of these types of requirements; other type of examples could emphasize other strengths and weaknesses that have not been identified in this work.

In a nutshell, the requirements of the proposed E-commerce Application are described as follows:

The E-commerce application must support the online purchase of CD, software products and books. This E-commerce application must allow users, who are initially


Web Sale Application - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Multimedia
Dirección E:\TREBALL REQUISITS\MDA\web\mapa\localizacion.html
Ir Vínculos
Shopping Cart
(A)
(B)
(C)
Buscar
+ INFO
AÑADIR A CESTA

**Alternative Rock**
- Britpop, Hardcore & Punk, Indie & Lo-Fi...

**Blues**
- Regional, Blues Rock, Modern, Traditional...

**Box Sets**
- New Releases, Browse by Genre, Editors' Picks...

**Broadway & Vocalists**
- Musicals, Classic Vocalists, Cabaret...

**Children's**
- Disney, Sesame Street, Music for Little People...

**Christian & Gospel**
- CCM, Praise & Worship, Christian Rock, Rap & Alternative...

**Classical**
- Instrumental & Opera

**Classic Rock**
- Reissues and the best of the '50s, '60s, '70s, and '80s

Search CDs by Artist

1 Manolo Garcia - Para Que No Se Duerman Mis Sentidos +dvd
Precio: 16,69 €

2 David Demaria - Barcos De Papel
Precio: 16,45 €

3 Melendi - Sin Noticias De Holanda (nva. Edicion)
Precio: 13,58 €

4 Varios - Eurojunior 2004 (+ Agenda)
Precio: 13,91 €

5 Arrebato - Que Salga El Sol Por Donde Quiera
Precio: 14,74 €

**SIN NOTICIAS DE HOLANDA (NVA. EDICION) :**
Melendi

Precio: 13,58 € Fecha: 12/2003

Reedición de su álbum, con dos temas extras, uno dedicado a su Asturias y otro a 'Moratalà', su barrio de adopción, que ya se han convertido en auténticos himnos de estas regiones.

Lista del CD :
1. Mi rumbita pa tus pies
2. Desde mi ventana
3. Se lo que hicisteis
4. Sin noticias de Holanda
5. Un recuerdo que olvidar
6. Con la luna llena
7. Hablando en plata
8. El informe del forense
9. Vuelvo a traficar
10. Una historia de tantas
11. Contando primaveras
12. Asturias
13. Moratalá

Fig. 2. Running example.

connected to the system as anonymous visitors, to access information about the different products that exist in the catalogue.

The E-commerce application must provide users with a shopping cart in which they can add the products to be purchased. The shopping cart must always be available to users in order to allow them to inspect it. When a product is added to the shopping cart, the system must automatically update the product stock.

Once users have selected the products to be purchased by adding them to the shopping cart, the E-commerce application must allow them to create and send a purchase order. To do this, users must identify themselves as registered customers. Furthermore, the system must register the times that a product is purchased as well as the client profiles that usually purchase this product.

Finally, all the information about products must be managed by administrators. Administrators must also be able to manage the information of the registered customers.

Figure 2 shows a possible implementation of part of the proposed running example. Pages in this figure support the adding of CDs to the shopping cart. To do this, users initially access a list of music categories (Page A). From this list, users can select one in order to obtain the list of CDs that belong to it (Page B). From this list, users can either find the CDs of a specific artist by means of a search engine or select a CD in order to obtain a detailed description (Page C). From this description, users can add the CD to the shopping cart. Note how the shopping Cart is always available.

### 3.3 Analysis Structure

The analysis of each MDD method is presented in accordance with the following structure.

(1) *General Description*. We present an overview of the method by describing characteristics such as its authors, the year in which it appeared, and the development process that it proposes.
(2) *Requirements Model*. We present an overview of the requirement techniques that each method proposes to create a web application requirements model. We exemplify these techniques by creating a partial version of the requirements model of the running example (further presented).
(3) *Evaluation of Requirements Specification*. We evaluate requirements models according to the criteria defined for the requirements specification perspective.
(4) *Evaluation of MDD*. We evaluate requirements models according to the criteria defined for the model-driven perspective.
(5) *Main Strengths and Weaknesses*. We conclude the analysis by presenting a summary of the main strengths and weakness of the method.

## 4. STUDY OF REQUIREMENTS SPECIFICATION TECHNIQUES IN MDWE METHODS

Next, a detailed study of the requirements specification techniques proposed by nine MDWE methods is presented. The methods are introduced in chronological order in accordance with the year in which they appear in the literature.

### 4.1 OOHDM: Object Oriented Hypermedia Design Model

*4.1.1 General Description.* OOHDM was developed by Schwabe and Rossi in 1994 [Schwabe and Rossi 1994; Schwabe et al. 1996]. It was one of the first approaches to provide a methodological solution for the development of web applications. This approach takes some ideas proposed in HDM [Garzotto et al. 1993] and applies them in a well-defined development process based on the object-oriented paradigm. OOHDM emphasizes the separation of the navigational aspect from other aspects such as the conceptual aspect and the interface aspect. Other approaches have been further inspired by this idea of separation of aspects. Finally, it is worth noting that OOHDM is not a closed approach and it is continuously being extended and improved.

The development process of this approach is divided into five main phases.

— *Requirements Gathering*. In this phase, the users that must interact with the web application are identified as well as the users' needs that the web application must support. Early versions of OOHDM do not consider the requirements phase. This phase was included after defining some extensions to the method [Vilain et al. 2000a].
— *Conceptual Design*. This phase consists in the definition of a conceptual schema in which the static aspect of the system is described.
— *Navigational Design*. In this phase, a navigation class diagram and a navigation structure diagram must be defined. The first one represents the static possibilities of navigation in the system. The second one extends the navigation class diagram including access structures and navigation contexts.
— *Abstract Interface Design*. This phase consists in the description of the user interface in an abstract way. To do this, an abstract interface model is developed using a special technique named ADVs [Cowan and Lucena 1995].
— *Implementation*. In this phase, the web application is implemented. The implementation is based on the previous models.

*4.1.2 Requirements Model.* Web application requirements are handled in the requirements gathering phase. This phase is divided into five steps.

(1) *Identification of Roles*. The different user roles that are going to interact with the web application are identified.

(2) *Specification of Scenarios.* Users describe the work that each role must perform by means of scenarios. Scenarios are described in natural language.
(3) *Specification of Use Cases.* Scenarios that describe a similar task are grouped, and a use case is defined for each group of scenarios. Use cases are described by textual templates.
(4) *Specification of User Interaction Diagrams.* Use cases are refined with *user interaction diagrams* (UIDs), which allow us to consider navigational aspects at the requirements level. For each use case, a UID is defined. Each UID graphically describes the interaction between the users and the system without considering specific aspects of the user interface.
(5) *Validation of Use Cases and User Interaction Diagrams.* Both use cases and UIDs must be validated by users.

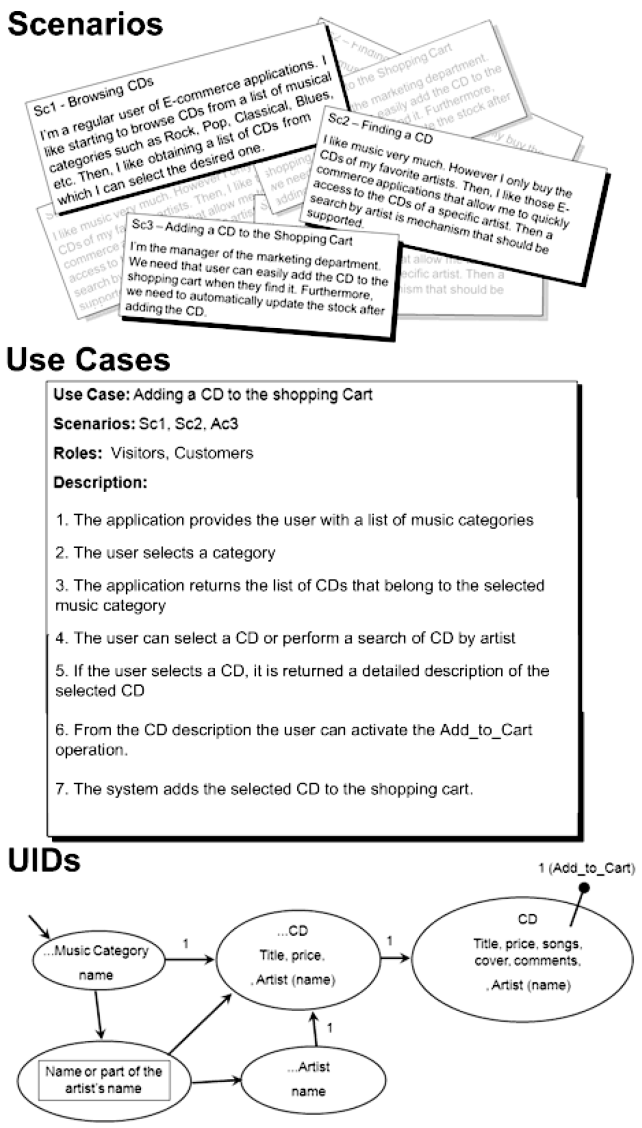
Figure 3 shows a partial view of the OOHDM requirements model that is obtained when we specify the requirements of the running example. This figure shows three examples of scenarios described by different users: Browsing CDs, Finding a CD, and Adding a CD to the shopping cart. These three scenarios describe a similar task. Then, a use case that supports this task is derived from them. The use case is defined by a name, the scenarios that it supports, the user roles that can perform it, and a description defined from actions of users and the system. Pre- and postconditions may also be defined.

The UID associated to this use case indicates that it must be performed as follows: First the system provides users with a list of music category names. From this list, users can select one and then they obtain a list of CDs. The title, the price and the artist's name is given for each CD. From the list of music categories, users can also introduce the name of an artist. From this name, a list of artists is provided. If users select an artist, they obtain a list with the artist's CDs. From the list of CDs, users can select one and then obtain a description of it. In this description, the title, the price, the songs, the cover, some comments and the artist's name of the CD are provided. Users can also add the CD to the shopping cart at this point.

*4.1.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. The technique proposed to specify them is the use case diagram. The description of actions associated to use cases allows analysts to describe the sequences of actions performed by the user and the system. Thus, we consider criterion FR2 to be fully supported. Although it is not explicitly shown in the example presented in Figure 3, use cases can be complemented with pre- and post conditions which provide support for both validating the inputs and defining relationships between outputs and inputs. This allows us to consider criteria FR1 and FR5 to be fully supported. In addition, exceptional situations can be defined by adding additional description to use cases. Thus, criterion FR3 is fully supported. Finally, it is not clear how the effect of parameters can be described by use cases. There is no element in use case descriptions that explicitly supports it. However, the flexibility provided by the textual descriptions of actions can be used to describe the effect of parameters in natural language. Thus, we consider criteria FR4 to be partially supported.

Data requirements are not explicitly supported by the method, so there is no technique specifically proposed for specifying them.

*Our Analysis.* We consider that data requirements are partially supported by UIDs. UIDs properly describe the data that can be accessed by users, although the data that can be accessed by system operations is not correctly described (criterion DR3, data accessibility). In the same way, UIDs also indicate some data that the system must store (criterion DR4, data entities and relationships). These data are the data that



Sc1 - Browsing CDs

I'm a regular user of E-commerce applications. I like starting to browse CDs from a list of musical categories such as Rock, Pop, Classical, Blues, etc. Then, I like obtaining a list of CDs from which I can select the desired one.

Sc2 – Finding a CD

I like music very much. However I only buy the CDs of my favorite artists. Then, I like those E-commerce applications that allow me to quickly access to the CDs of a specific artist. Then a search by artist is mechanism that should be supported.

Sc3 – Adding a CD to the Shopping Cart

I'm the manager of the marketing department. We need that user can easily add the CD to the shopping cart when they find it. Furthermore, we need to automatically update the stock after adding the CD.

**Use Case:** Adding a CD to the shopping Cart

**Scenarios:** Sc1, Sc2, Ac3

**Roles:** Visitors, Customers

**Description:**

1. The application provides the user with a list of music categories
2. The user selects a category
3. The application returns the list of CDs that belong to the selected music category
4. The user can select a CD or perform a search of CD by artist
5. If the user selects a CD, it is returned a detailed description of the selected CD
6. From the CD description the user can activate the Add_to_Cart operation.
7. The system adds the selected CD to the shopping cart.

Fig. 3. Example of the requirements model in OOHDM.

users can access. However, data that is required only to support internal operations of the system is not described. Furthermore, UIDs allow us to indicate only data entities (e.g., music categories, CDs, etc). Relationships between entities cannot be explicitly defined. From this analysis, we consider criteria DR3 and DR4 to be partially supported. The rest of the criteria associated to data requirements are not supported.

Navigational requirements are explicitly supported by the method. The technique proposed to specify them is the one based on UIDs. The different interactions between the system and the user describe the information that is published. In addition, connections between interactions indicate how users can navigate this information. Thus, we consider criteria NR1 (published information) and NR2 (navigation possibilities) to be fully supported. Interactions in which the user introduces information are used

Table I. A Summary of the Requirements Specification Evaluation of OOHDM

| **OOHDM Requirements Specification Evaluation** | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | no | yes |
| **Technique** | Use Cases | - | UIDs |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** 👍 | **1. Info. Types** ✋ | **1. Published Info.** 👍 |
| | **2. Action Sequence** 👍 | **2. Use Freq.** ∅ | **2. Nav. Possibilities** 👍 |
| | **3. Ab. Situations** 👍 | **3. Access Cap.** ✋ | **3. Info. Filters** 👍 |
| | **4. Param. Effects** ✋ | **4. Entities and Rel.** ∅ | **4. Published Func.** 👍 |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ∅ | |
| | | **6. Data Retention** ∅ | |

👍 fully supported, ✋ partially supported, ∅ not supported

to filter the published information that allows us to consider criterion NR3 to be fully supported. Finally, interactions are complemented with mechanisms for indicating the functionality that can be activated. Thus, criterion NR4 is also considered to be fully supported.

A summary of the requirements specification evaluation of OOHDM is shown in Table I.

*4.1.4 Evaluation of the Model-Driven Development Perspective.* The requirements model of OOHDM is constructed from two main elements: use case descriptions and UIDs. UIDs constitute a visual language for describing requirements that is precisely characterized by a metamodel. However, use case descriptions are based on semi-structured textual descriptions that do not fit a precise metamodel. For instance, there are not different elements to define user and system actions. They both are defined by using natural language in the same field (Description). Thus, a metamodel that describes the meaning of a user action or system action cannot be created, although these elements are defined in the requirements model. Thus, we consider criteria MR1 (there exists a metamodel) and MR4 (visual concrete syntax) to be partially supported.

OOHDM is supported by a tool called HyperDE [Nunes and Schwabe 2006]. This tool provides an HTML-based interface for creating the OOHDM models that correspond to the conceptual design (PIM level). It also introduces mechanisms to automatically generate web applications implemented in Ruby on Rails[2] through PIM-to-code transformations. However, this tool does not provide support to create the requirements model. Thus, each element of the OOHDM requirements model must be manually defined. In this sense, we consider criterion MR5 to be not supported.

Finally, this approach presents guidelines to help analysts to define the conceptual and navigational schema from UIDs. They describe a CIM-to-PIM transformation and are defined from a set of rules presented in Vilain et al. [2000a] and Vilain and Schwabe [2002]. These rules indicate how the elements that define both a class diagram (conceptual schema) and a context diagram (navigational schema) can be systematically derived from UIDs. These rules are textually described by using natural language. However, no formalism is used in the rule definition. Thus, a certain degree of ambiguity is introduced in the rule definition derived from the use of natural language. In addition, the rules consider only requirements described by UIDs.

[2]http://www.rubyonrails.org.es/

Table II. A Summary of the Model-Driven Evaluation of OOHDM

| **OOHDM Model-Driven Evaluation** | | | | |
|---|---|---|---|---|
| **1 Metamodel** | **2 Mapping Functions** | **3 Tool for mappings** | **4 Visual Syntax** | **5 RE Specification Tool** |
| ✋ | ✋ | ∅ | ✋ | ∅ |

👍 fully supported, ✋ partially supported, ∅ not supported

Use case descriptions are not considered. Thus, we consider criterion MR2 (mappings functions) to be partially supported. Finally, rules must be manually interpreted and applied. No tool is provided to support the application of these rules; therefore, criterion MR3 is considered to be not supported.

A summary of the model-driven evaluation of OOHDM is shown in Table II.

*4.1.5 Main Strengths and Weaknesses.* The main strength of this approach is based on the combination of well-known use-case descriptions, which allow analysts to specify functional requirements of a web application, with UIDs, which allow the visual description of navigational requirements and some data requirements. Note that OOHDM is one of the few methods that provide a technique (UIDs) that is specifically defined for capturing web application requirements. In addition, this method presents a complete requirements engineering process that clearly guides analysts in the interaction with stakeholders in order to obtain a precise requirements model.

As a main weakness, only the UID technique seems to be defined for MDD purposes. Only UIDs fit a requirements metamodel and propose a visual language. In addition, guidelines for deriving conceptual models are defined by considering only UIDs. Uses cases seem to be used only for describing requirements. Another important drawback of this approach is the lack of tools for supporting MDD. There is neither a tool for specifying requirements nor a tool for applying the proposed derivation guidelines.

## 4.2 WSDM: Web Site Design Method

*4.2.1 General Description.* WSDM was developed by De Troyer and Leune [1998]. It is a user-centred approach. WSDM defines a web application by describing the requirements of the different groups of users that must interact with it. It was one of the first approaches in considering the problem of the diversity of users in web applications.

The development process of this approach is divided into five main phases.

—*Mission Statement Specification*. In this phase, the purpose and the subject of the web application must be expressed. The target audience must also be declared.
—*Audience Modeling*. In this phase, users are classified and grouped in order to study system requirements according to each user group.
—*Conceptual Design*. In this phase, both a class diagram is designed to represent the static model of the system and a navigational model to represent the possibilities of navigation.
—*Implementation Design*. During this phase, the conceptual design models are complemented with information required for an actual implementation, such as a site structure model, a presentation model and a logical data model.
—*Implementation*. In this phase, the result of the implementation design phase is written in a specific programming language.

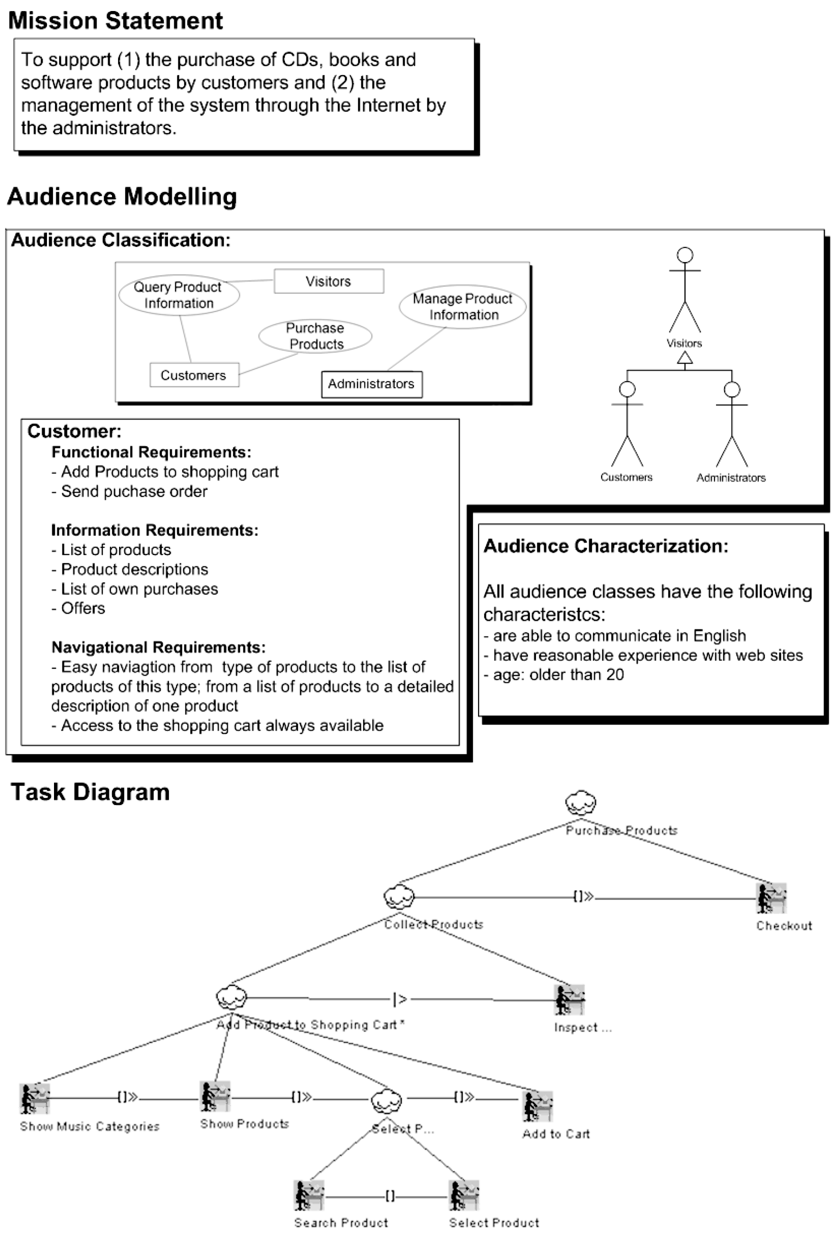
*4.2.2 Requirements Model.* Requirements are mainly handled in the Mission Statement Specification and Audience Modeling phases. In the first phase, a textual description of the system mission statement is done. Next, taking this description as source, the Audience Modeling phase is performed. In this phase, the concept of class audience, which is a group of potential visitors that have the same requirements, plays a key role. Thus, the Audience Modeling phase is performed in two steps.

(1) *Audience Classification*, in which people are classified according to the activities related to the purpose and subject of the web application. This formally identifies the different *audience classes*. WSDM proposes analyzing the organization environment where the application will be used and centers the attention on the stakeholders of the business processes supported by the application. In WSDM, the relationships between stakeholders and the business process activities performed are graphically represented by conceptual maps of roles and activities by means of activity diagrams. From this representation, the types of users that are going to interact with the web application (audience classes) are identified and represented in a taxonomy of audience classes. This taxonomy allows us to define inheritance relationships, which indicate that an audience class inherits all the requirements associated to another one. The requirements of each audience class are defined by means of a textual description.
(2) *Audience Class Characterization*, in which the characteristics of the members of each audience class are studied in detail. Some examples of user's characteristics are: level of experience with web sites in general, frequency of use, language issues, education/intellectual abilities, age, income, lifestyle, etc. For instance, younger people tend to be more experienced with web applications than older people are. These characteristics are described by using natural language. Some of the characteristics may be translated into usability requirements while others may be used later on in the implementation phase to guide the design of the "look and feel" of the web site, for example, choice of colors, fonts, graphics, etc.

Furthermore, recent works [De Troyer et al. 2008] introduce a task diagram based on the CTT technique [Paternò et al. 1997] to be used as a link between the requirements specification and both the navigational model and the class diagram that are created in the conceptual design phase. The purpose of this diagram is to describe the different tasks that each audience class need to be able to perform in order to identify the data and functionality that is needed for each task. In order to create these tasks the textual requirements associated to each audience class are taken as a source.

Figure 4 shows a partial version of the WSDM requirements model for the running example. The upper side of this figure shows the mission statement. Below the mission statement, there is a partial result of the audience modeling step. On the one hand, we show an audience classification with three audience classes: Customers, Visitors, and Administrators. Customers can query product information and purchase products; Visitors can query only product information; Administrators can manage product information. There is an inheritance relationship between visitors and customers, and another between visitors and administrators. In addition, the requirements associated to customers have been partially described. On the other hand, we can see how the same characteristics have been defined for the three audience classes (see audience characterization below the user taxonomy). Finally, in the lower left side of the figure, we can found a task diagram that describes some of the requirements associated to the Customer audience class. This diagram describes how customers can purchase products by interacting with the system.

*4.2.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified throughout the informal textual description associated to each audience class as well as the task diagrams. Task diagrams allow the specification of sequences of application tasks. This aspect



**Mission Statement**

To support (1) the purchase of CDs, books and software products by customers and (2) the management of the system through the Internet by the administrators.

**Audience Modelling**

**Audience Classification:**

**Customer:**

**Functional Requirements:**
- Add Products to shopping cart
- Send puchase order

**Information Requirements:**
- List of products
- Product descriptions
- List of own purchases
- Offers

**Navigational Requirements:**
- Easy naviagtion from type of products to the list of products of this type; from a list of products to a detailed description of one product
- Access to the shopping cart always available

**Audience Characterization:**

All audience classes have the following characteristcs:
- are able to communicate in English
- have reasonable experience with web sites
- age: older than 20

**Task Diagram**

Fig. 4. Example of a requirements specification in WSDM.

makes us consider criterion FR2 (sequence of system actions) to be fully supported. Considering that several task diagrams can be associated to a same requirement we consider criterion FR3 (abnormal situations) to be fully supported. The rest of criteria are not properly supported by the task diagram but they can be defined in the textual descriptions. The authors just prescribe the use of natural language for creating these descriptions, without indicating which aspects must be considered for specifying functional requirement. For instance, it is not indicated that we should describe input

Table III. A Summary of the Requirements Specification Evaluation of WSDM

| **WSDM Requirements Specification Evaluation** | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | no | yes |
| **Technique** | Natural language | Natural language | Natural language |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** ✋ | **1. Info. Types** ✋ | **1. Published Info.** ✋ |
| | **2. Action Sequence** 👍 | **2. Use Freq.** ✋ | **2. Nav. Possibilities** 👍 |
| | **3. Ab. Situations** 👍 | **3. Access Cap.** ✋ | **3. Info. Filters** ✋ |
| | **4. Param. Effects** ✋ | **4. Entities and Rel.** ✋ | **4. Published Func.** 👍 |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ✋ | |
| | | **6. Data Retention** ✋ | |

validations, or the effect of parameters, or input and output relationships to specify functional requirements. The authors leave the selection of these elements in the hands of the analysts. However, the use of natural language as a technique for specifying requirements facilitates the inclusion of any element in the WSDM requirements model. Thus, we consider the rest of functional criteria to be partially supported.

Data requirements are explicitly supported by the method. They are specified throughout the informal textual description associated to each audience class. Again, authors do not indicate the aspects that must be considered for specifying data requirements. However, due to the flexibility of natural language to describe requirements we consider all data criteria to be partially supported.

Navigational requirements are explicitly supported by the method. Although the criteria considered in this survey are not explicitly faced by authors, they can be specified throughout the informal textual descriptions associated to each audience class as well as the task diagrams. Task diagrams can be used to describe sequences of interaction tasks that users perform together with the systems. By properly naming these tasks, we can describe how users navigate the information provided by the system. Thus, we consider criterion NR2 to be fully supported. The task diagram also allows the description of system tasks, which connected to interaction tasks can be used to indicate the functionality that can be accessed by users throughout the navigation process. Therefore, we consider criterion NR4 to be fully supported. Search mechanisms can be represented by interaction actions with a proper name. However, it is not clear how the search filter can be described in detail. Thus, we consider criterion NR3 to be partially supported. Finally, task diagrams do not give a proper support to describe the published information in detail. This aspect can be supported by the flexibility provided by the informal textual descriptions so we consider criterion NR1 to be partially supported.

A summary of the requirements specification evaluation of WSDM is shown in Table III.

*4.2.4 Evaluation of the Model-Driven Development Perspective.* The requirements specification proposed by WSDM is based on the identification and classification of users and the description of the requirements associated to each user. To do this, the authors propose a precise notation to create user taxonomies. However, user requirements are described by using natural language. To improve this problem, a task diagram is proposed to be created in order to facilitate the definition of the navigational model and the class diagram from the specified requirements. A summary of the model-driven evaluation of WSDM is shown in Table IV.

Table IV. A Summary of the Model-Driven Evaluation of WSDM

| WSDM Model-Driven Evaluation | | | | |
|---|---|---|---|---|
| **1 Metamodel** | **2 Mapping Functions** | **3 Tool for mappings** | **4 Visual Syntax** | **5 RE Specification Tool** |
| ✋ | ∅ | ∅ | ✋ | ✋ |

Thus, although the classification of users and the task diagrams fits a metamodel, natural language description does not. Therefore, we consider criterion MR1 to be partially supported. There are no mapping functions that transform the requirements model into a conceptual model, and logically, neither there are tools for automatically applying mapping functions. Therefore, we consider criteria MR2 and MR3 to be not supported. As far as the use of a visual concrete syntax is concerned, it is used for specifying the user taxonomy and the task diagram, so criterion MR4 is considered to be partially supported. Since the task diagram is based on CTT, most of it can be created by using the CTT Environment.[3] Thus, criterion MR5 is considered to be partially supported.

*4.2.5 Main Strengths and Weaknesses.* One of the main strengths of WSDM is the consideration of users in requirements engineering activities. This aspect is very important in web application development since the web entails no obligation and users can stop using a web application because another is more attractive to them. Thus, web applications will only be used if they bring immediate advantage to the needs of each potential kind of user. To achieve this, we need to consider users from the early stages of the development process. In addition, this approach proposes a top-down strategy that allows us to specify requirements by starting with a general description (mission statement) and refining it to more concrete ones (audience characterization). This aspect is very useful for detecting new requirements when they are not always clear from the beginning. The use of textual descriptions provides a valuable flexibility to capture requirements that is complemented with more restricted descriptions based on a task diagram in order to create the WSDM navigational model and the class diagram.

However, the use of textual description constitutes the main weakness of this method in order to be used within MDD processes. In a MDD environment, these descriptions may be too ambiguous to be directly processed by model transformation tools. Although they are taken as source for creating a task diagram, which can be more easily interpreted by tools, this diagram does not fully capture all the requirements that are proposed to be included in the textual descriptions. In addition, textual descriptions may be overloaded and difficult to manage in the development of complex web applications. In Vilain et al. [2000b], it is stated that textual specifications are unwieldy when they are used in a validation process with users, mainly because of the lack of precision and conciseness. Finally, another important weakness of this approach is the lack of tools for supporting the specification of requirements as well as their transformation into conceptual models.

### 4.3 SOHDM: Scenario-Based Object-Oriented Hypermedia Design Methodology

*4.3.1 General Description.* SOHDM was developed by Lee et al. [1998]. This approach considers RE activities in the development process of a web application from its initial version. This approach takes some ideas from OOHDM [Schwabe et al. 1996]. However, as its name indicates, it is mainly based on the use of scenarios.

[3]http://giove.isti.cnr.it/tools/ctte/

The development process of this approach is divided into six main phases.

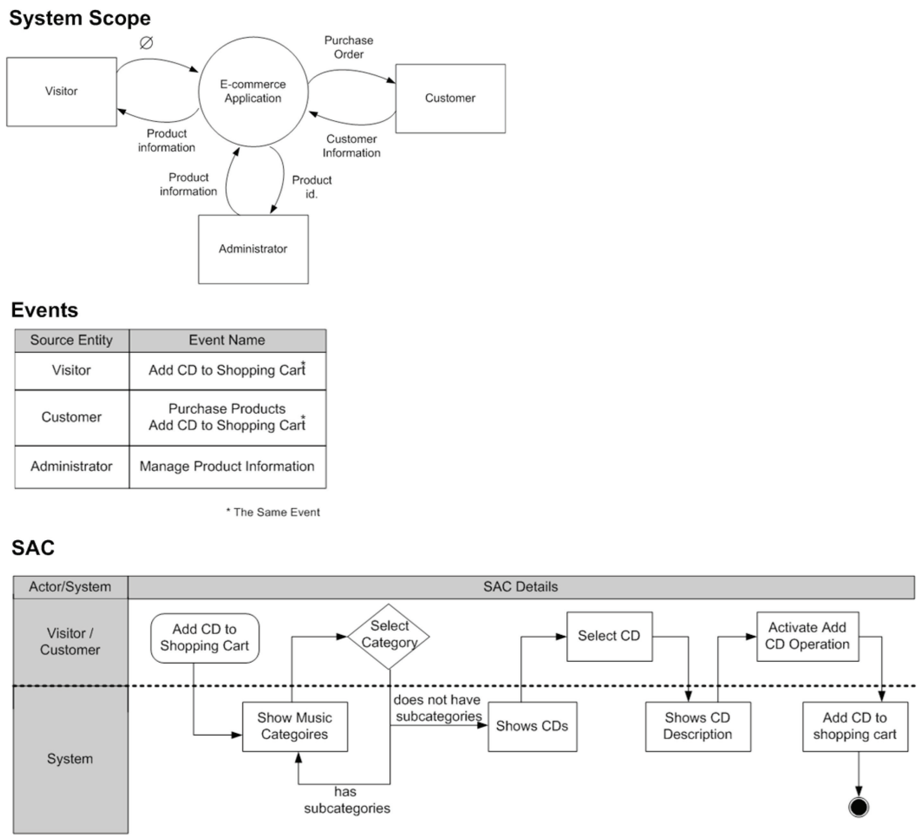
—*Analysis*. In this phase, the requirements of the web Application are described by using scenarios.
—*Object Model Realization*. This phase consists in creating a class diagram in which the static structure of the system is defined.
—*View Design*. This phase expresses how the system will be presented to the user.
—*Navigational Design*. In this phase, a navigational class model is developed in order to express the possibilities of navigation in the system.
—*Realization of the Implementation*. This phase consists in designing the web pages and the flow among them, other detailed interface aspects and a relational database.
—*Construction of the System*. In this phase, the web application is finally built.

*4.3.2 Requirements Model.* Requirements are handled in the Analysis phase. SOHDM proposes to specify web application requirements through three main steps:

(1) Definition of the scope of the system, which delimits the web application to be developed from external entities. In order to define the scope of the system, SOHDM proposes the *system scope diagram* that is based on the well-known *data flow diagrams* (DFD) [Demarco 1979], although context diagrams (e.g., zero-level DFD) are a good alternative. In the System Scope diagram, the authors identify external entities as well as the information exchanged between these external entities and the system.
(2) Identification of events that trigger the communication between external entities and the application. One or more events are identified for each external entity. Events are defined in a specific table.
(3) Association of scenarios to the identified events. Scenarios are described by means of *scenario activity charts* (SACs), which is a notation created by the authors. SAC describes business processes according to actors. An actor is an operator of specific activities, that is, a creator of events.

Figure 5 shows a partial view of the SOHDM requirements model of the running example. The upper side of this figure shows the scope of the web application in which three external entities are identified: Visitor, Customer, and Administrator. These external entities are the types of users that are going to interact with the web application. Below the system scope, we can find the events identified for each external entity. In accordance with these events, visitors can add products to the shopping cart, customers can add products to the shopping cart and purchase products, and administrators can manage the product information. In the lower side of this figure, the SAC that describes the event Add CD to Shopping Cart is shown. External entities are the primary candidates for actors in SACs. An event is a starting point, a trigger of a scenario. An activity is an operation by which an actor completes a scenario. A decision node represents an activity that enables an actor to choose the next activity. An activity flow is a sequence of activities. Termination is the end of a scenario.

*4.3.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified throughout the three diagrams proposed by SOHDM. The scope diagram allows us to indicate which input must be introduced by external entities in order to obtain a specific output. This aspect makes us consider criterion FR5 (relationships between output and inputs) to be fully supported. However, there are no mechanisms to define input validations, so criterion FR1 is considered to be not supported. The set of operations that the system must perform is fully described by SACs, in which the activities to accomplish a



| Source Entity | Event Name |
|---|---|
| Visitor | Add CD to Shopping Cart* |
| Customer | Purchase Products<br>Add CD to Shopping Cart* |
| Administrator | Manage Product Information |

* The Same Event

Fig. 5. Example of a requirements specification in SOHDM.

scenario are described. Thus, criterion FR2 is considered to be fully supported. Additional SACs can be defined in order to describe exceptional scenarios, so we consider criterion FR3 to be fully supported. Finally, parameters are described in SACs by attaching them (when it is required) to the input arcs that connect system activities. Thus, we consider criterion FR4 to be fully supported.

Data requirements are not explicitly supported by the method, so there is no technique specifically proposed for specifying them.

*Our Analysis.* We consider that the data that must be stored within the system could be partially extracted from either the parameters defined in SACs or the activities that involve an inquiry of information (criterion DR4, data entities, and relationships). However, no details about these data can be given in SACs. Since SACs are associated to actors, we can determine the users that can access the data described by them, although the data that is accessed by system operations cannot be detailed (criterion DR3, data accessibility). From this analysis, we consider criteria DR4 and DR3 to be partially supported. The rest of the criteria associated to the specification of data requirements are not supported.

Navigational requirements are not explicitly supported by the method, so there is no technique specifically proposed for specifying them.

*Our Analysis.* We consider that SACs could be used to partially describe data requirements. We can define actions of the system whose goal is to show information. We can

Table V. A Summary of the Requirements Specification Evaluation of SOHDM

| SOHDM Requirements Specification Evaluation | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | no | yes |
| **Technique** | Scope diagrams, SACs | - | - |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** ∅ | **1. Info. Types** ∅ | **1. Published Info.** ✋ |
| | **2. Action Sequence** 👍 | **2. Use Freq.** ∅ | **2. Nav. Possibilities** ✋ |
| | **3. Ab. Situations** 👍 | **3. Access Cap.** ✋ | **3. Info. Filters** ✋ |
| | **4. Param. Effects** 👍 | **4. Entities and Rel.** ✋ | **4. Published Func.** ✋ |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ∅ | |
| | | **6. Data Retention** ∅ | |

Table VI. A Summary of the Model-Driven Evaluation of SOHDM

| SOHDM Model-Driven Evaluation | | | | |
|---|---|---|---|---|
| **1 Metamodel** | **2 Mapping Functions** | **3 Tool for mappings** | **4 Visual Syntax** | **5 RE Specification Tool** |
| 👍 | ∅ | ∅ | 👍 | ∅ |

also define actions of the user whose goal is to access this information. In this way, we are able to describe how users navigate the information provided by the system (criterion NR2). In addition, we can describe the type of information accessed by the user (e.g., music categories, CDs, etc) by properly naming these actions, although details about this information cannot be given (criterion NR1, published information). The fact of describing the published information by means of actions allows us to connect them to system actions in order to indicate the functionality that can be accessed by users throughout the navigation process (criterion NR4). As far as mechanisms to filter the information, they can be emulated by means of the combination of actions in which users introduce data with actions in which the system searches for information by using the data introduced by users (criterion NR3). From this analysis, we consider all the criteria associated to the specification of navigational requirements to be partially supported.

A summary of the requirements specification evaluation of SOHDM is shown in Table V.

*4.3.4 Evaluation of the Model-Driven Development Perspective.* The requirements model proposed by SOHDM is mainly defined by a visual notation. Although a textual template is used, it must be constructed according to a specific structure. This makes it possible for the SOHDM to fit a precise metamodel. Thus, we consider criteria MR4 and MR1 to be fully supported. However, there is no tool that supports analysts in the construction of this requirements model that make us consider criterion MR5 to be not supported. Finally, there are neither mapping functions that facilitate the derivation of conceptual models from requirements nor tools for automatically applying mapping functions. Thus, criteria MR2 and MR3 are considered to be not supported.

A summary of the model-driven evaluation of SOHDM is shown in Table VI.

*4.3.5 Main Strengths and Weaknesses.* The main strength of this approach is that requirements are all specified by using graphical notations or structured textual templates that fit a precise metamodel. This aspect facilitates the use of this technique in MDD environments. Another important strength is the consideration of the system boundaries by means of its scope diagram. This aspect allows us to clearly delimitate the system at the requirements level as well as the ways in which it interact with external entities such as users or even other software systems.

The main weakness of this approach is that SACs are mainly focused on describing functional requirements in which the exchange of information between the system and the user is almost limited to required operation parameters. Thus, the description of navigational and data requirements is not always easy in the case of web applications that are mainly focused on just providing information. The major part of these web applications allows users to navigate information without performing any operation. Thus, representing all the different possibilities for navigating information with a DFD-based notation can overload the requirements specification, making it difficult to manage and understand. Finally, as happens with the previous two methods, there is no tool for specifying requirements.

## 4.4 UWE: UML-Based Web Engineering

*4.4.1 General Description.* UWE was developed by Nora Koch in 1998 [Koch 2000; Koch et al. 2006; Mandel et al. 1998]. UWE is a methodological approach for the development of web applications that completely bases its diagrammatic techniques on UML. This approach is currently open and a lot of research work is continuously being presented in order to improve it. UWE proposes a semi-automatic design process supported by the case tool MagicUWE.[4]

The development process of UWE is based on the Model Driven Architecture [OMG 2005]. The aim of the UWE development process is to automatically transform models in each step by using rules defined at the metamodel level. We can divide this process into three main stages.

— The process starts with the specification of requirements. These requirements define the CIM model.
— Platform-independent analysis models (PIMs) are derived from the requirements. On the PIM level, the different concerns of web applications (the content, the navigation, the business processes, and the presentation) are designed in separate models. These models are integrated into a so-called "big picture" that merges all the concerns together and is used in validation of the design models.
— Finally, the platform specific models (PSMs) are derived from this validation model. Program code can be generated from PSMs.

*4.4.2 Requirements Model.* Initial versions of UWE propose the specification of requirements by means of use case diagrams together with textual descriptions of use cases [Koch 2000]. In more recent works [Koch et al. 2006], UWE has been extended in order to support the UML profile for web requirements (WebRE) [Escalona and Koch 2007]. This profile has been defined by the UWE's authors in collaboration with the authors of the web engineering method NDT (presented in section 4.9). We focus on the most recent approach based on WebRE, considering that the other works may be obsolete.

UWE currently proposes the use of use case diagrams and activity diagrams in order to capture web application requirements. Use case diagrams are used to represent an overview of the web application requirements while activity diagrams provide a more detailed view. As Figure 6 shows, UWE distinguishes between two types of use cases: *navigation use cases* (marked with □) which comprise a set of *browse activities* (marked

[4]http://uwe.pst.ifi.lmu.de/

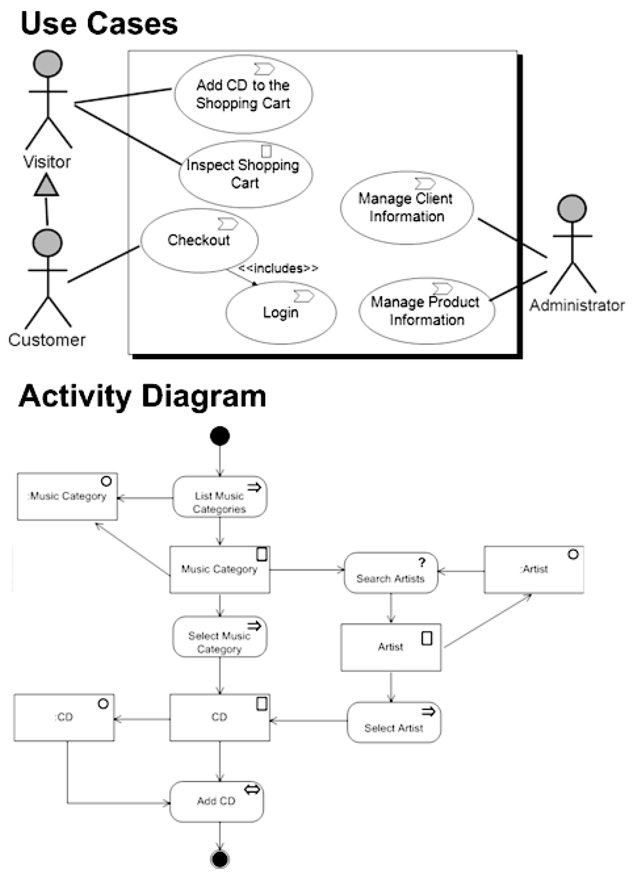


Fig. 6. Example of a requirements specification in UWE.

with ⇒) that the user will perform to reach a *target node* (marked with □). A browse activity is the action of following a link. A browse activity can be enriched by *search actions* (marked with ?). A search action has a set of parameters, which let queries be defined on a *content* (marked with ◯). The results are shown in the *target node*. The second kind of use case is the *webProcess* (marked with ⊐), which is defined by activities of type browse, search, and at least one *user transaction* (marked with ⇔). A user transaction represents more complex activities that are expressed in terms of transactions that have been initiated by the user, like checkout in an e-shop or an online reservation.

Figure 6 shows a partial view of the UWE requirements model for the running example. The upper portion of this figure shows the use case diagram. The lower portion shows the activity diagram that describes the process use case Add CD to the Shopping Cart. In accordance with this diagram, when users activate the link List Music Category, they access the node Music Category. From this node, users can select one category and then access the node CD. Another way of accessing this node is by searching for artists from the node Music Category. From this node, users can activate the Add CD transaction.

*4.4.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified by means of web process use cases. These use cases are first identified in the use case diagram and then described in detail by means of activity diagrams. These activity diagrams allow us to describe user transactions, which represent activities that the system must perform. These descriptions support analysts in the specification of system operations. Thus, we consider criterion FR2 to be fully supported. Additionally, pre- and post-conditions can be defined for each use case. These conditions can be used for validating the inputs and defining relationships between outputs and inputs. This aspect makes us consider criteria FR1 and FR5 to be fully supported. Furthermore, additional descriptions can be associated to use cases in order to consider exceptional situations. This allows us to consider criterion FR3 to be fully supported. The description of the effect of parameters is not explicitly considered in this requirements model, so criterion FR4 is considered to be not supported.

Data requirements are not explicitly supported by the method; therefore, there is no technique specifically proposed for specifying them.

*Our Analysis.* We consider that some data requirements are implicitly captured by the proposed techniques. The entities that the system must store (criterion DR4) can be extracted from the content elements defined in activity diagrams (e.g., music category, CD, etc). However, relationships between these entities cannot be defined. In addition, it is not clear how details about this information (e.g., properties of CDs) can be defined. Furthermore, through the association of actors to use cases, it can be determined which data (content elements) can be accessed by users. In the same way, content elements can be associated to transactions that define the data that can be accessed by system operations (criterion DR3). From this analysis, we consider criteria DR3 and DR4 to be partially supported. The rest of the criteria associated to data requirements are not supported.

Navigational requirements are explicitly supported by the method. They are captured by means of Navigation use cases. These use cases allow us to define the information published by the web application by means of node elements, which are associated to content elements. The connection of these nodes with browse activities indicates how users can navigate this information. These two aspects allow us to consider criteria NR1 and NR2 to be fully supported. Furthermore, we can also connect nodes to search activities that allow users to filter information published by nodes. This fact makes us consider criterion NR3 to be fully supported. Finally, the connection of nodes to transactions indicates the functionality that users can activate when they navigate the published information; therefore, we consider criterion NR4 to be fully supported.

A summary of the requirements specification evaluation of UWE is shown in Table VII.

*4.4.4 Evaluation of the Model-Driven Development Perspective.* The UWE requirements model is fully defined by means of a visual notation and fits the WebRE metamodel [Escalona and Koch 2007]. Thus, we consider criteria MR4 and MR1 to be fully supported.

The definition of UWE models was initially supported by the ArgoUWE tool [Knapp et al. 2003], which is a plugin of the ArgoUML open source modelling tool.[5] However, the development of this tool has been discontinued because it is not based on UML 2.0. It has been replaced by a new tool developed over the MagicDraw platform[6] called MagicUWE, which is based on UML 2.0 and provides a more comfortable interface.

[5]http://www.argouml.org
[6]http://www.magicdraw.com/

Table VII. A Summary of the Requirements Specification Evaluation of UWE

| **UWE Requirements Specification Evaluation** | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | no | yes |
| **Technique** | Web process use cases | - | Navigational use cases |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** 👍 | **1. Info. Types** ∅ | **1. Published Info.** 👍 |
| | **2. Action Sequence** 👍 | **2. Use Freq.** ∅ | **2. Nav. Possibilities** 👍 |
| | **3. Ab. Situations** 👍 | **3. Access Cap.** ✋ | **3. Info. Filters** 👍 |
| | **4. Param. Effects** ∅ | **4. Entities and Rel.** ✋ | **4. Published Func.** 👍 |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ∅ | |
| | | **6. Data Retention** ∅ | |

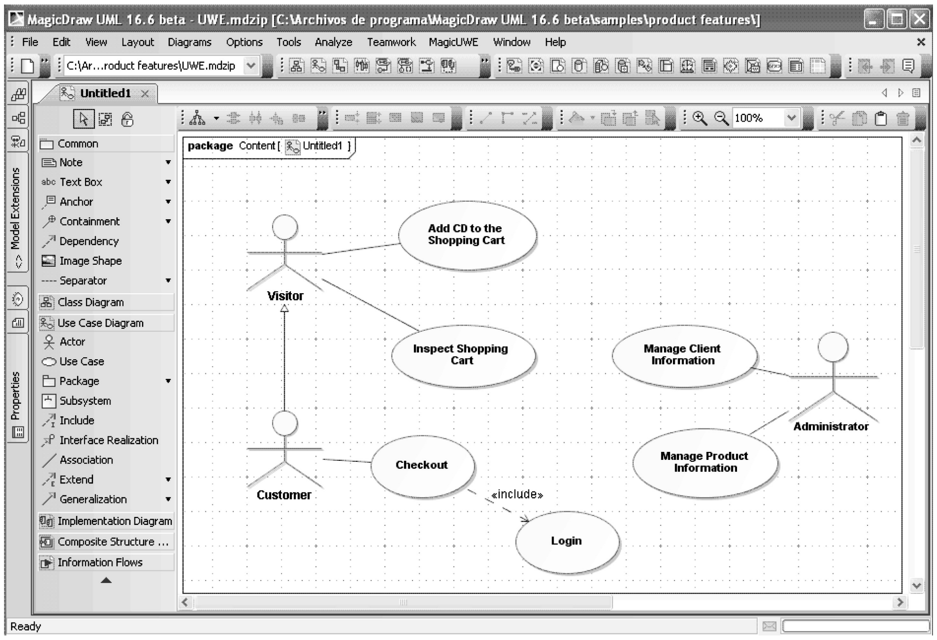


Fig. 7. A snapshot of MagicUWE.

MagicUWE is focused on supporting the modeling activities of the UWE development process. This tool provides full support to create the different views that define the UWE PIM model. However, the creation of the CIM model (the requirements model) is partially supported. In the current version of the tool, only the construction of the use case diagram is supported. There is no support for creating the activity diagrams that describe each use case. However, the authors are currently working on incorporating new stereotypes that allow the creation of a full UWE requirements model. Thus, we consider criterion MR5 to be partially supported. Note, however, that MagicUWE provides support for creating activity diagrams with process definition purposes. These activity diagrams are complemented with stereotypes to properly capture the characteristics of business processes. Figure 7 shows a snapshot of MagicUWE that illustrates how the use case diagram introduced in this section is created. Note that,

Table VIII. A Summary of the Model-Driven Evaluation of UWE

| UWE Model-Driven Evaluation | | | | |
|---|---|---|---|---|
| 1 Metamodel | 2 Mapping Functions | 3 Tool for mappings | 4 Visual Syntax | 5 RE Specification Tool |
| 👍 | 👍 | ✋ | 👍 | ✋ |

although MagicUWE is a plugin for the MagicDraw tool, UWE provides a profile that can be used in any UML-compliant tool.

As far as mapping functions is concerned, a model-to-model transformation is proposed in Koch et al. [2006] in order to obtain draft versions of both UWE navigational models and UWE content models from requirements models. In terms of MDA, this transformation constitutes a CIM-to-PIM transformation. It is based on the definition of mappings between the WebRE metamodel and the UWE metamodel, and they are specified as QVT transformations [OMG 2008]. This model-to-model transformation allows us to consider criterion MR2 as fully supported. When this model-to-model transformation was presented, there were no tools for automatically applying QVT transformations. The authors stated that they were looking forward to tools supporting the QVT transformation language. Currently, we can find tools such as the QVT operational Eclipse plug-in[7] that gives support to apply these transformations. However, the authors have not explicitly decided on the use of this tool and nor have they analyzed if the implementation of their transformation by means of this tool is feasible. Thus, we consider then criterion MR3 to be partially supported. Note, however, that there is significant tool support to transforms UWE models at the PIM level. On the one hand, the MagicUWE tool implements several ATL PIM-to-PIM transportations. On the other hand, there is an Eclipse plugin that generates JSF applications from UWE PIM models trough a PIM-PSM-code transformation.

A summary of the model-driven evaluation of UWE is shown in Table VIII.

*4.4.5 Main Strengths and Weaknesses.* The main strength of UWE is that it provides techniques that are especially created for the specification of web application requirements. Specifically, it provides a visual diagram to complement use cases, which allows us to properly capture navigational requirements. Another important characteristic of UWE is that it is fully oriented to be used in MDD of web applications. In addition, the UWE profile is totally based on UML, which allows using it in any UML-compliant tool. However, the tools provided by UWE in order to specify web requirements and derive conceptual models from them need to be improved. This constitutes its main weakness.

### 4.5 WebML: Web Modeling Language

*4.5.1 General Description.* WebML was developed by Piero Fraternali and Stefano Ceri in 2000 [Ceri et al. 2000, 2003]. WebML is a high-level modeling language for web applications. WebML follows the style of both Entity-Relationship and UML offering a proprietary notation and a graphical representation using the UML syntax. This approach is currently not closed and is continuously being extended and improved. Finally, WebML is one of the few approaches that present a tool that supports its development process. This tool is called WebRatio[8] and is being currently applied in an industrial environment.

[7]http://www.eclipse.org/m2m/
[8]http://www.webratio.com

The development process of this approach is divided into six incremental main phases.

— *Requirements Specification*. This phase focuses on collecting information about the application domain and the expected functions, and specifying them.
— *Conceptual Modeling*. This phase consists in defining WebML-based conceptual schemas, which express the organization of the application at a high level of abstraction, independently from implementation details.
— *Implementation*. This phase consists in the translation of the WebML-based conceptual schema into a specific implementation technology.
— *Testing and Evaluation*. In this phase, the web application is tested and validated in order to improve its internal and external quality.
— *Deployment*. This phase consists in deploying the web application on the top of a specific architecture.
— *Maintenance and Evolution*. In this phase, the application is maintained and possibly evolved after it is deployed.

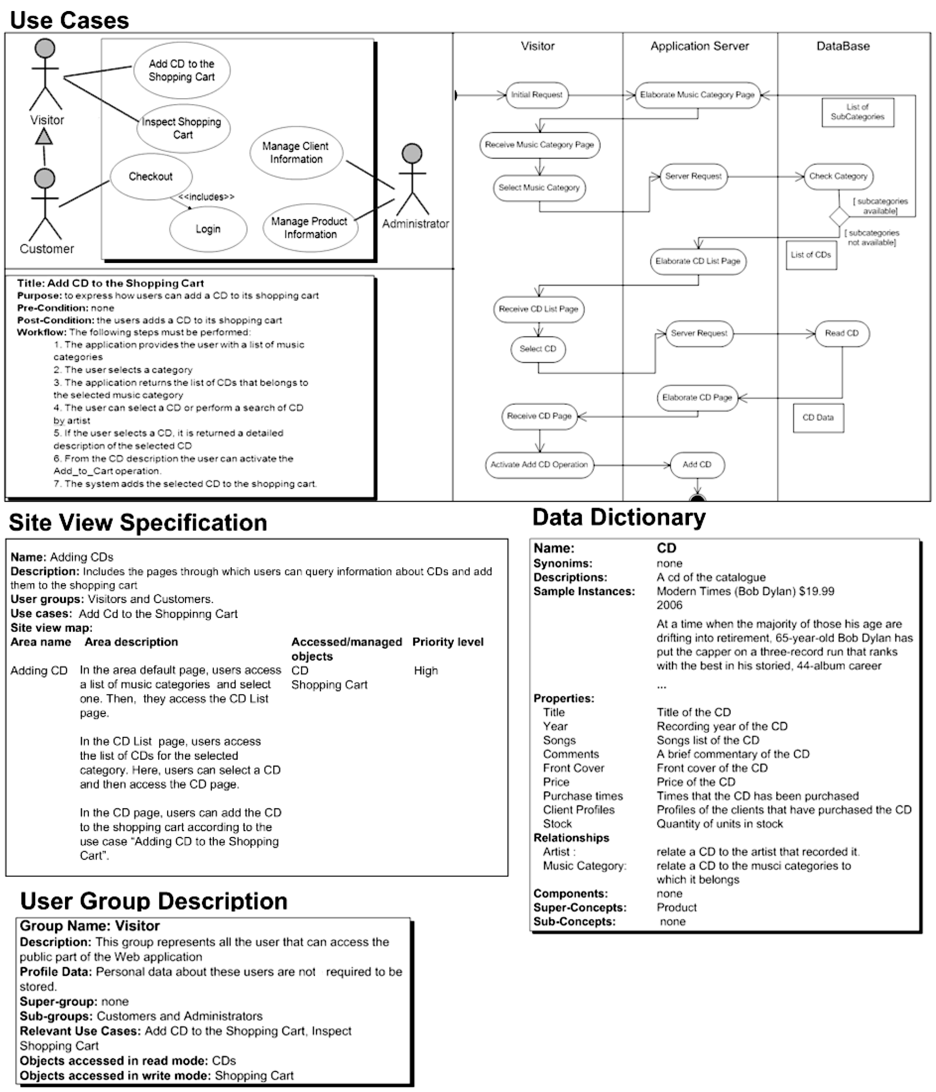
Furthermore, WebML also promotes a strategy of rapid prototyping from the conceptual schema for early detection of misunderstandings in the captured requirements.

*4.5.2 Requirements Model.* Requirements are handled in the requirements specification stage. WebML proposes the inclusion of the following aspects in its requirements model.

— The list of the user groups that will access the web application, together with a preliminary assignment of their access rights over the information content. To do this, user groups are first organized in a hierarchy in which inheritance relationships among them can be defined. Next, each user group is characterized by means of a structured textual template.
— The most significant application use cases, which show the interactions between the identified user groups and the application. To do this, a use case diagram is first defined. Next, use cases are described by means of a structured textual template. If use cases are complex, activity diagrams can be used to visually express their workflow.
— A data dictionary that collects the most relevant information objects in the application domain. To do this, a structured textual template is proposed.
— The informal specification of the site views that will allow users to accomplish the functions expressed by the identified use cases. To do this, a set of textual specification sheets must be completed.
— A set of presentation guidelines, which give indications about the look and feel of the interfaces to be developed. To do this, the use of mock-ups and Cascading Style Sheets (CSS) are proposed.

Additionally, WebML also proposes creating acceptance tests in order to evaluate nonfunctional requirements such as usability, performance, scalability, etc.

Figure 8 shows a partial version of the WebML requirements model for the running example. First, we can see a use case diagram in which user groups are hierarchically described and associated to use cases. In accordance with this diagram, Visitors can add CDs to the shopping cart and inspect the shopping cart, Customers can perform the same use cases as Visitors plus Checkout, and Administrators can manage client and product information. Below the use case diagram is the structured textual description of the use case Add CD to the shopping cart and its description by means of an activity diagram. According to these descriptions, users can add a CD by first selecting a music category. Then, users obtain a list of CDs from which they can select



**Title: Add CD to the Shopping Cart**
**Purpose:** to express how users can add a CD to its shopping cart
**Pre-Condition:** none
**Post-Condition:** the users adds a CD to its shopping cart
**Workflow:** The following steps must be performed:

1. The application provides the user with a list of music categories
2. The user selects a category
3. The application returns the list of CDs that belongs to the selected music category
4. The user can select a CD or perform a search of CD by artist
5. If the user selects a CD, it is returned a detailed description of the selected CD
6. From the CD description the user can activate the Add_to_Cart operation.
7. The system adds the selected CD to the shopping cart.

**Site View Specification**

**Name:** Adding CDs
**Description:** Includes the pages through which users can query information about CDs and add them to the shopping cart
**User groups:** Visitors and Customers.
**Use cases:** Add Cd to the Shoppinng Cart
**Site view map:**

| Area name | Area description | Accessed/managed objects | Priority level |
|---|---|---|---|
| Adding CD | In the area default page, users access a list of music categories and select one. Then, they access the CD List page.<br><br>In the CD List page, users access the list of CDs for the selected category. Here, users can select a CD and then access the CD page.<br><br>In the CD page, users can add the CD to the shopping cart according to the use case "Adding CD to the Shopping Cart". | CD<br>Shopping Cart | High |

**Data Dictionary**

| | |
|---|---|
| **Name:** | **CD** |
| **Synonims:** | none |
| **Descriptions:** | A cd of the catalogue |
| **Sample Instances:** | Modern Times (Bob Dylan) $19.99<br>2006<br>At a time when the majority of those his age are drifting into retirement, 65-year-old Bob Dylan has put the capper on a three-record run that ranks with the best in his storied, 44-album career<br>... |
| **Properties:** | |
| Title | Title of the CD |
| Year | Recording year of the CD |
| Songs | Songs list of the CD |
| Comments | A brief commentary of the CD |
| Front Cover | Front cover of the CD |
| Price | Price of the CD |
| Purchase times | Times that the CD has been purchased |
| Client Profiles | Profiles of the clients that have purchased the CD |
| Stock | Quantity of units in stock |
| **Relationships** | |
| Artist : | relate a CD to the artist that recorded it. |
| Music Category: | relate a CD to the musci categories to which it belongs |
| **Components:** | none |
| **Super-Concepts:** | Product |
| **Sub-Concepts:** | none |

**User Group Description**

**Group Name: Visitor**
**Description:** This group represents all the user that can access the public part of the Web application
**Profile Data:** Personal data about these users are not required to be stored.
**Super-group:** none
**Sub-groups:** Customers and Administrators
**Relevant Use Cases:** Add CD to the Shopping Cart, Inspect Shopping Cart
**Objects accessed in read mode:** CDs
**Objects accessed in write mode:** Shopping Cart

Fig. 8. Example of requirements specification in WebML.

the desired one. Then users can add the CD to the shopping cart. The set of web pages that are required to add a CD are identified on both the activity diagram and the site view specification shown below. Furthermore, according to the Visitor description (see user group description at the bottom portion of the figure), visitors are those users that can access only the public part of the web application. They can access CDs only to read, and the shopping cart can be accessed in write mode. Finally, the different properties that the system must store about a CD, its sub- and super-concepts and their relationships with other concepts are shown in the data dictionary template.

*4.5.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified by means of the use case diagrams and their associated descriptions. As explained in other methods that use this

Table IX. A Summary of the Requirements Specification Evaluation of WebML

| WebML Requirements Specification Evaluation | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | yes | yes |
| **Technique** | Use Cases, Activity Diagrams | Data dictionary | Site view specification templates |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** 👍 | **1. Info. Types** ∅ | **1. Published Info.** ✋ |
| | **2. Action Sequence** 👍 | **2. Use Freq.** ∅ | **2. Nav. Possibilities** 👍 |
| | **3. Ab. Situations** 👍 | **3. Access Cap.** 👍 | **3. Info. Filters** ✋ |
| | **4. Param. Effects** ✋ | **4. Entities and Rel.** 👍 | **4. Published Func.** ✋ |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ∅ | |
| | | **6. Data Retention** ∅ | |

technique, criteria FR1 (input validation) and FR5 (relation between input and output) are fully supported by means of the pre- and post-conditions that can be defined for each use case. Criterion FR2 (sequence of operations) is fully supported by means of the descriptions associated to use cases (both textual template and activity diagram). Criterion FR3 (abnormal situations) is fully supported by means of the exception descriptions that can be associated to use cases. Finally, criterion FR4 (description of the effect of parameters) is not explicitly considered in this requirements model. However, we consider it to be partially supported because it can be easily included in the textual descriptions associated to use cases.

Data requirements are explicitly supported by the method. They are basically specified by means of the data dictionary templates and the user group descriptions. On the one hand, data dictionary templates allow us to describe the properties of concepts as well as their relationships with other concepts. This fact makes us consider criterion DR4 to be fully supported. On the other hand, user group descriptions allow us to indicate the accessing capabilities of users over the described concepts. The accessing capabilities of system actions can be described in the use case descriptions. Thus, criterion DR3 is fully supported. The rest of the criteria are not supported.

Navigational requirements are explicitly supported by the method. They are specified by means of the site view specification templates. These templates allow us to divide the web application into areas and also indicate the web pages that must appear in each area and how they must be connected. This type of descriptions allows us to indicate how users navigate information. Thus, we consider criterion NR2 to be fully supported. However, these descriptions are done in natural language, which makes it difficult to describe the information provided by each page in detail. This fact makes us consider criterion NR1 to be partially supported. In the same way, we must use natural language descriptions to describe filter mechanisms or the functionality published in each page, so criteria NR3 and NR4 are considered to be partially supported.

A summary of the requirements specification evaluation of WebML is shown in Table IX.

*4.5.4 Evaluation of the Model-Driven Development Perspective.* Some of the requirements specification techniques proposed by WebML present a visual notation such as activity diagrams and use case diagrams and others are based on textual notations such as user group templates or site view specifications. Thus, we consider criterion MR4 (visual syntax) to be partially supported. With respect to a metamodel, there exists some

Table X. A Summary of the Model-Driven Evaluation of WebML

| **WebML Model-Driven Evaluation** | | | | |
|---|---|---|---|---|
| **1 Metamodel** | **2 Mapping Functions** | **3 Tool for mappings** | **4 Visual Syntax** | **5 RE Specification Tool** |
| ✋ | ✋ | ∅ | ✋ | ∅ |

information described in the requirements model that cannot be described in a metamodel, which is basically that created by natural language descriptions such as the site view description. Therefore, we consider criterion MR1 to be partially supported.

WebML is supported by the commercial tool WebRatio. This tool is mainly focused on providing both support for the Conceptual Modeling phase and mechanisms for automatically generating code from conceptual models through PIM-to-Code transformations. The specification of requirements is not supported by this tool, so criterion MR5 is considered to be not supported.

With regard to mapping functions, some guidelines are given in Ceri et al. [2003] to derive a conceptual design from requirements (which define a CIM-to-PIM Transformation). These guidelines indicate how a coarse design can be obtained from the functional requirements and the site view maps. This coarse design represents a preliminary conceptual design that is further translated into WebML models. Other works [Brambilla 2003; Brambilla et al. 2006] explain how to obtain WebML models from the workflow descriptions associated to use cases. The main drawback of these two types of guidelines is that they are textually described by using natural language. No formalism is used in the definition that introduces a certain degree of ambiguity in their definition. Thus, we consider criterion MR2 to be partially supported. Finally, the given guidelines must be manually interpreted and applied. No tool is provided to support the application of them; therefore, criterion MR3 is considered to be not supported.

A summary of the model-driven evaluation of WebML is shown in Table X.

*4.5.5 Main Strengths and Weakness.* The main strength of this approach is that it proposes a very complete set of techniques to specify the requirements of web applications. These techniques consider the three types of requirements studied in this work and other ones such as personalization or interface requirements. In addition, WebML is supported by WebRatio, a commercial tool that is currently being used in the development of many web applications. This gives the authors a lot of experience in the specification of requirements in real web development projects.

Its main weakness is based on the extensive use of textual descriptions, which can make the use of this requirement model in MDD environments that require automation difficult. In addition, the proposed RE techniques are still not supported by WebRatio; therefore, the requirements model needs to be done manually.

### 4.6 OO-H: Object-Oriented Hypermedia Method

*4.6.1 General Description.* The Object-Oriented Hypermedia Method was presented by Gomez et al. [2000] and Cachero [2003]. This approach emerged formerly as an extension of OO-Method [Pastor et al. 1997], a design method for object-oriented systems. However, in the most recent publications, the authors dissociate OOH from OO-Method and focus on dealing with personalization of web sites [Garrigos et al. 2005]. This approach is supported by the tool Visual Wade.[9]

The development process of this approach is divided into four main stages.

[9]http://www.visualwade.com/

—*Requirements Analysis*. In this phase, the web application requirements are specified for each different type of user.
—*Engineering*. This phase involves all the activities related to the analysis and design of the software product. Specifically, there are five activities: two analysis activities (domain and navigational analysis) and three design activities (domain, navigation and presentation design).
—*Construction and Adaptation*. This phase constitutes the implementation of the web application.
—*Client Evaluation*. In this phase, clients evaluate the developed web application.

*4.6.2 Requirements Model.* Requirements are handled in the requirements analysis phase. OO-H is mainly focused on both providing abstract mechanisms in order to capture web applications at the conceptual level and defining strategies for the automatic generation of code. RE activities for web applications are not given much consideration. OO-H proposes only the use of UML use case diagrams. OO-H does not propose any other technique to complement use cases. In Cachero [2003] states[10] that: "Although UML proposes to complement Use Case diagrams with other mechanisms (scenarios, activity diagrams, collaboration or sequence diagrams, etc.), OO-H does not suggest the use of any. If other mechanisms are used, they are used only as documentation and they do not influence the rest of the phases of the OOH development process." Although in an extension of the method [Cachero and Koch 2002] OO-H proposes to complement use cases with activity diagrams, they are used during analysis activities. They are not used as a technique for the specification of web application requirements.

In order to improve the support provided for requirements activities, a web Requirements Metamodel Extension (WE-RMExt) [Molina et al. 2008] has been recently defined. This metamodel extends the common web engineering metamodel presented by the MDWEnet initiative [Valverde et al. 2007]. The proposed extension defines the elements that participate in the requirements management process. This metamodel distinguishes between *functional requirements,* which indicate what the system must do, and *nonfunctional requirements*, which indicate how the system must do it. Each requirement is associated to a goal and a stakeholder. Furthermore, requirements are organized into tree-like structure by means of a *descomposed Into* relationship between requirements. Requirements are also associated to measurement elements that provide a powerful mechanism to quantify requirements with evaluation purpose. Finally, it is work nothing that, although the proposed metamodel clearly states the element that must be considered in requirements specifications, there is no concrete syntax that facilitates the instantiation of this metamodel.

Figure 9(a) shows a partial view of the use case diagram proposed by initial version of OO-H in order to specify the requirements of the running example. Figure 9(b) shows an example of a tree-like instantiation of the metamodel proposed in recent works for the running example.

*4.6.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified by means of the use case diagram. They are also considered in the metamodel defined in recent works. Thus, considering that pre- and post conditions may be associated to use cases we consider the criteria FR1 (validation of inputs) and FR5 (relation of outputs and inputs) to be fully supported. This approach, however, does not propose any technique for describing

[10]Text translated and adapted from its original version in Spanish to English.

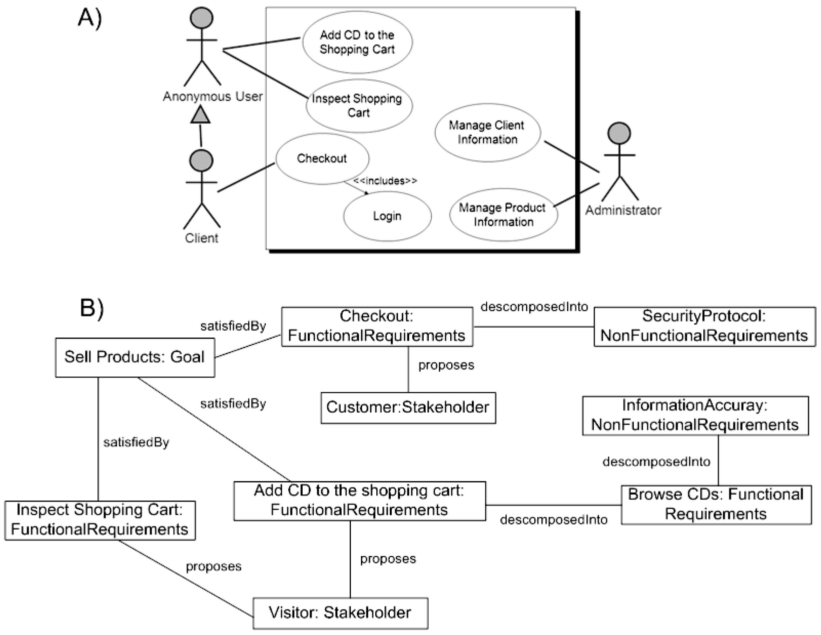

A)
Anonymous User
Client
Administrator
Add CD to the Shopping Cart
Inspect Shopping Cart
Checkout
<<includes>>
Login
Manage Client Information
Manage Product Information
B)
Sell Products: Goal
satisfiedBy
Checkout: FunctionalRequirements
descomposedInto
SecurityProtocol: NonFunctionalRequirements
proposes
Customer:Stakeholder
InformationAccuray: NonFunctionalRequirements
descomposedInto
Browse CDs: Functional Requirements
satisfiedBy
satisfiedBy
Inspect Shopping Cart: FunctionalRequirements
Add CD to the shopping cart: FunctionalRequirements
descomposedInto
proposes
proposes
Visitor: Stakeholder

Fig. 9. Example of requirements model in OOH.

use cases in detail. Thus, we consider criteria FR2 (sequence of operations) and FR3 (abnormal situations) to be not supported. Criterion FR2 (the effect of parameters) is also not supported.

Data requirements are not explicitly supported by the method, therefore there is no technique specifically proposed for specifying them. Furthermore, the techniques proposed to capture requirements do not provide support to describe them. Thus, we consider all the criteria related to data requirements specification to be not supported.

Navigational requirements are not explicitly supported by the method. However, the functional requirement element introduced in the proposed web requirements metamodel is used to define navigational capabilities. For instance, it can be indicated that the user must be able to browse CDs. In addition, the nonfunctional requirement element can be used to indicate that the system should provide information accuracy during this browse activity. However, it cannot be indicated which specific information related to CDs must be published or how users must navigate it. Thus, we consider criterion NR1 (published information) and criterion NR2 (navigation possibilities) to be partially supported. Finally, criteria NR3 (mechanisms to filter information) and NR4 (published functionality) are not supported.

A summary of the requirements specification evaluation of OOH is shown in Table XI.

*4.6.4 Evaluation of the Model-Driven Development Perspective.* Early versions of OOH proposes only a UML use case diagram in order to specify requirements, which is totally based on a visual notation and fits the UML metamodel. The recent works that improve the requirements support present a web requirements metamodel. However, a concrete visual syntax for this metamodel is not provided. Thus, we consider criterion MR1 (metamodel) to be fully supported and criterion M4 (visual syntax) to be partially supported.

Table XI. A Summary of the Requirements Specification Evaluation of OOH

| OOH Requirements Specification Evaluation | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | no | no |
| **Technique** | Use Case diagram | - | - |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** 👍 | **1. Info. Types** ∅ | **1. Published Info.** ✋ |
| | **2. Action Sequence** ∅ | **2. Use Freq.** ∅ | **2. Nav. Possibilities** ✋ |
| | **3. Ab. Situations** ∅ | **3. Access Cap.** ∅ | **3. Info. Filters** ∅ |
| | **4. Param. Effects** ∅ | **4. Entities and Rel.** ∅ | **4. Published Func.** ∅ |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ∅ | |
| | | **6. Data Retention** ∅ | |

Table XII. A Summary of the Model-Driven Evaluation of OOH

| OOH Model-Driven Evaluation | | | | |
|---|---|---|---|---|
| **1 Metamodel** | **2 Mapping Functions** | **3 Tool for mappings** | **4 Visual Syntax** | **5 RE Specification Tool** |
| 👍 | ∅ | ∅ | ✋ | 👍 |

OO-H is supported by the tool Visual Wade. This tool provides complete graphical support to perform the domain and navigational analysis. It also provides support to generate code through PIM-to-code transformations. However, it does not provide support for creating OOH requirements models. In spite of this, tools can be used to create the OOH requirements model if it is considered that this model is completely based on UML, therefore, general modeling tools such as Rational Rose can be used. Thus, we consider criterion MR5 to be fully supported. Finally, mapping functions for deriving OOH conceptual models from requirements are not provided and logically there are no tools for applying mapping functions. Thus, we consider criteria MR2 and MR3 to be not supported. Note, however, that there are some PIM-to-PIM transformations that apply specific measurements to the OOH models, which have been incorporated into the WebSa tool [Meliá and Gomez 2006].

A summary of the model-driven evaluation of OOH is shown in Table XII.

*4.6.5 Main Strengths and Weaknesses.* The main strength of this approach relies on the use of use case diagrams, which is a well-known technique that facilitates its learning and use. In addition, as explained in other approaches, this fact allows analysts to use general modelling tools in order to create the requirements model. Furthermore, recent works of this method have focused on requirements measurement, which provide a valuable mechanism to evaluate requirements in a model-driven way. These works present several PIM-to-PIM transformations that have been implemented in the context of the WebSa tool [Meliá and Gomez 2006]. However, the MDD aspects studied in this survey (requirements specification and transformation of requirements into conceptual models) are handled with less attention.

The main weakness is that requirements specification is basically done through use cases, which are not complemented with other techniques to describe web application requirements in detail. This aspect makes the OOH requirements model insufficient for specifying requirements in the context of MDD of web applications.

### 4.7 OOWS: Object Oriented Web Solutions

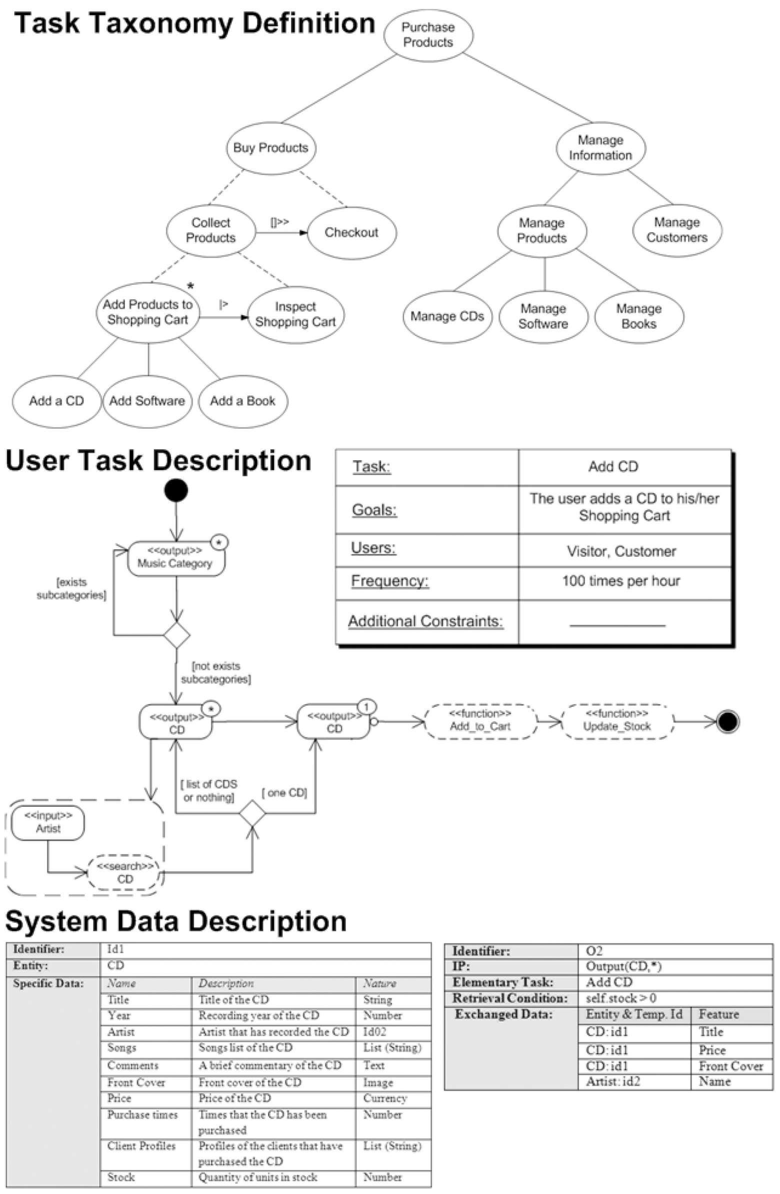
*4.7.1 General Description.* Object-Oriented Web Solutions (OOWS) was presented by Joan Fons and Oscar Pastor in 2000 [Pastor et al. 2000, 2005]. This approach is an extension of the OO-Method approach [Pastor et al. 1997] to support web application development. Unlike the OO-H approach, OOWS is still based on the OO-Method. This makes OOWS one of the few MDD methods that provide support to automatically generate both web application interfaces and fully operative functionality from models. On the one hand, the commercial tool Olivanova[11] automatically generates the functionality of the web application from the models originally proposed by OO-Method. On the other hand, the OOWS Suite [Valverde et al. 2007] automatically generates web application interfaces from the OOWS models and integrates them with the functionality obtained from Olivanova. The development process of this approach is divided into three main phases.

— *Requirements Analysis.* In this phase, the requirements of web applications are specified by means of a model that is based on the concept of task. Early versions of OOWS do not consider the requirements phase. This phase was included after defining some extensions to the method [Valderas 2008].
— *System Specification.* This phase consists in the description of the web application at the conceptual level. To do this, different models are proposed: (1) the *object model* for describing the static structure of the web application, (2) the *functional and dynamic models* for describing the behavior of the web application, (3) the *navigational and presentation models* for describing the web application user interface.
— *Solution Generation*. In this phase, the web application is automatically generated from the models defined in the previous phase. To do this, the tools introduced in this article are used.

*4.7.2 Requirements Model.* Requirements are handled in the requirements analysis phase. This phase is performed in three main steps.

(1) *Definition of a Task Taxonomy*. In this step, the tasks that users must be able to perform by interacting with the web application must be indicated. To do this, a hierarchical task analysis [Shepherd 2001] is performed. After this analysis, a task taxonomy is obtained in which tasks are defined in a hierarchical way. Temporal relationships can also be defined among tasks in order to indicate the plan according to which tasks must be performed.
(2) *Description of User Tasks*. In this step, leaf tasks of the task taxonomy are described in detail. To do this, a characterization task template is defined for each task. In these templates, the following are indicated for each task: the main goal, the users that can perform it, the performance frequency and additional constraints related to either inputs and outputs of the tasks or other aspects such as security, availability, scalability, etc. Next, a technique based on activity diagrams is proposed to describe how each task must be performed. This technique allows the tasks to be described from the interaction between the system and the user that is required to perform them. To do this, the concept of Interaction Point (IP) is introduced. IPs represent a moment during the performance of a task in which the system and users exchange information.
(3) *System Data Description*. Finally, a technique based on information templates is proposed to detail both the information that the system must store and the information exchanged by the system and the user.

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



| | |
|---|---|
| Task: | Add CD |
| Goals: | The user adds a CD to his/her Shopping Cart |
| Users: | Visitor, Customer |
| Frequency: | 100 times per hour |
| Additional Constraints: | ___________ |

| | | | |
|---|---|---|---|
| **Identifier:** | Id1 | | |
| **Entity:** | CD | | |
| **Specific Data:** | *Name* | *Description* | *Nature* |
| | Title | Title of the CD | String |
| | Year | Recording year of the CD | Number |
| | Artist | Artist that has recorded the CD | Id02 |
| | Songs | Songs list of the CD | List (String) |
| | Comments | A brief commentary of the CD | Text |
| | Front Cover | Front cover of the CD | Image |
| | Price | Price of the CD | Currency |
| | Purchase times | Times that the CD has been purchased | Number |
| | Client Profiles | Profiles of the clients that have purchased the CD | List (String) |
| | Stock | Quantity of units in stock | Number |

| | | |
|---|---|---|
| **Identifier:** | O2 | |
| **IP:** | Output(CD,*) | |
| **Elementary Task:** | Add CD | |
| **Retrieval Condition:** | self.stock > 0 | |
| **Exchanged Data:** | Entity & Temp. Id | Feature |
| | CD: id1 | Title |
| | CD: id1 | Price |
| | CD: id1 | Front Cover |
| | Artist: id2 | Name |

Fig. 10. Example of requirements model in OOWS.

Figure 10 shows a partial view of the OOWS requirements model for the running example. The upper portion of this figure shows the task taxonomy that hierarchically indicates the user tasks that describe the running example. Note how two types of task refinement are proposed: structural (depicted by solid lines), which refines complex tasks into simpler ones, and temporal (depicted by dashed lines), which also refines complex tasks into simpler ones but in this case, subtasks are related by a temporal relationship. These relationships are indicated by the temporal operators used in the CTT approach [Paternò et al. 1997].

Below this taxonomy, the description of the task Add CD (note that it is a leaf task) is shown. Its associated characterization template indicates that the goal is for the user to add a CD to its shopping cart. This task can be done by visitors and customers, and as estimation, it should be performed 100 times per hour. This task has no additional constraints. In accordance with its performance description (activity diagram), users add CDs to the shopping cart as follows: the task starts with an IP (nodes depicted with solid lines) in which the system provides the user with a list of Music Categories (note that IPs that represent an exchange of information in which the system provides users with information are stereotyped with the *output* keyword). From the list of music categories, the user can select a category and then the list of CDs that belong to it are shown. From this list, the user can either select a CD and then a description of the CD is shown, or activate a system action (nodes depicted with dashed lines), which searches for the CDs of an artist (system actions that search information are explicitly stereotyped with the keyword *search*). The artist must be introduced by the user by means of an IP. IPs that represent an exchange of information in which the user introduces information into the system are stereotyped with the *input* keyword). If the search returns only one CD, the system provides the user with its description. Otherwise, the system provides the user with a set of CDs. Finally, when the user has obtained a CD description s/he activates the system action, which adds the CD to the shopping cart (system actions that execute functionality are stereotyped with the keyword *function*). After that, the system updates the stock and then the task finishes.

The lower portion of Figure 10 shows the information templates used to specify data requirements. The template on the left side indicates the properties about a CD that the system must store. The template on the right side associates some of these properties to the IP that provides a list of CDs. This last template indicates the data that the system must show for each CD of the list.

*4.7.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified by means of the descriptions associated to leaf tasks. On the one hand, validation of inputs and relationships between inputs and outputs can be defined as constraints in the characterization template. Thus, we consider criteria FR1 and FR5 to be fully supported. On the other hand, the sequence of operations can be described by means of sequences of function system actions in the activity diagrams, so criterion FR2 is considered to be fully supported. Abnormal situations can be defined by associating additional descriptions to tasks. Thus, criterion FR3 is considered to be fully supported. Finally, the input of parameters is specified by Input IPs. However, the description of their effect is not explicitly supported by the model, so criterion FR4 is considered to be not supported.

Data requirements are explicitly supported by the method. They are specified by means of information templates. Templates associated to entities allow us to indicate the properties that the system must store about each entity, the type of each property, and the relationships about entities. This last aspect is indicated by associating the type of an attribute to another entity. Thus, we consider criteria DR1 and DR4 to be fully supported. The data accessibility is described, on the one hand, by connections of IPs and system actions defined in activity diagrams, which indicate the data that can be accessed by system actions, and, on the other hand, by the users associated to each task in the characterization templates, which indicates the users that can access the data that is shown in each task. Thus, we consider criterion DR3 to be fully supported. The frequency with which the data is used is described by means of the *frequency* field that is defined in the task characterization template. If it is indicated how often a task is performed it is implicitly indicated how often the data managed in this task is used. Thus, we consider criterion DR2 to be fully supported.

Table XIII. A Summary of the Requirements Specification Evaluation of OOWS

| **OOWS Requirements Specification Evaluation** | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | yes | yes |
| **Technique** | Task charac. templates Activity diagrams | Information templates | Task taxonomy, Activity diagrams |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** 👍 | **1. Info. Types** 👍 | **1. Published Info.** 👍 |
| | **2. Action Sequence** 👍 | **2. Use Freq.** 👍 | **2. Nav. Possibilities** 👍 |
| | **3. Ab. Situations** 👍 | **3. Access Cap.** 👍 | **3. Info. Filters** 👍 |
| | **4. Param. Effects** ∅ | **4. Entities and Rel.** 👍 | **4. Published Func.** 👍 |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ∅ | |
| | | **6. Data Retention** ∅ | |

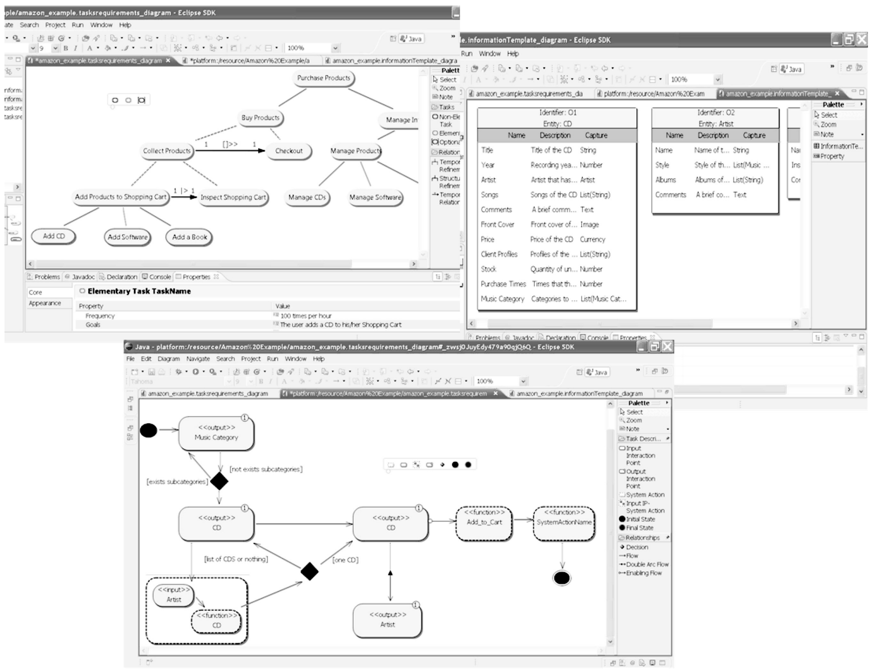
The rest of the criteria associated to the data requirements specification are not supported.

Navigational requirements are explicitly supported by the method. They are described by means of the task taxonomy and the activity diagrams. On the one hand, temporal relationships describe in a general way how users should navigate information (e.g., users must first access the information related to the collection of products before the information related to checkout). Additionally, activity diagrams describe (for each task) how users must navigate information by means of sequences of IPs. Thus, we consider criterion NR2 to be fully supported. The information published is defined by means of the entity associated to each Output IP. Additionally, this information is described in detail by means of the templates associated to these IPs. Thus, criterion NR1 is considered to be fully supported. Mechanisms to filter information are explicitly supported by search systems actions, so we consider criterion NR3 to be fully supported. Finally, the functionally that can be activated during the navigation process is described by the function system actions defined in activity diagrams. Thus, criterion NR4 is considered to be fully supported.

A summary of the requirements specification evaluation of OOWS is shown in Table XIII.

*4.7.4 Evaluation of the Model-Driven Development Perspective.* The techniques proposed by OOWS to specify requirements are based either on visual notations (activity diagrams and task taxonomy) or on textual templates (task characterization and information templates). Thus, we consider criterion MR4 (visual syntax) to be partially supported. Although textual templates are used, they are defined in accordance with a precise structure. This allows the whole OOWS requirements model to fit a precise metamodel in which all the concepts required to specify the supported requirements are explicitly described. This metamodel can be found in Valderas [2008]. Thus, we consider criterion MR1 to be fully supported.

As explained previously, this method is supported by Olivanova and the OOWS suite, which provide full support to create PIM models and generate code from them through PIM-PSM-code and PIM-to-code transfor5mations. With regard to the OOWS requirements model, it can be defined by a CASE tool developed upon the Eclipse platform [Valderas 2008]. This tool can completely create the requirements model, so criterion MR5 is considered to be fully supported. Figure 11 shows a snapshot of this tool. This figure shows how the partial requirements model is being created.



Elementary Task TaskName

| Property | Value |
|---|---|
| Frequency | 100 times per hour |
| Goals | The user adds a CD to his/her Shopping Cart |

Identifier: O1
Entity: CD

| Name | Description | Capture |
|---|---|---|
| Title | Title of the CD | String |
| Year | Recording yea... | Number |
| Artist | Artist that has... | Artist |
| Songs | Songs of the CD | List(String) |
| Comments | A brief comm... | Text |
| Front Cover | Front cover of... | Image |
| Price | Price of the CD | Currency |
| Client Profiles | Profiles of the ... | List(String) |
| Stock | Quantity of un... | Number |
| Purchase Times | Times that th... | Number |
| Music Category | Categories to ... | List(Music Cat... |

Identifier: O2
Entity: Artist

| Name | Description | Capture |
|---|---|---|
| Name | Name of t... | String |
| Style | Style of th... | List(Music ... |
| Albums | Albums of... | List(String) |
| Comments | A brief co... | Text |

Fig. 11. Tool for creating the OOWS requirements model.

Table XIV. A Summary of the Model-Driven Evaluation of OOWS

| OOWSModel-Driven Evaluation | | | | |
|---|---|---|---|---|
| **1 Metamodel** | **2 Mapping Functions** | **3 Tool for mappings** | **4 Visual Syntax** | **5 RE Specification Tool** |
| 👍 | 👍 | 👍 | ✋ | 👍 |

The OOWS requirements model is supported by two CIM-to-PIM transformations [Valderas 2008] that obtain partial versions of the OOWS object model and the OOWS navigational model from the task-based requirements model. These transformations have been defined by means of graph transformations which allow the authors to use general graph transformation tools to apply the model-to-model transformations. In fact, the authors propose a tool-supported strategy that is based on the open source tool AGG[12] in order to automatically apply model-to-model transformations. Thus, we consider criteria MR2 and MR3 to be fully supported. Finally, it is worth noting that this tool-supported strategy for automatically deriving OOWS conceptual model from requirements is a key factor in an approach proposed by the authors to rapidly prototype requirements [Valverde et al. 2007].

A summary of the model-driven evaluation of OOWS is shown in Table XIV.

*4.7.5 Main Strengths and Weaknesses.* The main strength of this approach is that it has been specifically created to specify web application requirements, so it provides new techniques that are oriented to properly support aspects that are encouraged by web applications such as Navigation. In addition, this approach proposes a top-down

[12]http://tfs.cs.tu-berlin.de/agg/

strategy that allows the specification of requirements by starting with a general description (task taxonomy) and refining it to more concrete ones (activity diagram and information template). This is very useful for detecting new requirements when they are not always clear from the beginning. Another interesting characteristic of this approach is that navigational capabilities are described in a global way by means of the task taxonomy. This is different from techniques that are based on use cases, which provide a narrow view of this aspect since navigational requirements are described individually for each use case. Note that navigational requirements are not specified in the use case diagram, they are specified only by the techniques proposed to describe use cases.

Finally, another important strength is that it is fully oriented to support the MDD of web applications. On the one hand, there is a tool that creates the whole OOWS requirements model. On the other hand, there is a tool-supported strategy based on graph transformations and the AGG tool, which automatically derives OOWS conceptual model from requirements.

The main weakness of this approach is the possible complexity of the requirements model. In order to properly specify the requirements of web applications multiple elements need to be created (a taxonomy, a characterization template and an activity diagram for each leaf task, several information templates to specify data requirements). Furthermore, the creation of the task taxonomy that best describes user's needs is not always easy, and it requires some experience in the performance of hierarchical task analysis. Finally, note that activity diagrams have been complemented with several graphical elements that do not belong to the standard UML activity diagrams. This aspect could make these diagrams difficult to understand by untrained people.

### 4.8 W2000

*4.8.1 General Description.* W2000 was developed by Baresi et al. [2001]. This method is presented as an extension and customization of UML with web design concepts borrowed from the Hypermedia Design Model (HDM) [Garzotto et al. 1993]. The development process of this approach is divided into five main phases.

— *Requirements Analysis*. This phase consists in the study of the web application requirements by means of UML use cases.
— *State Evolution Design*. In this phase, analysts must indicate how contents evolve. This phase is not mandatory, but it is required only for applications with complex behaviors.
— *Hypermedia Design*. This phase consists in the design of the web application navigational structure. This phase is based on the use of HDM.
— *Functional Design*. This phase specifies the main user operations of the application.
— *Visibility Design*. In this phase, different perspectives of the web application are defined for each different type of user.

*4.8.2 Requirements Model.* Requirements are handled in the requirements analysis phase. The requirement analysis proposed by W2000 is performed from two main activities: functional requirements analysis and navigational requirements analysis. After identifying the different types of user that can interact with the web application, the functional requirements analysis identifies the main user operations while the navigational requirements analysis indicates the main information and navigation structures needed by the different users.

In order to perform both activities, W2000 proposes the use of UML use case diagrams. Two types of use cases are proposed: functional use cases and navigational use cases. The functional use case diagram is a typical use case diagram that identifies the

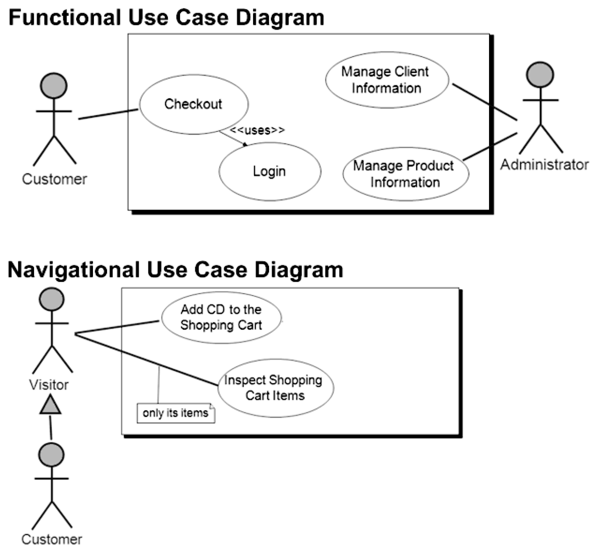

Functional Use Case Diagram
Customer
Checkout
<<uses>>
Login
Manage Client Information
Manage Product Information
Administrator
Navigational Use Case Diagram
Visitor
Add CD to the Shopping Cart
Inspect Shopping Cart Items
only its items
Customer

Fig. 12. Example of requirements specification in W2000.

main functionalities and associates them with types of user. The navigational use case diagram indicates the navigation capabilities associated to each type of user. These navigational capabilities can be constrained with accessibility conditions.

Figure 12 shows a partial version of the W2000 use case diagrams for the Amazon example. The upper portion of this figure shows a functional use case diagram that indentifies functionalities such as login or management of product information. These functionalities are associated to the Customer and Administrator types of user, respectively. The lower portion of this figure shows a navigational use case diagram that identifies navigational capabilities such as *add CD to the shopping cart* or *inspect shopping cart items*. Note how this last capability presents an accessibility constraint. Visitors can inspect only the items that they have added to the shopping cart.

*4.8.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified by means of the functional use case diagram. As happens with other approaches, use cases fully support the criteria FR1 (validation of inputs) and FR5 (relation of outputs and inputs) by means of the pre- and post conditions that may be associated to use cases. This approach, however, does not propose any technique for describing use cases in detail. Thus, we consider criteria FR2 (sequence of operations) and FR3 (abnormal situations) to be not supported. Criterion FR2 (the effect of parameters) is also not supported.

Data requirements are not explicitly supported by the method, therefore there is no technique specifically proposed for specifying them. Furthermore, the technique proposed to specify requirements (use case diagrams) do not provide support to describe them. Thus, we consider all the criteria related to data requirements specification to be not supported.

Navigational requirements are explicitly supported by the method. They are described by means of navigational use case diagram. However, this special use case diagram identifies only navigational capabilities. For instance, it can be indicated that the user must be able to search CDs, but it cannot be indicated which specific information related to CDs must be published or how users must navigate it. Thus, we consider criterion NR1 (published information) to be partially supported and criterion

Table XV. A Summary of the Requirements Specification Evaluation of W2000

| W2000 Requirements Specification Evaluation | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | no | yes |
| **Technique** | Functional use case diagram | - | Navigational use case diagram |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** 👍 | **1. Info. Types** ∅ | **1. Published Info.** ✋ |
| | **2. Action Sequence** ∅ | **2. Use Freq.** ∅ | **2. Nav. Possibilities** ∅ |
| | **3. Ab. Situations** ∅ | **3. Access Cap.** ∅ | **3. Info. Filters** ∅ |
| | **4. Param. Effects** ∅ | **4. Entities and Rel.** ∅ | **4. Published Func.** ∅ |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** ∅ | |
| | | **6. Data Retention** ∅ | |

Table XVI. A Summary of the Model-Driven Evaluation of W2000

| W2000 Model-Driven Evaluation | | | | |
|---|---|---|---|---|
| **1 Metamodel** | **2 Mapping Functions** | **3 Tool for mappings** | **4 Visual Syntax Tool** | **5 RE Specification** |
| 👍 | ∅ | ∅ | 👍 | ✋ |

NR2 (navigation possibilities) to be not supported. Finally, criteria NR3 (mechanisms to filter information) and NR4 (published functionality) are not supported.

A summary of the requirements specification evaluation of W2000 is shown in Table XV.

*4.8.4 Evaluation of the Model-Driven Development Perspective.* The W2000 requirements model is fully defined by using the visual notation proposed for the UML use case diagram. In addition, elements defined in this model are all used according to the semantics described in a metamodel. Therefore, we consider criteria MR4 and MR1 to be fully supported. There is not tool that supports the creation of this requirements model. However, since this requirements model is extensively based on the UML use case diagram, general modelling tools such as Rational Rose can be used to create part of it. Thus, we consider criterion MR5 to be partially supported. Mapping functions for deriving conceptual models from requirements are not provided, so there are no tools for applying mapping functions. Therefore, criteria MR2 and MR3 are considered to be not supported.

A summary of the model-driven evaluation of W2000 is shown in Table XVI.

*4.8.5 Strengths and Weaknesses.* The main strength of this approach is that it proposes a well-known technique for specifying requirements such as use cases, which are extended with new mechanisms (a special use case diagram for capturing navigational requirements) in order to support navigational requirements. This solution allows analysts to identify functional and navigational capabilities by using a technique that can be partially supported by general modeling tools. This facilitates its use in MDD processes.

Its main drawback is that use cases are not complemented with techniques that allow analysts to describe requirements in detail. Analysts can identify functional and navigational capabilities, but they cannot describe how these capabilities must be supported by the web application such as the UIDs of OOHDM or the activity diagrams of UWE do.

## 4.9 NDT: Navigational Development Techniques

*4.9.1 General Description.* NDT was presented by Escalona et al. [2004] and Escalona and Aragón [2008]. NDT is a methodological process for web application development that is focused on the requirements and analysis phases. It proposes the intensive use of textual templates in the requirements phase and the systematic derivation of analysis models from these templates. This approach proposes the use of prototypes to validate requirements. The NDT Suite[13] has been developed to support this approach. The development process of this approach is divided into three main stages.

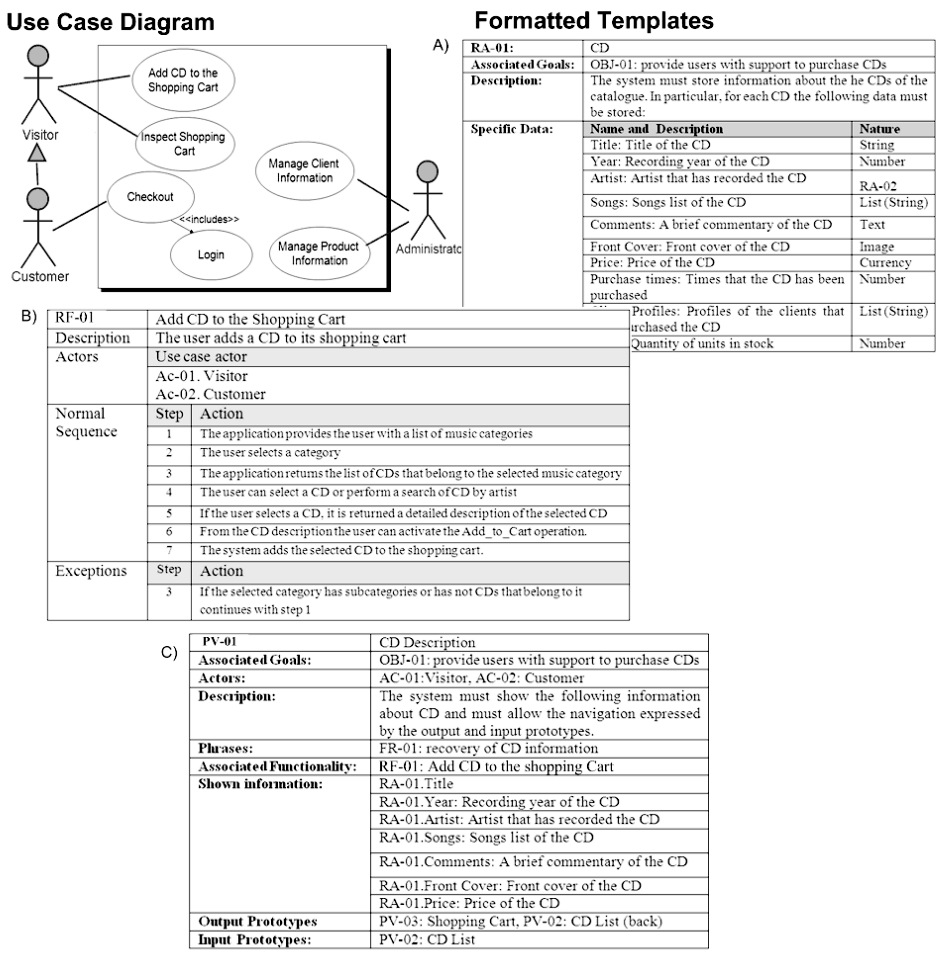
— *Requirements Treatment*. In this phase, the web application requirements are collected and described.
— *Analysis*. In this phase, analysis models are systematically derived from the requirements specification. These analysis models are the conceptual model and the navigational model.
— *Prototyping*. This phase consists in the development of web application prototypes from analysis models. These prototypes are used to validate requirements.

*4.9.2 Requirements Model.* NDT is an approach specifically created for handling web application requirements. This approach fits together with UWE the UML profile for web requirements (WebRE) [Escalona and Koch 2007].

In order to specify requirements, NDT proposes the use of uses case diagrams and several types of formatted templates. NDT classifies requirements into the following types: storage information, actor, functional, interaction and nonfunctional requirements. For each type, NDT defines a special template, that is, a table with specific textual fields that are completed by the development team during the requirements elicitation phase.

Figure 13 shows a partial version of the NDT requirements model for the running example. On the one hand, the use case diagram is shown on the upper left side of the figure. On the other hand, three formatted templates are shown: template A is an example of template for storage information requirements. This template indicates the information that must be stored about a CD by means of a list of features. Template A is an example of a template for functional requirements. Specifically, this template defines the use case Add CD to the Shopping Cart. Template C is an example of a template for interaction requirements. These templates implicitly describe the navigation structure of the web application. They are based on the concept of Visualization Prototype, which represents a chunk of information that must be shown to users. Each of these chunks is defined as set of features specified in a storage information template (e.g., the field shown information in template C is defined from features specified in template A). In addition, visualization prototypes can be associated to functional templates to indicate that users must be able to activate this functionality when they access the chunk of information (e.g., a user can activate the functionality Add CD when s/he accesses the information chunk defined in template C). Finally, visualization prototypes may have output prototypes, which indicate the prototypes than can be accessed from the current one, and input prototypes, which indicate the prototypes from which the current one can be accessed. This way of relating prototypes indicates how users can navigate information.

[13]http://www.iwt2.org/ndt-suite.php



A)

| | | |
|---|---|---|
| **RA-01:** | CD | |
| **Associated Goals:** | OBJ-01: provide users with support to purchase CDs | |
| **Description:** | The system must store information about the he CDs of the catalogue. In particular, for each CD the following data must be stored: | |
| **Specific Data:** | **Name and Description** | **Nature** |
| | Title: Title of the CD | String |
| | Year: Recording year of the CD | Number |
| | Artist: Artist that has recorded the CD | RA-02 |
| | Songs: Songs list of the CD | List (String) |
| | Comments: A brief commentary of the CD | Text |
| | Front Cover: Front cover of the CD | Image |
| | Price: Price of the CD | Currency |
| | Purchase times: Times that the CD has been purchased | Number |
| | Profiles: Profiles of the clients that urchased the CD | List (String) |
| | Quantity of units in stock | Number |

B)

| | | |
|---|---|---|
| RF-01 | Add CD to the Shopping Cart | |
| Description | The user adds a CD to its shopping cart | |
| Actors | Use case actor | |
| | Ac-01. Visitor<br>Ac-02. Customer | |
| Normal Sequence | Step | Action |
| | 1 | The application provides the user with a list of music categories |
| | 2 | The user selects a category |
| | 3 | The application retums the list of CDs that belong to the selected music category |
| | 4 | The user can select a CD or perform a search of CD by artist |
| | 5 | If the user selects a CD, it is returned a detailed description of the selected CD |
| | 6 | From the CD description the user can activate the Add_to_Cart operation. |
| | 7 | The system adds the selected CD to the shopping cart. |
| Exceptions | Step | Action |
| | 3 | If the selected category has subcategories or has not CDs that belong to it continues with step 1 |

C)

| | |
|---|---|
| **PV-01** | CD Description |
| **Associated Goals:** | OBJ-01: provide users with support to purchase CDs |
| **Actors:** | AC-01:Visitor, AC-02: Customer |
| **Description:** | The system must show the following information about CD and must allow the navigation expressed by the output and input prototypes. |
| **Phrases:** | FR-01: recovery of CD information |
| **Associated Functionality:** | RF-01: Add CD to the shopping Cart |
| **Shown information:** | RA-01.Title |
| | RA-01.Year: Recording year of the CD |
| | RA-01.Artist: Artist that has recorded the CD |
| | RA-01.Songs: Songs list of the CD |
| | RA-01.Comments: A brief commentary of the CD |
| | RA-01.Front Cover: Front cover of the CD |
| | RA-01.Price: Price of the CD |
| **Output Prototypes** | PV-03: Shopping Cart, PV-02: CD List (back) |
| **Input Prototypes:** | PV-02: CD List |

Fig. 13. Example of requirements specification in NDT.

*4.9.3 Evaluation of the Requirements Specification Perspective.* Functional requirements are explicitly supported by the method. They are specified by means of the functional use case diagram and the formatted templates for requirements of this type. The possibility of associating pre- and post conditions to use cases allows the method to fully support criteria FR1 (validation of inputs) and FR5 (relation of outputs and inputs). The template associated to use cases allow analysts to describe the sequences of actions performed by the user and the system, therefore we consider criterion FR2 to be fully supported. In addition, exception situations can be defined in these templates, so criterion FR3 is also considered to be fully supported. Finally, there is no element in use case descriptions that explicitly supports the description of effect parameters. However, the flexibility provided by the textual descriptions of actions can be used to describe it in natural language. Thus, we consider criterion FR4 to be partially supported.

Data requirements are explicitly supported by the method. They are mainly described by means of formatted templates for storage information. These templates indicate which features the system must store about each significant concept. In addition, the type of each feature can also be indicated. This aspect is also used to define

Table XVII. A Summary of the Requirements Specification Evaluation of NDT

| **NDT Requirements Specification Evaluation** | | | |
|---|---|---|---|
| | **Functional Req.** | **Data Req.** | **Navigational Req.** |
| **Explicit Sup.** | yes | yes | yes |
| **Technique** | Use case diagram, functional templates | Storage information templates | Interaction templates |
| **Criteria:** | **FR** | **DR** | **NR** |
| | **1. Input Validity** 👍 | **1. Info. Types** 👍 | **1. Published Info.** 👍 |
| | **2. Action Sequence** 👍 | **2. Use Freq.** ∅ | **2. Nav. Possibilities** 👍 |
| | **3. Ab. Situations** 👍 | **3. Access Cap.** ✋ | **3. Info. Filters** 👍 |
| | **4. Param. Effects** ✋ | **4. Entities and Rel.** 👍 | **4. Published Func.** 👍 |
| | **5. Out. & In. Rel.** 👍 | **5. Int. Constraints** 👍 | |
| | | **6. Data Retention** 👍 | |

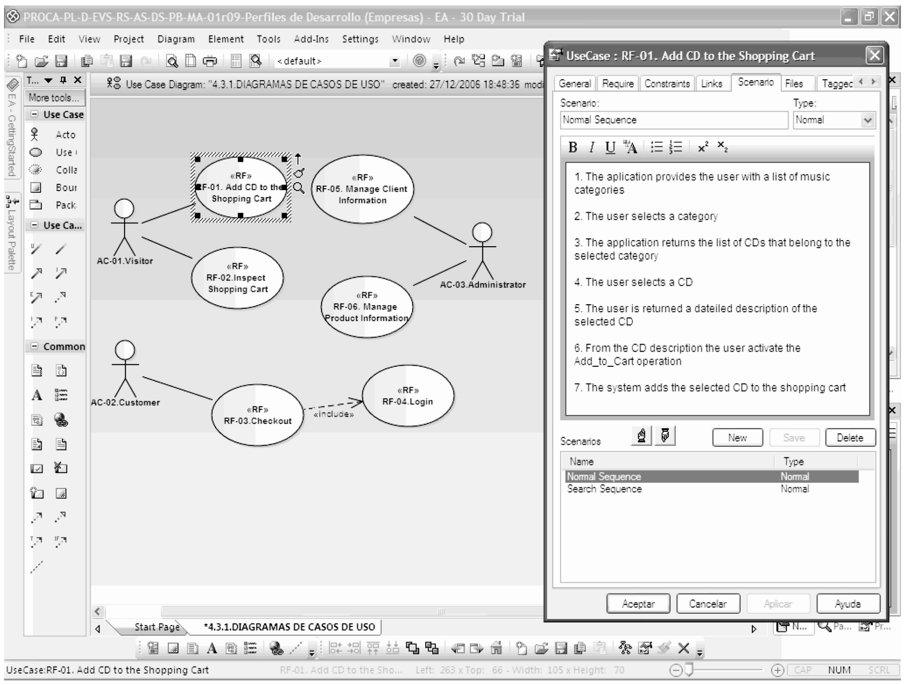
relationships between concepts by indicating that the type of a feature is another concept. Thus, we consider criteria DR1 (data types) and DR4 (entities and relationships) to be fully supported. Furthermore, although it is not shown in Figure 13, the NDT suite (which is further presented) allows us to define a validity period to the different data defined in the storage information templates. In the same way, this tool also allows the association of integrity constraints to storage information templates. Thus, we consider criteria DR5 (integrity constraints) and DR6 (Data Retention) to be fully supported. Finally, the formatted templates for interaction requirements allow us to indicate the users that can access the information. However it is not clear how access by system operations is represented. In this case, we consider criterion DR3 (accessing capabilities) to be partially supported. The rest of the criteria associated to data requirements are not supported.

Navigational requirements are explicitly supported by the method. They are described by means of interaction formatted templates. This type of template is mainly used to indicate which information the system must show. In addition, the possibility of linking them by using output and input relations describes how users navigate the information. Thus, we consider criteria NR1 (published information) and NR2 (navigation possibilities) to be fully supported. These templates include a field that is explicitly created to indicate the functionality provided along with the information. Therefore, we consider criterion NR4 to be fully supported. Finally, mechanisms to filter information can be defined by means of interaction templates. These templates can be used to describe the queries that users can perform over the stored data by interacting with the web application interface. Thus, we consider criterion NR2 to be fully supported.

A summary of the requirements specification evaluation of NDT is shown in Table XVII.

*4.9.4 Evaluation of the Model-Driven Development Perspective.* The NDT requirements model is defined by means of a use case diagram and a set of formatted textual templates. The creation of this model is fully supported by the NDT Suite. NDT suite is a profile for the commercial modelling tool Enterprise Architect.[14] This suite incorporates different extensions that allows the creation of the full NDT requirements model, so we consider criterion MR5 to be fully supported. Figure 14 shows a snapshot of this tool.

[14]http://www.sparxsystems.com/


PROCA-PL-D-EVS-RS-AS-DS-PB-MA-01r09-Perfiles de Desarrollo (Empresas) - EA - 30 Day Trial
File Edit View Project Diagram Element Tools Add-Ins Settings Window Help
Use Case Diagram: "4.3.1.DIAGRAMAS DE CASOS DE USO" created: 27/12/2006 18:48:36
«RF» RF-01. Add CD to the Shopping Cart
«RF» RF-05. Manage Client Information
«RF» RF-02.Inspect Shopping Cart
«RF» RF-06. Manage Product Information
«RF» RF-03.Checkout
«include»
«RF» RF-04.Login
AC-01.Visitor
AC-03.Administrator
AC-02.Customer
Start Page
*4.3.1.DIAGRAMAS DE CASOS DE USO
UseCase:RF-01. Add CD to the Shopping Cart
UseCase : RF-01. Add CD to the Shopping Cart
General Require Constraints Links Scenario Files Tagged
New Save Delete
Aceptar Cancelar Aplicar Ayuda

Scenario: Normal Sequence
Type: Normal

1. The aplication provides the user with a list of music categories

2. The user selects a category

3. The application returns the list of CDs that belong to the selected category

4. The user selects a CD

5. The user is returned a dateiled description of the selected CD

6. From the CD description the user activate the Add_to_Cart operation

7. The system adds the selected CD to the shopping cart

Scenarios

| Name | Type |
|---|---|
| Normal Sequence | Normal |
| Search Sequence | Normal |

Fig. 14. Snapshot of the NDT suite.

Table XVIII. A Summary of the Model-Driven Evaluation of NDT

| W2000 Model-Driven Evaluation | | | | |
|---|---|---|---|---|
| 1 Metamodel | 2 Mapping Functions | 3 Tool for mappings | 4 Visual Syntax | 5 RE Specification Tool |
| 👍 | 👍 | 👍 | ✋ | 👍 |

With regard to the visual syntax, the NDT suite provides a visual notation for the use case diagrams and a visual representation (as classes) for the formatted templates. However, many part of the specification of these templates must be textually defined. Thus, we consider criterion MR4 (concrete visual syntax) to be partially supported. This model fits the WebRE metamodel [Escalona and Koch 2007], so we consider criterion MR1 to be fully supported.

Finally, the NDT Suite incorporates a module that implements a set of transformations specified in QVT by means of as Enterprise Architect templates. This module allows to automatically obtaining NDT analysis models from requirements (CIM-to-PIM Transformations). Thus, we consider criteria MR2 and MR3 to be fully supported. In addition, other modules are included to support activities related to the validation and verification of both requirements and derived models. A module for generating prototypes is also provided. Finally, it is worth noting that the NDT suite is currently being used by several Spanish private companies and government institutions.

A summary of the model-driven evaluation of NDT is shown in Table XVIII.

*4.9.5 Strengths and Weakness.* The main strength of this approach is that it provides mechanisms to properly describe the requirements of a web application. This approach supports the specification of the three types of requirements considered in this survey:

data (storage information), functional, and navigational (interaction) requirements as well as other types of requirements such as user types. In addition, note that although textual descriptions are proposed, they are based on structured templates that facilitate the automation of MDD process. In fact, NDT is supported by a set of tools that is currently being used in the development of real web applications.

The main weakness of this approach is that the use of textual templates is not always easy. These templates are very suitable for describing requirements such as data or functional requirements. However, in complex web applications, a lot of different possibilities of navigation may have to be considered. To specify all of them by using just textual templates could overload the requirements specification. In the same way, the use of textual templates instead of other diagrammatic tools such as the UIDs of OOHDM or the activity diagrams of UWE makes it difficult to obtain a vision of the navigational aspect of a web application from the requirements model. Finally, the number of textual templates that must be defined may result difficult to manage and maintain in large web applications. As the authors of NDT themselves state [Escalona and Koch 2007], templates are not easy to complete as they require intensive interviews.

## 5. SUMMARY AND LESSONS LEARNED

This work presents a survey that analyzes how functional, data and navigational requirements are considered by the most relevant model-driven web engineering methods. In particular, we focus on those methods that have explicitly faced this problem in research publications. In order to study these methods, we consider two perspectives in which different evaluation criteria have been defined regarding to the specification of requirements and MDD.

With regard to the specification of requirements, all the analyzed approaches consider functional requirements in the specification activity. In addition, almost all the criteria associated to requirements of this type are fully supported. Only the effect of parameters is sometimes not supported.

Data and navigational requirements are not always considered. Data requirements are considered only by WSDM, WebML, OOWS and NDT. However, WSDM do not provide explicit support to any criteria associated to requirements of this type. WebML supports the description of entities and relationships, and accessing capabilities. These two criteria are also supported by OOWS and NDT. In addition, OOWS support the criteria related to information types and the frequency of use, and NDT supports the criteria related to information types, integrity constraints, and data retention.

Navigational Requirements are considered by OOHDM, WSDM, UWE, WebML, OOH, OOWS, W2000 and NDT. OOHDM, UWE, OOWS and NDT provide fully support to all the criteria studied in this survey, that is. they provide mechanisms to describe published information, navigation capabilities, information filters, and published functionality. WSDM and WebML also support all these criteria although some of them are partially supported. W2000 allows us to indicate only the published information. Finally, OOH partially supports the description of published information and navigation capabilities.

As far as the proposed techniques are concerned, some approaches such as OOHDM, UWE, W2000, WebML, OOH, or NDT propose the use of Use Cases in the requirements specification. However, these approaches suggest additional techniques for complement use cases in order to support data and/or navigational requirements. OOHDM and UWE provide graphical diagrams to describe use cases in detail, which are focused on facilitating the specification of navigational requirements. W2000 introduces two types of use case models in order to identify functional and navigational capabilities. However, techniques for describing use cases in detail are not provided. WebML and

Table XIX. A Summary of the Requirements Specification Perspective Analysis

| Method | Functional Req. | Data Req. | Navigational Req. |
|---|---|---|---|
| OOHDM | use cases, textual templates | - | UIDs |
| WSDM | natural language, task diagram | natural language | natural language task diagram |
| SOHDM | scope diagram, SACs | - | - |
| UWE | web process use cases, stereotyped activity diagrams | - | navigation use cases, stereotyped activity diagrams |
| WebML | user templates, activity diagrams, use cases | data dictionary | site specification templates |
| OOH | use cases | - | web requirements metamodel (WE-RMExt) |
| OOWS | task characterization templates, activity diagrams | information templates | task taxonomy, activity diagrams |
| W2000 | functional use cases | - | navigational use cases |
| NDT | use cases, formatted templates | formatted templates | formatted templates |

NDT extend use cases with textual templates that allow us to specify both data and navigational requirements. Finally, OOH improves the capture of web requirements with a metamodel that clearly describes the aspects that need to be considered in a web application requirements specification.

WSDM, SOHDM, and OOWS provide requirements specification techniques for web applications that are not based on use cases. WSDM proposes user taxonomies that are complemented with informal description created in natural language. In addition, task diagrams are also defined in order to identify the required data and functionality. SOHDM is based on scenarios and offers a proprietary notation to describe them. However, this notation is mainly focused on the specification of functional requirements. As far as OOWS is concerned, it provides a requirements model based on the concept of task. This model allows us to specify functional, data and navigational requirements of web applications. To do this, a task taxonomy is first created. Next, a technique based on activity diagrams is introduced to describe user tasks from the interaction between the system and the user. Finally, textual templates are proposed to describe the data that the system must store.

A summary of the requirements specification perspective analysis is shown in Table XIX.

With regard to MDD support, all the approaches introduce techniques that include a concrete syntax that is fully or partially visual. In addition, most of these techniques are mainly based on structured notions that can fit a precise metamodel.

As far as tool support is concerned, few approaches provide it. Only UWE, OOWS and NDT provide proprietary tools for creating the proposed requirements model. The construction of the UWE requirements model is partially supported by the ArgoUWE and MagicUWE tools. These tools offer graphical support to create use case diagrams. However, the construction of the stereotyped activity diagrams is not supported (although authors are currently working on it). The creation of the OOWS requirements model is fully supported by a RE tool that is based on the eclipse platform. This tool allows every element of the model to be created. NDT is supported by the NDT Suite. This tool provides graphical support to create the different elements of the NDT requirements model.

WebML, OOHDM, and OOH are supported by WebRatio, HyperDE, and Visual Wade, respectively. However, these tools are focused on providing graphical interfaces

Table XX. Summary of the MDD Perspective Analysis

| Method | Visual Syn. and MetaM. | Req. Spec. Tool | Deriv. of Conceptual Models |
|---|---|---|---|
| OOHDM | ok | - | *Transformations* *specified with:* textual guidelines *applied:* manually (no tool) |
| WSDM | ok | - | - |
| SOHDM | ok | - | - |
| UWE | ok | ArgoUWE, MagicUWE | *Transformations* *specified with:* QVT *applied:* manually (no tool) |
| WebML | ok | - - | *Transformations* *specified in:* textual guidelines *applied:* manually (no tool) |
| OOH | ok | - | - |
| OOWS | ok | RE tool based on Eclipse | *Transformations* *specified with:* graph transformations *applied:* automatically (AGG tool) *implemented as*: AGG graph transformations |
| W2000 | ok | - | - |
| NDT | ok | NDT Suite | *Transformations* *specified with::* QVT *applied:* automatically (NDT Suite) *implemented as*: Enterprise Architect templates |

for the definition of web conceptual models (PIM models) and the generation of code from these models (PIM-to-code transformations) and provide little support to the specification of requirements. OOH is also supported by the WebSA tool in the application of model-to-model transformations with model measurements purposes.

Finally, in most of the analyzed methods the way in which requirements are translated to a conceptual model (CIM-to-PIM transformation) depends on the experience and skills of the analyst. Mechanisms for supporting the derivation of web application conceptual models from requirements models are provided by only five approaches: OOHDM, UWE, WebML, OOWS, and NDT.

OOHDM and WebML present guidelines to derive the corresponding conceptual models from requirements. However, they are described in natural language, which may be ambiguous. Furthermore, no tools are provided to help analysts to apply the proposed guidelines. UWE proposes together with NDT a common web requirements metamodel and a set of model-to-model transformations. These transformations take a description based on this metamodel as source and obtain conceptual models of web applications. These model-to-model transformations are specified in the QVT Relations standard. However, the authors do not propose tools for applying them. NDT has developed an implementation of the QVT transformations by means of templates of the Enterprise Architect tool, which are automatically applied to transform the NDT requirements model into a conceptual model. UWE is currently working on an implementation of the CIM-to-PIM transformations in ATL. Finally, OOWS present two CIM-to-PIM transformations specified and implemented by means of graph transformations. This allows OOWS to provide a tool-supported strategy based on the AGG tool in order to automatically apply the model-to-model transformations to obtain OOWS conceptual models from the requirements model.

A summary of the MDD perspective analysis is shown in Table XX.

To conclude this section, we can claim that.

— there are few techniques that are specifically defined for the specification of web application requirements. Furthermore, the newly created techniques can still be considered young techniques and a lot of research work must be performed in order to improve them. In particular, we should guide our efforts to improve the specification of data and navigational requirements and the integration of these specifications with functional requirements.
— there are few tools that support the construction of the proposed web application requirements models. Thus, almost all the models proposed by the different methods must be manually created.
— once requirements are specified, there is little support for allowing the systematic or automatic derivation of the conceptual model that properly satisfies the requirements model. Logically, few tools are provided to support this aspect.

## 6. RELATED SURVEYS

In this section, we present other existing surveys that are related to the development of web applications.

The surveys done by the Cutter consortium [Epner 2000] and McDolland and Welland [2001] focus on interviewing developers and stakeholders to identify the main problems of web development projects. Other surveys conducted by GeorgiaTech's Graphics, Visualization, & Usability Center [GVU 1997] and NUA [1998] have focused on understanding the profile of the users of the Internet and the Web. Lang and Fitzgerald [2007] interview 438 organizations of Ireland that were involved in the development of several web projects between 2002 and 2005. In particular, this study analyses the design processes that were followed and the underlying logic and forces which shape those processes. Other survey related with the analysis of web design processes is the one done by Vora [1998] in which 138 people involved in the development of web applications were interviewed to know aspects such as the web designer profile or the key components of the web designer's "toolkit". Newman and Landay [1999] interview eleven web designers from five companies to conclude that sketches, prototypes, written documents, presentations, and finished web sites are artifacts usually used in the design process. In the study presented by Lowe [2003], developers of several companies are interviewed in order to know how they interact with clients and the processes that they follow for understanding client needs. Taylor et al. [2002] analyze the techniques and standards used for web site developments within 25 UK organizations.

From a model-driven perspective, we can find several surveys that focus on analyzing conceptual modelling activities in the development of web applications. Schwinger and Koch [2006] present a study of eleven web application development methods. In this study, the proposed modeling approaches as well as the capabilities of code generation are studied. Preciado et al. [2005] study fifteen MDD methods from the perspective of rich internet application modeling. This study takes into account aspects such as multimedia modelling, personalization modelling, and tool support. Koch [1999] studies eleven MDD methods are studied by focussing on its development process. This survey indicates the stages included in each development process as well as the modelling techniques and notations used. Tool support is also studied. The survey presented by Woukeu et al. [2003] also focuses on the development process of several MDD methods by studying their phases and the modelling techniques used.

Finally, Escalona et al. [2004] present a survey in which requirements techniques of MDD methods are studied. This survey presents a general overview of the development process of each method and explains which phase handles requirements as well as which techniques are used. We differ from this survey in different aspects: (1) We provide a more critical view. We not only explain the technique used but also indicate its main strengths and weakness. (2) In addition, we demonstrate the applicability of the proposed requirements specification techniques by applying them in a running example. (3) Finally, in addition to requirements techniques, we also study the mechanisms proposed by each method to support the use of these techniques in a MDD environment.

## 7. CONCLUSIONS

This article has presented the state-of-the-art in requirements specification techniques used in MDD of web applications. More specifically, from a preliminary study of MDD methods, we have selected ten that proposed techniques for handling requirements and we have studied them in detail. For each of these methods, we have studied two main aspects: the techniques proposed to specify requirements and the support provided for MDD. In order to facilitate the comparison of these approaches, we have applied them to the same running example. Finally, a report of lessons learned as well as a comparison with other existing surveys that are related to MDD of web applications have been added as a complement to this comprehensive survey.

## ACKNOWLEDGMENTS

We want to express our gratitude to the authors of the different methods that have been analyzed in this survey for their helpful and valuable suggestions and comments. In particular, we want to thanks Gustavo Rossi, Olga de Troyer, Nora Koch, Piero Fraternali, Marco Brambilla, Cristina Cachero, Luciano Baresi, Maria José Escalona, Geert-Jan Houben, and Kees van der Sluijs.

## REFERENCES

ACERBIS, R., BONGIO, A., BRAMBILLA, M., TISI, M., CERI, S., AND TOSETTI, E. 2007. Developing eBusiness Solutions with a model driven approach: The case of Acer EMEA. In *Proceedings of the 7th International conference on web Engineering* 539–544.

ASHWORTH, C. M. 1989. Using SSADM to specify requirements. In *Proceedings of the IEE Colloquium on Requirements Capture and Specification for Critical Systems*. AIMS Systems, London.

ATKINSON, C. AND KUHNE, T. 2003. Model-driven development: A metamodeling foundation. *IEEE Softw. 20*, 5, 36–41.

BARESI, L., GARZOTTO, F., AND PAOLINI, P. 2001. Extending UML for modeling Web applications. In *Proceedings of the 34th Hawaii International Conference on System Sciences*.

BRAMBILLA, M. 2003. Extending hypertext conceptual models with process-oriented primitives. In *Proceedings of the 22nd International Conference on Conceptual Modeling*. 246–262.

BRAMBILLA, M., CERI, S., FRATERNALI, P., AND MANOLESCU, I. 2006. Process modeling in Web applications. *ACM Trans. Softw. Engin. Method. (ACM TOSEM) 15* , 4, 360–409.

BURDMAN, J. 1999. *Collaborative Web Development*. Addison-Wesley, Reading, MA.

CACHERO, C. 2003. Una extensión a los métodos OO para el modelado y generación automática de interfaces hipermediales. PhD dissertation (in Spanish). Universidad de Alicante. Alicante, Spain.

CACHERO, C., AND KOCH, N. 2002. Navigation analysis and navigation design in OO-H and UWE. Tech. rep. Universidad de Alicante, Spain

CERI, S., FRATERNALI, P., AND BONGIO, A. 2000. Web modeling language (WebML): A modeling language for designing Web sites. *Comput. Netw. 33*, 1-6, 137–157.

CERI, S., FRATERNALI, P., BONGIO, A., BRAMBILLA M., COMAI S., AND MATERA M. 2003. *Designing Data-Intensive Web Applications*. Morgan Kaufman.

CONALLEN, J. 1999. *Building Web Applications with UML*. Addison Wesley.

CONKLIN, J. 1987. Hypertext: An introduction and survey. *IEEE Comput. 20*, 9, 17–41.

COWAN, D. D. AND LUCENA, C. J. P. 1995. Abstract data views, An interface concept to enhance design for reuse. *IEEE Trans. Softw. Engin. 21*, 3.

DEMARCO, T. 1979. *Structured Analysis and System Specification*. Yourdon Press.

De Troyer, O. and Leune, C. 1998. WSDM: A user-centered design method for web sites. Computer networks and ISDN systems, In *Proceedings of the 7th International World Wide web Conference*, Elsevier, 85–94.

De Troyer, O., Casteleyn, S., and Plessers, P. 2008. WSDM: Web semantics design method. *Web Engineering: Modelling and Implementing Web Applications*. Human-Computer Interaction Book Series, Springer. 303–351.

England, E. and Finney, A. 1999. *Managing Multimedia: Project Management for Interactive Media*. Addison-Wesley, Reading, MA.

Epner, M. 2000. Poor project management number-one problem of outsourced e-projects. Research Briefs, Cutter Consortium.

Escalona, M. J. and Koch, N. 2004. Requirements engineering for Web applications: A comparative study. *J. Web Engin.*, Rinton Press, *2*, 3, 193–212.

Escalona, M. J. and Koch, N. 2007. Metamodelling the requirements of Web systems. Lecture Notes in Bussiness Information Process, vol. 1, Springer Verlag, 267–288.

Escalona, M. J. and Aragón, G. 2008. NDT: A model driven approach for Web requirements. *IEEE Trans. Softw. Engin. 34*, 3, 377–390.

Escalona, M. J., Mejías, M., and Torres, J. 2004. Developing systems with NDT and NDT-tool. In *Proceedings of the 13th International Conference on Information Systems Development*. 149–159.

Fons, J., Pelechano, V., Albert, M., and Pastor, O. 2003a. Development of Web applications from web enhanced conceptual schemas. In *Proceedings of the 22th International Conference on Conceptual Modeling (ER'03).* Lecture Notes in Computer Science, v. 2813.

Fons, J., Valderas, P., Ruiz, M., Rojas, G., and Pastor, O. 2003b. OOWS: A method to develop web applications from Web-oriented conceptual models. In *Proceedings of the 3rd International Workshop on Web Oriented Software Technology*. 65–70.

Frasincar, F., Houben, G. J., and Vdovjak, R. 2002. Specification framework for engineering adaptive Web applications. In *Proceedings of the 11th International World Wide Web Conference*. Hawaii, USA. http://www2002.org/CDROM/alternate/682/.

Garrigos, I., Gomez, J., Barna, P., and Houben, G. J. 2005. A reusable personalization model in web application design. In *Proceedings of the 2nd Workshop on Web Information Systems Modelling* (In Conjunction with ICWE'05).

Garzotto, F., Paolini, P., and Schwabe, D. 1993. HDM - A model-based approach to hypertext application design. *ACM Trans. Inf. Syst. 11*, 1, 1–26.

Gellersen, H. W. and Gaedke, M. 1999. Object-oriented Web application development. *Internet Comput. 3*, 1, 60–68.

Gomez, J., Cachero, C., and Pastor, O. 2000. Extending a conceptual modelling approach to Web application design. In *Proceedings of the 12th International Conference on Advanced Information Systems Engineering*. 79–93.

Greenspun, P. 1999. Philip and Alex guide to Web publishing. http://photo.net/wtr/thebook.

GVU. 1997. Graphics, visualization, and usability center's 8th WWW user survey. http:/www.cc.gatech.edu/gvu/user_surveys/.

IEEE. 1998. *Guide to Software Requirements Specifications*. ANSI/IEEE Standard 830-1998.

Isakowitz, T., Stohr, E., and Balasubramanian, P. 1995. RMM: A methodology for structured hypermedia design. *Comm. ACM 8*, 38, 34–44.

Kappel, G., Pröll, B., Retschitzegger, W., and Schwinger, W. 2001. Modelling customizable Web applications - A requirement's perspective. In *Proceedings of the International Workshop on Data Semantic in Web Information Systems*.

Knapp, A., Koch, N., Moser, F., and Zhang, G. 2003. ArgoUWE: A CASE tool for Web applications. In *Proceedings of EMSISE'03*.

Koch, N. 1999. A comparative study of methods for hypermedia development. Tech. rep. 9905, Ludwig-Maximilians-University Munich, Germany.

Koch, N. 2000. Software engineering for adaptive hypermedia applications. PhD dissertation, Ludwig-Maximilians-University, Munich, Germany.

Koch, N., Zhang, G., and Escalona, M. J. 2006. Model transformations from requirements to Web system design. In *Proceedings of the ICWE'06*.

Lang, M., And Fitzgerald, B. 2007. Web-based systems design: A study of contemporary practices and an explanatory framework based on "method-in-action". *Int. J. Req. Eng. 12*, 4, 203–220.

LANGE, D. 1996. An object-oriented design approach for developing hypermedia information systems. *J. Organ. Comput. Electron. Comm. 6*, 3, 269–293.

LEE, H., LEE, C., AND YOO, C. 1998. A scenario-based object-oriented methodology for developing hypermedia information systems. In *Proceedings of the 31st Annual Hawaii International Conference on System Sciences*.

LOWE, D. 2003. Web system requirements: An overview. *Int. J. Req. Eng. 8*, 2, 102–113.

LOWE, D. AND EKLUND, J. 2002. Client needs and the design process in Web projects. In *Proceedings of the Web Engineering Track of the WWW2002 Conference*.

MANDEL, L., KOCH, N., AND MAIER, C. 1998. Extending UML to model hypermedia and distributed systems. Tech. rep. 9804, Ludwig-Maximilians-Universität München, Institut für Informatik.

MDWE. 2007. *International Workshop on Model-Driven Web Engineering*. http://wise.vub.ac.be/mdwe2007/.

MECCA, G., ATZENI, P., AND CRESCENZI, V. 1999. The ARANEUS guide to Web-site development. Tech. rep., University of Rome, Rome, Italy.

MELIÁ, S. AND GOMEZ, J. 2006. The WebSA approach: Applying model driven engineering to Web applications. *J. Web Eng. 5*, 2, 121–149.

MELLOR, S. J., CLARK, A. N., AND FUTAGAMI, T. 2003. Model-driven development - Guest editor's introduction. *IEEE Softw. 20*, 5, 14–18.

MOLINA, F., PARDILLO, J., CACHERO, C., AND TOVAL, A. 2008. Towards a requirements-aware common web engineering metamodel. In *Proceedings of the 2008 Latin American Web Conference*. IEEE Computer Society, 75–82.

NANARD, J. AND NANARD, M. 1995. Hypertext design environments and the hypertext design process. *Commun. ACM 38*, 8, 49–56.

NEWMAN, M. W. AND LANDAY, J. A. 1999. Sitemaps, storyboards, and specifications: A sketch of Web site design practice as manifested through artifacts. In *Proceedings of the ACM Symposium on Designing Interactive Systems*. 263–274

NUA. 1998. Nua Internet surveys. http://www.nua.ie/surveys/.

NUNES, D. A. AND SCHWABE, D. 2006. Rapid prototyping of Web applications combining domain specific languages and model driven design. In *Proceedings of the 6th International Conference on Web Engineering*. 153–160.

OLSINA, L. 1998. Building a Web-based information system applying the hypermedia flexible process modeling strategy. In *Proceedings of the 1st International Workshop on Hypermedia Development, Hypertext*.

OMG 2005. Model Driven Arhictecture. Object Management Group. http://www.omg.org/mda/.

OMG. 2008. Meta Object Facilities (MOF) Query / Views / Transformations 1.0 (QVT). Object Management Group. http://www.omg.org.

OVERMYER, S. P. 2000. What's different about requirements engineering for Web sites? *Int. J. Req. Eng. 5*, 1, 62–65.

PASTOR, O., INSFRAN, E., PELECHANO, V., ROMERO, J., AND MERSEGUER, J. 1997. OO-METHOD: An OO software production environment combining conventional and formal methods. In *Proceedings of the 9th International Conference on Advanced Information Systems Engineering*. 145–158.

PASTOR, O., ABRAHÃO, S., AND FONS, J. 2000. OOWS: An object-oriented approach for Web-solutions modeling. In *Proceedings of the International Conference of Information Society*. 126–129.

PASTOR, O., FONS, J., PELECHANO, V., AND ABRAHÃO, S. 2005. Conceptual modelling of web applications: the OOWS approach. In *Proceedings of the Theory and Practice of Metrics and Measurement for Web Development*, E. Mendes Ed., Springer.

PATERNÒ, F., MANCINI, C., AND MENICONI, S. 1997. ConcurTaskTree: A diagrammatic notation for specifying task models. In *Proceedings of INTERACT'97*. Chapman & Hall, 362–369.

PRECIADO, J. C., TRIGUEROS, M. L., SANCHEZ, F., AND COMAI, S. 2005. Necessity of methodologies to model rich Internet applications. In *Proceedings of the 7th IEEE International Workshop on Web Site Evolution*. 7–13.

SCHWABE, D. AND ROSSI, G. 1994. From domain models to hypermedia applications: An object-oriented approach. In *Proceedings of the International Workshop on Methodologies for Designing and Developing Hypermedia Applications*.

SCHWABE, D., ROSSI, G., AND BARBOSA, S. D. J. 1996. Systematic hypermedia application design with OOHDM. In *Proceedings of the 7th ACM Conference on Hypertext*. 116–128.

SCHWABE, D. AND DE ALMEIDA PONTES, R. 1998. OOHDM-WEB: Rapid prototyping of hypermedia applications in the WWW, Tech. rep. MCC 08/98, Department of Informatitcs, PUC-Rio.

SCHWINGER, W. AND KOCH, N. 2006. Modeling Web applications. In *Web Engineering – Systematic Development of Web Applications*. Wiley, 39–64.

SELIC, B. 2003. The pragmatics of model-driven development. *IEEE Softw. 20*, 5, 19–25.

SHEPHERD, A. 2001. *Hierarchical Task Analysis*. Taylor & Francis, London.

SOUER, J., VAN DE WEERD, I., VERSENDAAL, J., AND BRINKKEMPER, S. 2007. Situational requirements engineering for the development of content management system-based web applications. *Int. J. Web Eng. Tech. 3*, 4.

TAYLOR, M. J., MCWILLIAM, J., FORSYTH, H., AND WADE, S. 2002. Methodologies and website development: A survey of practice. *Inf. Soft. Tech. 44*, 6, 381–391.

THOMSON, J., GREER, J., AND COOKE, J. 1998. Algorithmically detectable design patterns for hypermedia collections. In *Proceedings of Workshop on Hypermedia Development Process, Methods and Models (Hypertext'98)*.

VALDERAS, P. 2008. A requirements engineering approach for the development of Web applications. PhD dissertation. Technical University of Valencia.

VALDERAS, P., PELECHANO, V., AND PASTOR, O. 2007. A transformational approach to produce Web application prototypes from a Web requirements model. *Int. J. Web Eng. Technol. 3*, 1, 4–42.

VALLECILLO, A., KOCH, N., CACHERO, C., COMAI, S., FRATERNALI, P., GARRIGÓS, I., GÓMEZ, J., KAPPEL, G., KNAPP, A., MATERA, M., MELIÁ, S., MORENO, N., PRÖLL, B., REITER, T., RETSCHITZEGGER, W., RIVERA, J. E., SCHAUERHUBER, A., SCHWINGER, W., WIMMER, M., AND ZHANG, G. 2007. MDWEnet: A practical approach to achieving interoperability of model-driven Web engineering methods. In *Proceedings of the 3rd International Workshop on Model-Driven Web Engineering* (In Conjunction with ICWE'07).

VALVERDE, F., VALDERAS, P., AND FONS, J. 2007. A MDA-based environment for web applications development: From conceptual models to code. In *Proceedings of the International Workshop on Web Oriented Software Technology*. 164–178.

VILAIN, P. AND SCHWABE, D. 2002. Improving the Web application design process with UIDs. In *Proceedings of the 2nd International Workshop on Web Oriented Software Technology (IWWOST02)*.

VILAIN, P., SCHWABE, D., AND SIECKENIUS, C. 2000a. A diagrammatic tool for representing user interaction in UML. Lecture Notes in Computer Science.

VILAIN, P., SCHWABE, D., AND SIECKENIUS, C. 2000b. Use cases and scenarios in the conceptual design of Web application. Tech. rep. MCC 12/00. Departamento de Informática. PUC-Rio. Rio de Janeiro, Brasil.

VORA, P. R. 1998. Designing for the Web: A survey. *ACM Interactions 5*, 3, 13–30

WOUKEU, A., CARR, L., WILLS, G., AND HALL, W. 2003. Rethinking Web design models: Requirements for addressing the content. Tech. rep. ECSTR-IAM03-002, University of Southampton.

YOO, J. AND BIEBER, M. 2001. A systematic relationship analysis for modeling information domains. http://citeseer.ist.psu.edu/312025.html.

Received July 2008; revised October 2009, June 2010; accepted September 2010