A Model-Driven Approach to Include Adaptive Navigational Techniques in Web Applications

Gonzalo Rojas, Vicente Pelechano and Joan Fons
Department of Information Systems and Computation
Technical University of Valencia, Spain
{grojas, pele, jjfons@dsic.upv.es}

Abstract

Adaptivity is an increasingly demanded characteristic of Web applications. However, adaptive techniques usually implemented in Adaptive Hypermedia Systems have been hardly considered by current Model-Driven Web Development Methods. This work presents an approach to describe adaptive navigation techniques at early stages of the Web development process. Using the primitives of the OOWS Navigational Model, we have defined a strategy to incorporate three types of techniques (link-hiding, link-ordering and link-annotation) to the Web modelling process. A User Modelling proposal that is necessary for the complete specification of these techniques is also introduced. The impact of the introduced descriptions of techniques on the final adaptive applications is shown by means of a case study.

1. Introduction.

Most of the currently available hypermedia applications provide users with a limited set of possible interactions, with little consideration of the particular characteristics, preferences and needs of each user.

Adaptive Hypermedia arises as an approach that seeks to achieve usage experiences that are more suitable to the individual information needs. Research efforts made by Adaptive Hypermedia community have produced clearly defined concepts and adaptation strategies. However, they are mainly focused on late stages of software development. The developed applications are highly dependant on implementation details and on particular data instances.

As a special and widespread kind of hypermedia systems, Web applications are incorporating adaptivity features. The Model-Driven Web development community has been augmenting the expressiveness of their Web conceptual models for achieving high-level descriptions of Adaptive Web Systems.

However, most of these efforts are limited to the definition of rules that constrain the accessibility of the navigational structures. This strategy gives little support to the implementation of well known adaptive techniques that have been already implemented in existing Adaptive Hypermedia Systems (AHS). The classification of techniques for adaptive navigation and presentation that has been proposed by the Adaptive Hypermedia community is hardly considered and is not expressed at conceptual level, taking little advantage of these previous research efforts.

The main contribution of this work is introducing descriptions of well-known adaptive techniques at conceptual level of the Web development process. In this way, we are able to obtain adaptive Web applications that are more independent of implementation concerns than the most adaptive hypermedia systems.

In the context of OOWS [5], an OO-based modelling method, the specific contributions of this work are:
(a) A User Modelling approach that allows describing the intended users through the definition of groups of similar users (stereotypes) and three graphical models, which contain the needed attributes and operations for achieving the adaptivity.
(b) A set of adaptive navigation techniques that are defined in terms of different conceptual primitives of the OOWS Navigational Model.

The adaptive techniques that are introduced in the OOWS Web development method are classified into three categories that are broadly used by the Adaptive Hypermedia community: (a) the management of nodes accessibility (*link-hiding*); (b) the ordering of the navigational links (*link-ordering*); and (c) the addition of informative hints to displayed links (*link-annotation*).

The rest of this paper describes the three parts of our proposal: Section 2 presents an overview of the OOWS Modelling approach, focused on its Navigational Model. Section 3 describes our proposal for User Modelling, consisting in the definition of user stereotypes and the specification of user attributes through three graphical models. Section 4 describes the adaptive navigation techniques introduced into the OOWS process, presenting examples of their conceptual description and the resulting implementations. Section 5 presents a brief review of related works on conceptual modelling of adaptive Web applications. Finally, Section 6 presents some conclusions and future works.

2. The OOWS navigational modelling approach.

OOWS (Object-Oriented Web Solution) modelling method [5] is the extension of the OO-Method proposal for Automatic Code Generation [7], which introduces the required expressiveness to capture the navigational and presentational requirements of web applications.

The OOWS Conceptual Modelling process consists of three main steps:
(1) *Requirements Elicitation*, where the requirements of the Web application are described by means of UML Use Cases and Scenarios techniques;
(2) *Classic Conceptual Modelling*, where system structure and behaviour are described, using UML-compliant Class, Sequence and State Diagrams; and
(3) *Navigational and Presentational Modelling*, where OOWS Navigational and Presentational Diagrams are built, strongly based on the Class Diagram defined in the previous stage.

The *OOWS Navigational Model* is composed of a set of *Navigational Maps*. Each map corresponds to a global view of the Web application for a given group of users. It is represented by a directed graph, in which the nodes are *Navigational Contexts* and the arcs are *Navigational Links* that define the valid navigation paths.

Navigational Contexts represent interaction points between users and application, and provide a set of cohesive data and operations. There are two types of navigational contexts: (a) *Exploration Contexts*, which are reachable from any node; and (b) *Sequence Contexts*, accessible only via predefined navigational paths.

Figure 1 shows an example of a Navigational Map, corresponding to an online bookstore and defined to a *Client* user type. It is composed of five exploration contexts (with "E" label) and three sequence contexts (with "S" label). Dashed arrows represent the navigational links to the exploration contexts. The solid arrows correspond to the predefined navigational paths that allow accessing the Books Navigational Context, and from this accessing to the other sequence contexts.

Each Navigational Context is composed of *Navigational Classes*, which represent views of the classes included in the Class Diagram. Each navigational context has one mandatory navigational class from which the information retrieval starts, called *Manager Class*, and other optional navigational classes that provide complementary information, called *Complementary Classes*.

Navigational classes contain the visible attributes and executable operations that are available for the user in the corresponding context. All the navigational classes are related through unidirectional binary relationships, so called *Navigational Relationships*. Each of them is defined over an association, aggregation, composition or specialization relationship included in the Class Diagram.

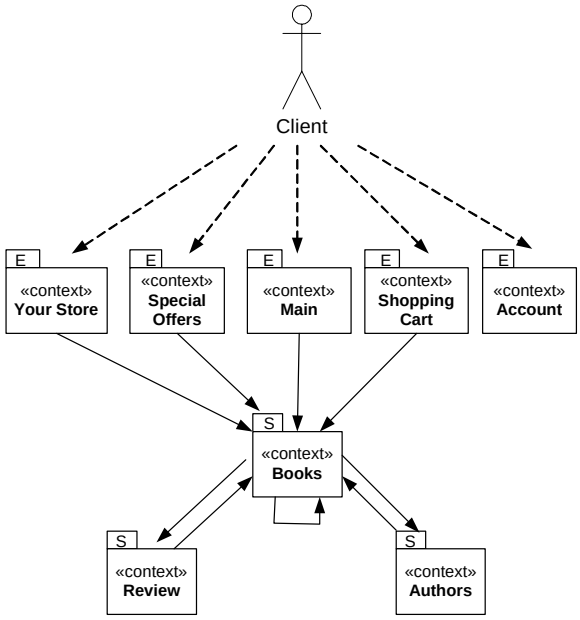

Figure 1. Example of OOWS navigational map and the two types of navigational contexts.

OOWS defines two kinds of Navigational Relationships:
(a) *Context dependency relationship* (dashed arrows), which represents a basic information recovery by crossing a structural relationship between classes.
(b) *Context relationship* (solid arrows), which represents an information recovery plus a navigation to a target context. Context relationships have the following properties:

(1) A *context attribute* that indicates the target context of the navigation (depicted as [*target context*]).

(2) A *link attribute* that specifies the attribute (usually one of the target navigational class) used as the "anchor" to activate the navigation to the target context.

Operation links can also be attached to an operation. An operation link represents the target context (depicted as [*target context*]) that the user will reach after that operation's execution.

Figure 2 shows the *Books* Navigational Context included in Figure 1, which describes a page of a specific book in the online bookstore. *Book* manager class shows basic purchase data of the presented book, while information from complementary classes is retrieved through the shown navigational relationships.

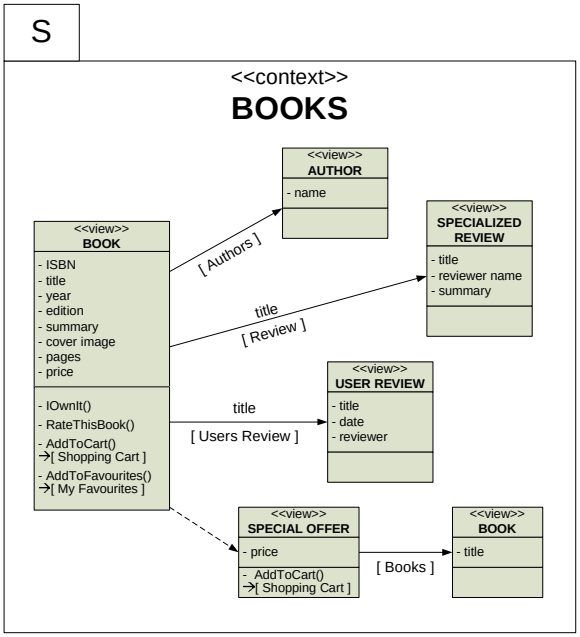

Figure 2. Books navigational context for a web-based bookstore.

Figure 3 shows an implementation of the *Books* Navigational Context. In frame 1, it is possible to distinguish the main data of the presented book, corresponding to attributes and operations of the *Book* manager class; the book review made by a specialist is shown through the *Book-Specialized Review* (see frame 2); book reviews made by other readers are shown through the *Book-User Review* (see frame 3); the data retrieved through the *Book-Special Offer* relationship are shown in frame 4 and correspond to a special offer (so-called "two for one" offer) related to the presented book.

The labels at the top of the page are anchors of links to other Exploration Contexts. Those links and their target contexts are always available. Meanwhile, the contexts that are accessible through the link of the book's author (frame 1), the link of the full-version of the specialized review (2) and through the links to the other readers' reviews (3) are contained in the Sequence Contexts (*Author* and *Review*, see Figure 1) and they are only accessible following a predefined path.

It is possible to define *filters* associated to a navigational class, which allow constraining the objects of this class that are retrieved through a navigational relationship.

As a presentational feature, OOWS includes the *Ordering Pattern*, which allows defining a certain order by which data of the navigational class are presented to the user. This order is based on some attribute of the manager class. The ordering can be ascendant or descendant.

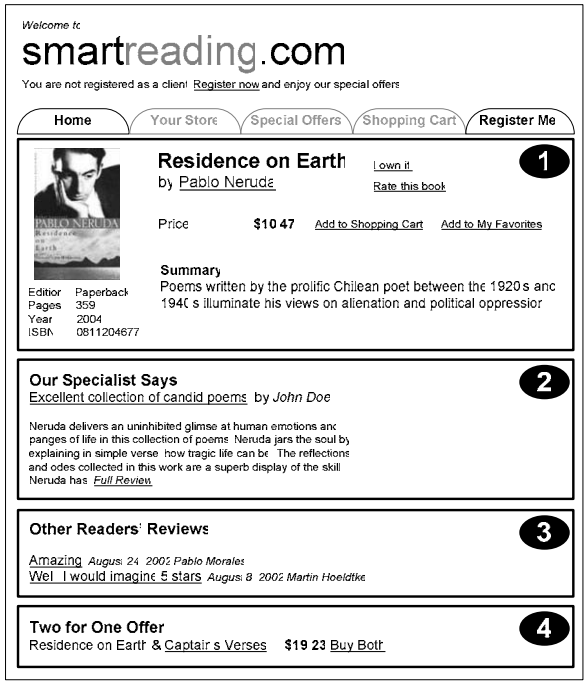

Figure 3. Implementation of Books navigational context.

Let us consider the following example: the section "Your Store" of an online bookstore shows some books that the system recommends to the user. These books must have been published not earlier than 1980 and must be presented in alphabetical order.

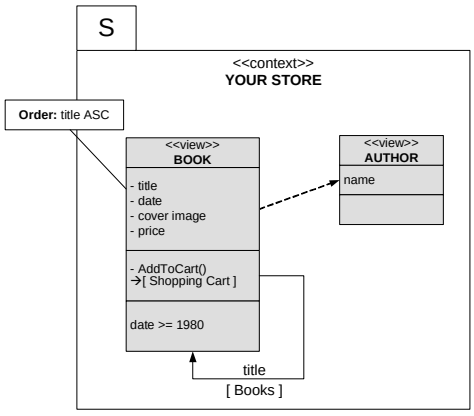

Figure 4. Navigational context for "Your Store" functionality.

The navigation description of this requirement is done through the context shown in Figure 4. It includes a filter (lower section of the *Book* manager class), which allows constraining the books to be displayed to those whose *date* attribute value is greater or equal than 1980.

Furthermore, the Ordering Pattern is applied to the manager class (left-side label), which indicates that the instances of this class must be ordered in ascendant order by their corresponding values of the *title* attribute. Figure 5 shows an implementation of these features.

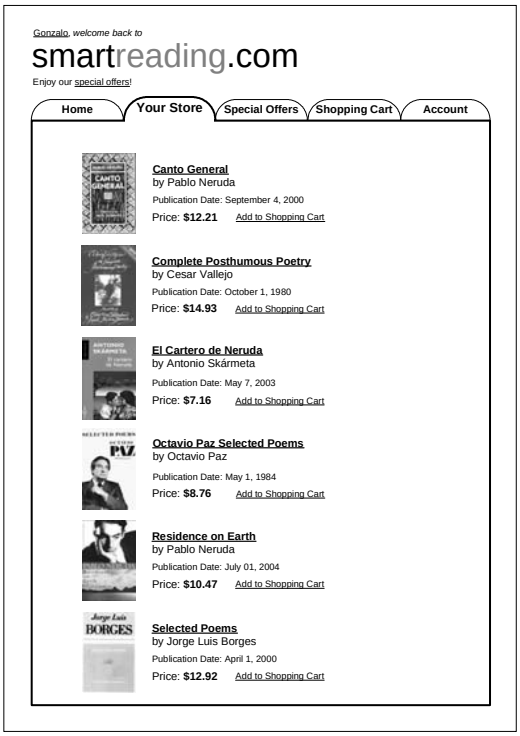

Figure 5. Implementation of "Your Store" functionality.

The conceptual tools of the OOWS Navigational Model provide multiple alternatives for the navigational design of the same web application depending on the user. Distinct navigational maps and variants of the inner structure of the intended contexts (nodes) can be associated to different kinds of users. For this reason, this model is very suitable to incorporate adaptive features to its descriptions. To achieve this goal, we need first a clear description of the intended application's users. In the next section, we present our proposal of User Modelling.

3. Modelling the user.

The goal of our User Modelling proposal is to obtain high level descriptions of the users of the Web application for achieving the system adaptation. We introduce two main steps:

(a) *Definition of User Stereotypes*, where the developer defines a set of user groups (stereotypes) that are ordered in a hierarchical structure;

(b) *Definition of User Model diagrams*, where three user diagrams are built, describing users in terms of: personal information; their relations with a particular application domain; and the navigational actions performed at execution-time.

In these steps, we apply two concepts broadly used for user's descriptions in existing Adaptive Hypermedia Systems. These concepts are:

(a) *Stereotypes*: This concept allow distinguishing several typical or "stereotypes" users, defining user groups whose members are likely to have certain homogeneous application-relevant characteristics [1]. Typical stereotypes are: skilled / novice users, child / young /adult clients.

(b) *Overlay Model*: It represents an individual user's knowledge of the subject as an "*overlay*" of the domain model. For each domain model concept, an individual overlay model stores some value, which is an estimation of the user knowledge level of this concept [8].

The *Stereotype* concept is the basis of the first step, while the *Overlay Model* is applied to the second one. Our proposal uses Overlay Models to describe not only the user's knowledge about a concept, but also other domain-dependent relationships between users and concepts, such as preferences and assigned rankings to a given concept.

3.1. Step 1: Definition of user stereotypes.

In this step, a set of user groups or user stereotypes is defined, according to the adaptation needs of the application. This classification allows taking benefits of some features that some users could share. Thus, the same implementation of an adaptive technique can be used by more than one specific user.

In the process of stereotypes specification, the developer must fulfil three tasks (adapted from [1]):

(a) *Stereotypes identification*: According to the application requirements, the developer identifies the subgroups of users that are likely to share some features and which can interact with the system. These subgroups are called *stereotypes*.

(b) *Stereotypes ordering:* Once defined, stereotypes are hierarchically ordered. These hierarchies allow avoiding redundancy in the definition of similar stereotypes, by inheritance of shared attributes.

(c) *Identification of key user features*: The developer determines some key user features that describe the defined stereotypes, along with the values of these features that allow deciding the inclusion of users into each stereotype.

For instance, let us suppose that the online bookstore must distinguish the followings stereotypes: USER (anyone accessing the bookstore); CLIENT (registered user); CHILD, TEEN, YOUNG ADULT and ADULT

(according to his age); INFREQUENT READER, OCCASIONAL READER and FREQUENT READER (according to his reading habits).

These stereotypes are hierarchically organized. Afterwards, for including users into these stereotypes, the following user characteristics are defined: for USER, no special feature is needed (it is the most general stereotype); CLIENT asks users to execute the *register()* operation. For determining whether a user belongs to CHILD, TEEN, YOUNG ADULT or ADULT stereotypes, a *birthdate* attribute is defined; finally, a *readingHabit* attribute describes how frequently a given user reads. The resulting stereotype hierarchy and the values of the attributes that describe each stereotype are shown in Figure 6.

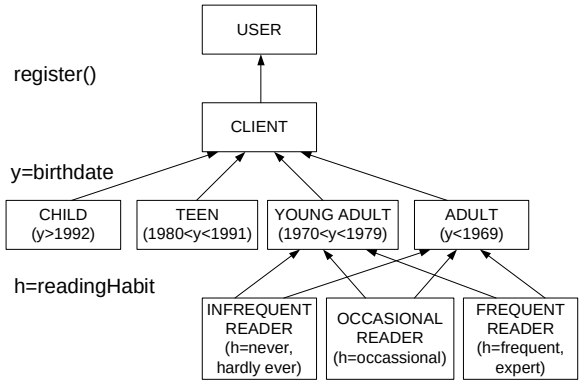

Figure 6. Example of user stereotypes hierarchy.

The definition of Stereotypes is a useful tool for classifying the application users for adaptive purposes. However, their expressiveness is limited, so the adaptive techniques must consider not only the attributes that describe the stereotype, but also the rest of the data expressed through the three diagrams of our proposed User Model.

3.2. Step 2: Definition of User Diagrams.

In this step, users are described through three UML-compliant Class Diagrams, which represent the three most used categories of User Modelling in AHS (examples can be found in [8] and [9]). The diagrams are the following:
(a) *Domain-Independent User Diagram*, which captures the user's personal data;
(b) *Domain-Dependent User Diagram*, which contains the user data related to the application domain; and
(c) *Navigational Behaviour Diagram*, which allows collecting data about the navigational actions that the user performs during a session.

The modeller classifies the attributes and operations that have been previously introduced in the stereotypes definition step, distinguishing those that depend of the application domain from the independent ones. These features are the first structures of the corresponding User Model Diagrams, which are completed with the conceptual structures needed for the fulfilment of the adaptivity requirements.

3.2.1. ***Domain-Independent User Diagram***. This diagram describes those personal user characteristics that have some relevance for the adaptation goals of the application. The values of the modelled data are independent of the user interaction with a particular Web application or specific application domains. For instance, the set of considered attributes may include *name*, *date of birth* or *city of residence*. Figure 7 shows an example of this diagram.

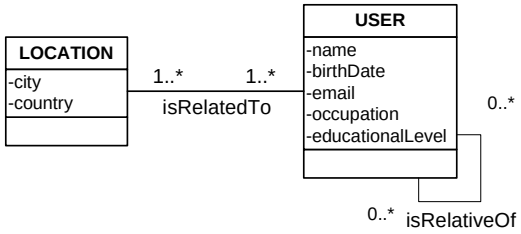

Figure 7. Example of a Domain-Dependent User Diagram.

3.2.2. ***Domain-Dependent User Diagram***. The user attributes and operations that are related to the application domain, along with their relationships with specific domain concepts, are specified in this diagram as an *overlay model* of the previously built structural diagram (Class Diagram).

From the concepts included in this diagram, the system can establish how relevant the domain concepts are for a given user. To achieve this, a *Relevance* property must be defined. This definition depends on the specific application domain and the adaptivity requirements. For instance, in an e-learning scenario, the relevance can be defined as the suitability of contents according to a learning goal, whereas in an e-commerce application the relevance may correspond to the similarity degree of a product to others previously purchased, considering the evaluation of these products that are explicitly assigned by the own user.

To model the relevance, the modeller should take into account specific functionalities of the application, such as the ranking assigned to some article on sale by the user, the purchase of the article or the learning unit the student has visited. The *Relevance* property is used to define the adaptive rules for navigational context instances. A more detailed description of the relevance property is presented in Section 4.

Figure 8 shows the Domain-Dependent User Diagram of the mentioned online bookstore. In the example, the designer pays special attention to the *Client-Book* relationships: books that the client has *visited* (i.e., has accessed their web pages); book titles that the user already *owns*; those ones he mostly prefers (*my_favourites*); and those ones in which he is *not interested*. Numerical values are used for describing the *ranking* assigned to some titles by the user.

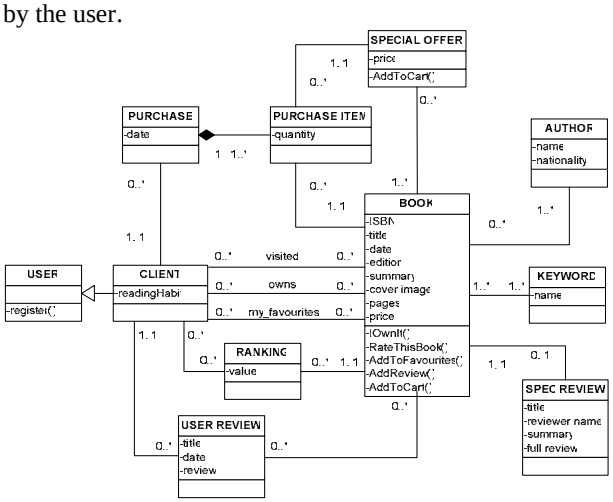

Figure 8. Domain Dependent User Diagram for an online bookstore.

3.2.3. ***Navigational Behaviour Diagram***. It describes the user's interaction with navigational structures of the application during a session. It is defined as an *overlay model* over the navigational structures that are specified for the Web application. This allows describing a user in terms of his visits to a given navigational context, the activations of links (context relationships) and the operations that he executes.

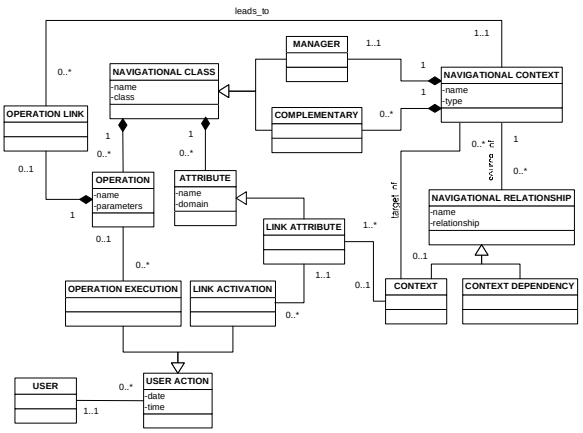

Figure 9. Example of OOWS Navigational Behaviour Diagram.

Figure 9 shows an example of this diagram. It is composed of a view of the OOWS navigational meta-model (which allow modelling the navigational structures), the *User* class and the *User Action* class, along its corresponding subclasses, which describe the activation of a link or the execution of an operation.

These data are used to update the User Model, specifically the Domain-Dependent User Diagram. Specific visited contexts and activated links can reveal important features of the user, such as the knowledge level about the accessed contents or the preferences for them. This update process can be achieved through update rules, whose definition is one of the future works of our proposal.

4. Adaptive techniques in the OOWS navigational model.

OOWS Navigational Model provides the developer with the required expressiveness to define adaptive navigation techniques at conceptual level. There are multiple techniques of adaptive navigation which are implemented in existing Adaptive Hypermedia Systems. Brusilovsky [9] classifies them into the following six groups:

a) *Adaptive Link Hiding*: these techniques restrict the navigation space by hiding the links to pages that are irrelevant or forbidden to a certain kind of users. The "hiding" can be implemented, for instance, by deactivating or not showing the corresponding link(s).

b) *Adaptive Link Ordering*: these techniques sort the links of a given page, according to the user characteristics and adopting some criteria to emphasize some links from the others, e.g., the closer to the top of the page the link is displayed, the more important the target information is.

c) *Adaptive Link Annotation*: this kind of techniques allows complementing links with comments about the pages that are accessible through those links. These annotations can be textual (comments) or visual (icons, different colours or text fonts) and give information about the relevance of the target pages or their current state (visited, unavailable, etc.).

d) *Direct Guidance*: the goal of these techniques is to suggest the next best node, that is to say, the most relevant page to be accessed immediately after the current one.

e) *Adaptive Link Generation*: these techniques consist on the generation of new links, which have not been considered in the application authoring.

f) *Map Adaptation*: It corresponds to the distinct ways of adapting the navigational maps, influencing the structure or topology of the map.

In this section, we describe the way that OOWS allows describing adaptive navigation techniques at conceptual level. As a first step of this integration, we have focused

our work on the first three groups of techniques, privileging those ones that do not alter the topology of the navigational maps. For this reason, the consideration of *map adaptation* and *adaptive link generation* techniques are delayed to further works. In the same way, the consideration of *direct guidance* techniques is also postponed, because it requires a deeper analysis of navigational structures at instance level.

The adaptive techniques that this work proposes make use of the following three properties:

(a) *Accessibility*. It is a property applicable to Navigational Links and Navigational Relationships of a context. It allows constraining the access to the contents that are retrieved through these structures, depending on the user characteristics. This property is the basis of the implementation of Link-Hiding techniques for the mentioned structures.

(b) *Importance Order*. This property supports the Link-Ordering and Link-Annotation techniques for Navigational Relationships. For each context, a value is assigned by the modeller to each relationship, depending on the importance that he estimates each relationship has for a given stereotype.

(c) *Relevance.* It refers to the importance of the domain concept instances to the current application user. It is used by the OOWS adaptive techniques that are defined for navigational class instances. According to the adaptivity requirements, the developer describes the relevance in terms of the primitives included in the Domain-Dependent User Diagram. For instance, in the e-bookstore application, the relevance of a *B* book for a *C* client can be described as follows:

- The number of instances of *Keyword* class that are related to *B* and also with books that have been already purchased, owned, marked as favourite book, highly rated or visited by *C*. This definition is based on the following concepts (included in Domain-Dependent User Diagram, see Figure 8): *Book*, *Keyword*, *Client*, *Purchase* and *Ranking* classes; *Book-Keyword*, *owns*, *my_favourites* and *visited* relationships.
- The number of visited relationships between *C* and Book instances that are related to the same *Author* instance that is related to *B.* The considered concepts are: *Book*, *Author* and *Client* classes; *Book-Author* and *visited* relationships.

The adaptive techniques that this work introduces will be described according to their corresponding group of techniques (*link-hiding*, *link-ordering* and *link-annotation*), and to the conceptual structure to be adapted. Table 1 shows the type of techniques that this work comprises, along with the OOWS navigational structures to which those techniques are applied.

Table 1: OOWS Adaptive navigation techniques

	Navigational Class instances	Navigational Relationships	Navigational Links
Link-Hiding	X	X	X
Link-Ordering	X	X	
Link-Annotation	X	X	

4.1. Adaptive Link-Hiding.

OOWS provides the conceptual tools to define Link-Hiding techniques for different navigational structures.

4.1.1. Link-Hiding for Navigational Links. This technique is based on the "accessibility" property of this structure. Each navigational link of the Navigational Map that is defined for a given stereotype is marked with the "accessible" or "not accessible" value, according to the adaptivity requirements. If a link is "not accessible" to this stereotype, these users can not access to the target Exploration Context that is target of that link.

For instance, in the online bookstore, the developer has assigned accessibility values to the Navigational Links for users of *Client* and *User (non-client)* stereotypes. Figures 10 and 11 show the resulting Navigational Maps for each of these stereotypes, respectively.

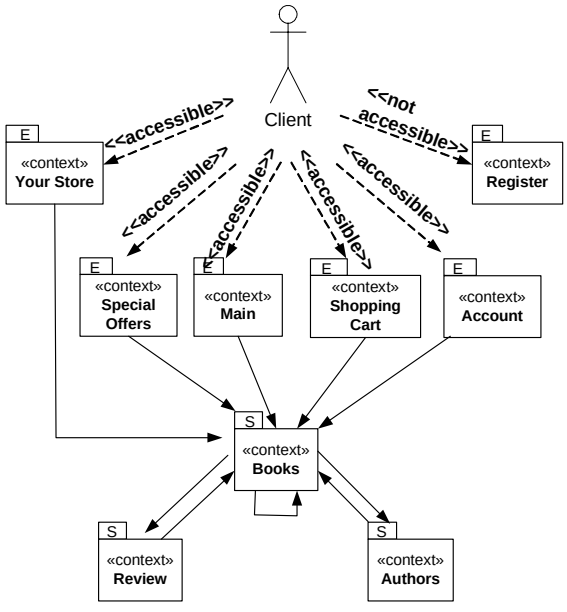

Figure 10. Accessibility values for navigational links for Client stereotype.

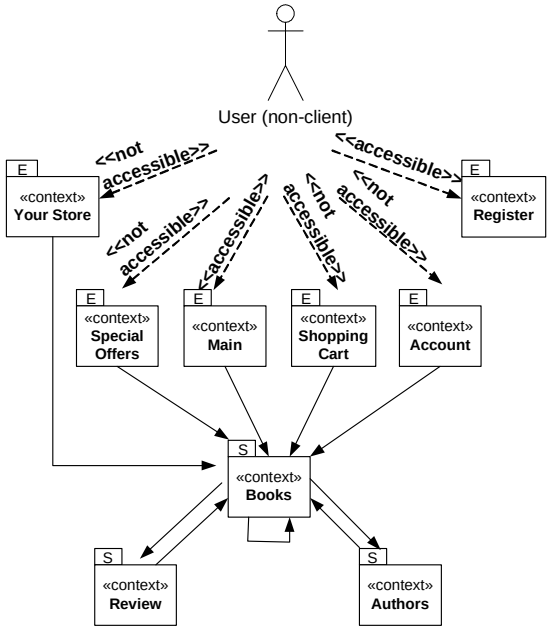

Figure 11. Accessibility values for navigational links for Client stereotype.

Considering the assigned accessibility values, a *Link-Hiding* technique is implemented, deactivating the <<not accessible>> links. Figure 12 shows the implementation for *Client* stereotype, in which the link to the Registration page is the only one that is inaccessible, whereas in Figure 13, corresponding to the implementation for *User non-client* stereotype, most of the displayed links are deactivated.

Figure 12. Link-Hiding for Client stereotype.

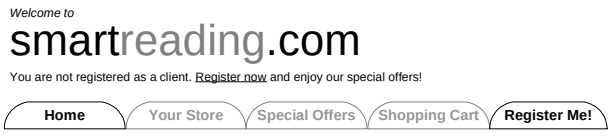

Figure 13. Link-Hiding for User (non-client) stereotype.

4.1.2 Link-Hiding for Navigational Relationships. Our proposal allows managing the access to Sequence Navigational Contexts by means of defining the Accessibility property for Navigational Relationships. For each context of a Navigational Map, the developer should assign the "accessible" or "not accessible" value to its navigational relationships, according to the different user. The accessibility property is defined for the navigational relationships whose source is the manager class of their context. The rest of the navigational relationships inherit the accessibility values from the relationships that allow accessing to them.

Figure 14 shows the accessibility values that the modeller has assigned to the navigational relationships of the *Book* context, considering the users of the *Infrequent Reader* stereotype. The user feature of not making any previous purchase on the site has been also considered, that is to say, in the Domain-Dependent User diagram there is no object of *Purchase* class that is associated to the correspondent instance of *Client* class.

In the example of the Figure 14, the modeller has considered that this kind of users has little interest on the *Specialized Review* of the book, marking the corresponding relationship with the "not accessible" value. Let us consider the following requirement of adaptivity: *"the access to a special offer associated with a book is restricted to those clients with a certain amount of previous purchases"*. According to the characteristics of the stereotype, the additional user feature compels to assign the same value for the *Book-Special Offer* relationship.

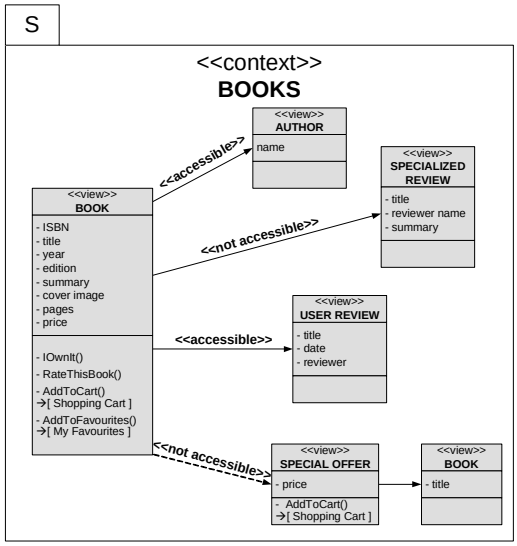

Figure 14. Accessibility Values assignment to the navigational relationships of Book context.

Figure 15 shows the implementation of the resulting context. We can see two different ways of hiding the links corresponding to the <<not accessible>> relationships. In both cases, the whole retrievable information has not been displayed. However, in the case of the *Book-Special Offer* relationship (see Frame 4), some information indicates the existence of information that is not accessible.

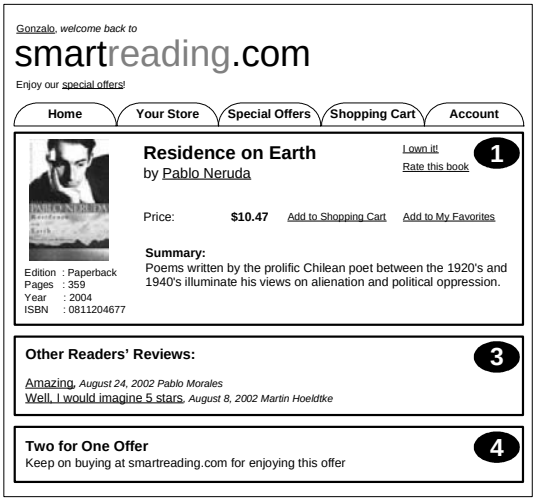

Figure 15. Implementation of Link-Hiding for Navigational Relationships.

4.1.3 Link-Hiding for Navigational Class Instances. This technique allows hiding the links to some instances of a navigational class, according to their corresponding values of relevance for a given user. This kind of link-hiding technique is applied to one or more navigational contexts whose manager class is the navigational class to be constrained. This manager class is extended with a filter, which constrains the objects to be shown. This filter allows accessing to the objects that have a relevance value greater than a given fix value, determined by the modeller.

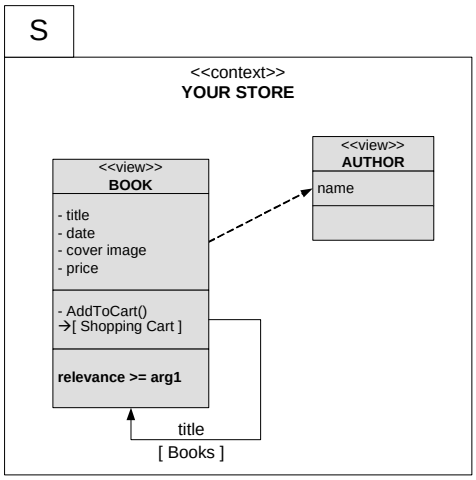

Figure 16. Conceptual specification of Link-Hiding technique for instances of the Book navigational class.

Figure 16 shows the applying of this technique to the context shown in Figures 4 and 5. The modeller constrains the books to be proposed, showing only those whose relevance for the current user is equal or greater than an *arg1* value. The final implementation of this technique is shown in Figure 17.

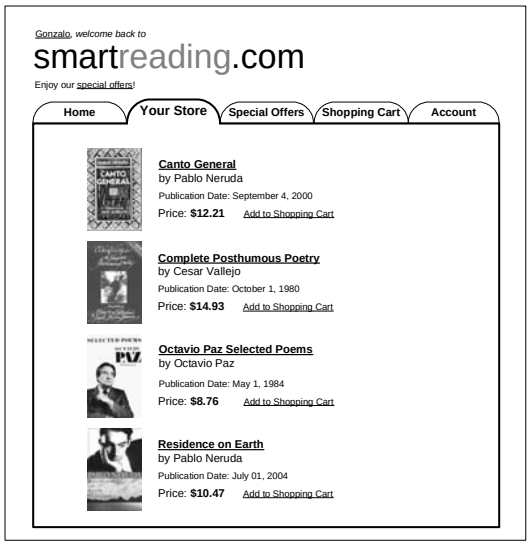

Figure 17. Implementation of Link-Hiding technique for instances of Book navigational class.

The context of Figure 16 only differs from the shown in Figure 4 on the inclusion of the filter by relevance. From a little change at conceptual level, it is possible to obtain an implementation highly different from the presented in Figure 5.

As Figure 17 shows, the links to the Book instances maintain their original order, but those ones whose relevance for this user is less than *arg1* have been hidden.

4.2. Adaptive Link-Ordering and Link-Annotation.

4.2.1 Link-Ordering and Link-Annotation for Navigational Relationships. Both kinds of adaptive techniques are supported by the Importance Order Property of Navigational Relationships. The modeller assigns importance values to the navigational relationships whose source is the manager class, establishing an order of importance among these relationships for a given user stereotype. In this way, displaying of information retrieved from one or another relationship is prioritized.

Figure 18 shows an example of importance values (depicted with the <<IO:value>> tag) that the modeller has assigned to the contextual relationships of Books context, for the *Occassional Reader* stereotype (*Client* instances with "*occasional*" value for the *readingHabit* attribute).

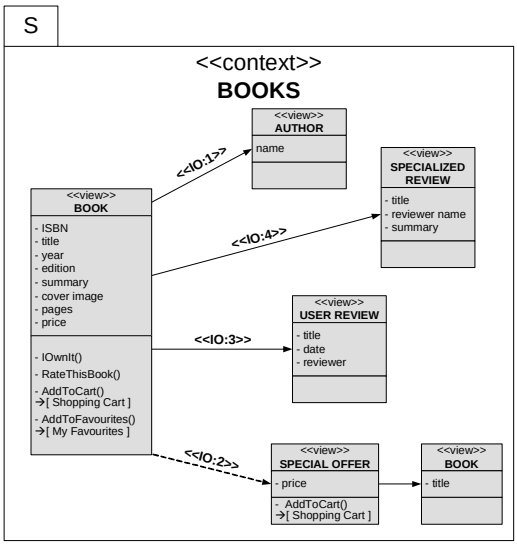

Figure 18. Importance Order values for Navigational Relationships.

The modeller assigns the initial order values, which may be modified according to adaptivity requirements. For instance, data about some user actions (modelled in the Navigational Behaviour diagram) may increase the score of the corresponding relationship for this user.

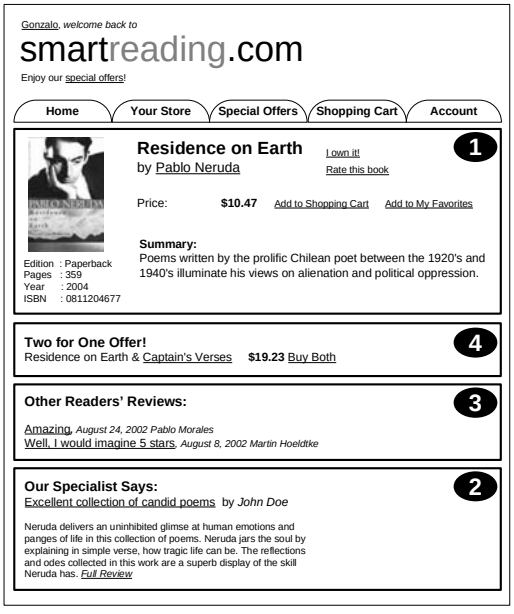

Figure 19. Implementation of Link-Ordering technique for Book Navigational Context.

The determined order of importance is used for the implementation of a Link-Ordering and a Link-Annotation technique. The difference between them is noticeable in the Presentation Layer of the application. In the first case (Figure 19), the information items that are retrieved through the navigational relationships are ordered in the layout of the page: items of the most important relationship are closer to the top of the page; in the case of the Link-Annotation (Figure 20), the information items maintain their original location into the page, but a visual hint is assigned to each of them, giving information of the individual importance for the current user (in Figure 17, a bigger quantity of "book" icons next to the data items means that the implied relationship is more important).

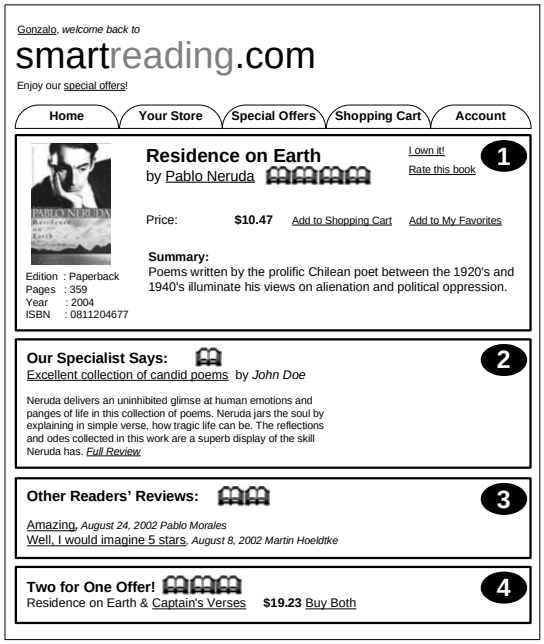

Figure 20. Implementation of Link-Annotation technique for Book Navigational Context.

4.2.2. Link-Ordering and Link-Annotation for Navigational Class instances. This technique allows ordering the links to the instances of a navigational class, according to their corresponding values of relevance for a given user.

Considering the navigational contexts whose manager class is the navigational class to be constrained, the link-ordering technique is described in the OOWS Navigational Model by means of the Ordering pattern. The developer incorporates this pattern to the manager class, ordering its instances by the relevance values, in descendant or ascendant order.

In Figure 21, the requirement of presenting the proposed books in a descendant order of relevance has been modelled. Figure 22 shows the resulting implementation. The whole information associated to the relevant link is ordered. On the top, the most relevant book to this user.

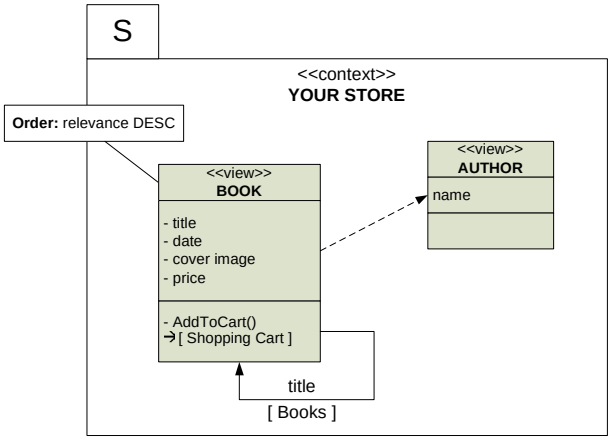

Figure 21. Conceptual specification of Link-Ordering technique for instances of Book navigational class.

There is not a navigational description for the Link-Annotation technique for Navigational class instances. In this case, the relevance order of the instances is considered directly by the presentation layer of the application, where visual clues are added to each link (more coloured "star" icons implies a more important book, see Figure 23).

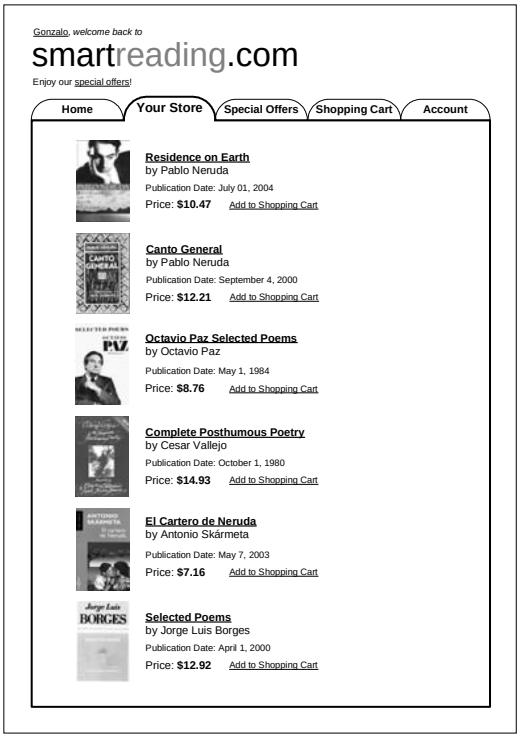

Figure 22. Implementation of the "Your Store" functionality, applying Link-Ordering technique.

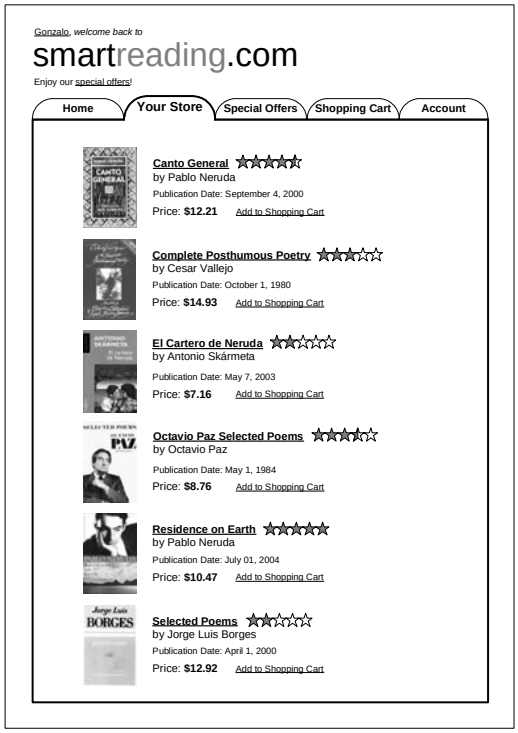

Figure 23. Implementation of the "Your Store" functionality, applying Link-Annotation.

5. Related work.

Some existing Model-driven Web development methods have been increasing the expressiveness of their models to support adaptivity features. Most of them are based on the definition of adaptive rules. For instance, in the *WebML* adaptivity approach [10], a set of Event-Condition-Action rules allows the system to make adaptations of its navigational structure in reaction to context changes produced by link activations. The adaptivity proposal of *OO-HDM* [3] describes users inside the structural model, assigning roles and defining algorithms that implement different adaptive rules for distinct user profiles. The *OO-H* proposal [4] separates the static and the variable parts of the Web application. By means of XML adaptive rules, the adaptation occurs over the variable part. Even if the OOWS proposal considers implicitly the definition of adaptive rules, it describes the adaptive features at a higher abstraction level in terms of adaptive techniques, instead of making rule-based descriptions. In this way, we obtain adaptive descriptions that are closer to the adaptivity requirements, intuitive for the modeller and with a more direct mapping to the final implementation.

Both *HERA* and *UWE* adaptivity approaches make a clear separation of the modelling and adaptation of content, navigation and presentation. This characteristic is also shared with our approach. The *HERA* approach [2] specifies the adaptivity by defining schema views on data for each model, performing adaptation without referring to concrete data instances. The main difference with *HERA* approach is that OOWS is object-oriented, providing a higher expressiveness in terms of the functional dimension of the Web application. The *UWE* approach defines a reference model [6] that formally specifies the features of an adaptive hypermedia application, incorporating a user meta-model and an adaptation meta-model. Our work can complement this proposal, providing the expressiveness to precisely define the concrete adaptive techniques to be implemented in terms of *UWE* adaptive rules.

6. Conclusions.

This work has presented our approach for incorporating adaptive navigation techniques into the conceptual modelling process for Web applications. We have augmented the expressiveness of a navigational model, introducing the required concepts to describe some well-known adaptive navigation techniques at a high abstraction level.

The flexibility of the OOWS navigational model makes possible that a non-adaptive specification can evolve into a diagram of an adaptive Web application, by using simple primitives. Both the User modelling proposal and the modelling of adaptive techniques have made use of concepts previously developed by Adaptive Hypermedia and User Modelling research communities. This help to increase the interoperability of the obtained Web applications with existing Adaptive Hypermedia Systems.

Some future works that complement this proposal are: (1) describing adaptive techniques of direct guidance, map adaptation and link-generation to the navigational modelling process; (2) describing adaptive presentation techniques, through the OOWS presentational model, adopting a similar approach; and (3) defining rules for updating the Domain-Dependent User diagrams from the navigational user actions.

7. Acknowledgments.

This work has been developed with the support of MEC under the project DESTINO TIN2004-03534 and cofinanced by FEDER.

10. References.

[1] A. Kobsa, "User Modeling: Recent Work, Prospects and Hazard", *Adaptive User Interfaces: Principles and Practice*, North Holland Elsevier, Amsterdam, 1993, pp. 111-128.

[2] F. Frasincar, G. J. Houben, and R. Vdovjak. "Specification framework for engineering adaptive web applications", *Proceedings of The Eleventh International World Wide Web Conference*, Honolulu, Hawaii, USA, May 7-11, 2002.

[3] G. Rossi, D. Schwabe, and R. Guimarães, "Designing Personalized Web Applications", *Proceedings of the World Wide Web Conference (WWW'10)*, Hong Kong, China, 2000, pp. 275-284.

[4] I. Garrigós, J. Gómez, and C. Cachero, "Modelling Dynamic Personalization in Web Applications", *Proceedings of Third International Conference on Web Engineering (ICWE'03)*,LNCS, Vol.2722, Springer-Verlag, Oviedo, Spain, 2003, pp. 472-475.

[5] J. Fons, V. Pelechano V., M. Albert, and O. Pastor, "Development of Web Applications from Web Enhanced Conceptual Schemas", *Proceedings of the International Conference on Conceptual Modelling, 22nd Edition, ER'03*, LNCS, Vol. 2813. Springer-Verlag, Chicago, USA, 2003, pp. 232-245.

[6] N. Koch, and M. Wirsing, "The Munich reference model for Adaptive Hypermedia Applications", *Proceedings of the Second International Conference on Adaptive Hypermedia and Adaptive Web Systems*, LNCS vol.2347, Málaga, Spain, 2002, pp. 213-222.

[7] O. Pastor, J. Gómez, E. Insfrán, and V. Pelechano, "The OO-Method Approach for Information Systems Modelling: From Object-Oriented Conceptual Modelling to Automated Programming", *Information Systems, Vol. 26, N. 7*, Springer-Verlag, Chicago, vol.2347, 2001, pp. 507-534.

[8] P. Brusilovsky, "Methods and Techniques of Adaptive Hypermedia", *User Modeling and User-Adapted Interaction, 6 (2-3)*, 1996, pp.87-129.

[9] P. Brusilovsky, "Adaptive Hypermedia", *User Modeling and User Adapted Interaction, 11*, 2001, pp. 87-100.

[10] S. Ceri, P. Fraternali, and S. Paraboschi, "Data-driven, one-to-one web site generation for data-intensive applications", *Proceedings of 25th International Conference on Very Large Data Bases (VLDB'99)*, Morgan Kaufmann, Edinburgh, Scotland, September 7-10, 1999, pp. 615-626.