Improving the Design of Existing Web Applications

Mario Luca Bernardi[1], Giuseppe Antonio Di Lucca[2] and Damiano Distante[3]
[1,2]Department of Engineering, University of Sannio, Italy
[3]Faculy of Economics, Unitelma Sapienza University, Italy
{[1]mlbernar,[2]dilucca}@unisannio.it, [3]damiano.distante@unitelma.it

***Abstract*—Despite several methodologies have been defined to support the disciplined development of Web applications, often such methodologies are not applied in the practice, mainly due to the short time-to-market and resource constraints. As a consequence, existing (i.e. legacy) Web applications often lack in design quality. This paper proposes a model-driven semi-automatic redesign approach to improve the design of existing Web applications. The approach analyzes the client side HTML pages of the application to recover its conceptual model according to the Ubiquitous Web Applications design methodology. The recovered model is then used as a starting point to define a new design for the application, adopting the Model-View-Controller architectural pattern and the JavaServer Faces technology. A concrete example of the application of the approach to redesign a real world Web site is also described.**

Keywords: Web Applications Quality Improvement; Web Application Redesign; Reverse Engineering; UWA

I. Introduction

Web applications (WAs) are characterized by continuous maintenance and evolution operations to meet new functional and non-functional requirements of the evolving context in which they are used. Several methodologies have been proposed in the literature to support their design and development. The list of such methodologies includes the Object-Oriented Hypermedia Design Method (OOHDM) [13], the UML-based Web Engineering approach (UWE) [5], the Web Modeling Language (WebML) [2] and the Ubiquitous Web Applications (UWA) design framework [16]. Very often such methodologies are not applied in the practice, mainly due to the short time-to-market and the limited resources often constraining the development (and maintenance) of WAs. As a consequence, existing (i.e. legacy) WAs often lack in design and documentation quality which cause maintenance and evolution to become difficult and risky tasks. In these cases, the recovery of models describing the current and actually implemented design and a re-engineering (mainly a redesign) of the WA is strongly recommended to improve its quality and to reduce the effort required for the WA maintenance and evolution.

Design patterns [4] have proven to be a mean to improve the quality attributes affecting the maintainability of a software system [7]. In recent years, the usage of design patterns has become popular also in the development of WAs, giving origin to a new class of WAs known as pattern-based WAs [11]. Pattern-based development processes allow to improve the quality attributes affecting the architectural reusability and consistency of the produced application, as well as its evolvability and maintenability.

In this paper we propose a model-driven semi-automatic redesign approach for WAs which enables to improve their design and to migrate them towards a Model-View-Controller (MVC) architectural design pattern and the JavaServer Faces (JSF) technology.

The proposal relies on the UWA design methodology and integrates the REUWA reverse engineering approach proposed in [1] with the forward model-driven design approach described in [3].

The approach allows to improve the quality of an existing WA as it: (*i*) produces up-to-date documentation of the current design; (*ii*) uses the recovered model to redesign the application using a proper Web engineering method, such as UWA, in a model-driven style and adopting the MVC architectural pattern; (*iii*) enables introducing changes in the design of the application in order to meet new requirements or to better satisfy existing ones.

The rest of the paper is organized as follows. Section II discusses some of the works related to ours. Section III describes the overall model-driven redesign approach. A practical application of the approach to redesign a real-world web site is reported in Section IV. Finally, Section V provides some conclusive remarks and introduces future work we aim to carry out.

II. Related Work

Several approaches have been proposed for the Reverse Engineering (RE) of a WA. The approaches in [9], [12] recover an architectural view of the WA depicting its components (in terms of pages, page components, etc.) and the relationships among them at different levels of detail. In [8] an approach for abstracting a description of the functional requirements implemented by the WA is proposed. An approach based on UWA for the user-centered RE of the business processes implemented by a WA is proposed in [14]. The VAQUISTA [17] system allows the presentation model of a web page to be reverse engineered, in order to migrate it to another environment.

Most of the existing RE approaches for WAs are usually focused to recover models related just to one aspect of the WA. The RE phase of our redesign approach, instead, relies

on the REUWA reveapproach [1] which is able to recover, in a semi-automatic way, a set of UWA models representing the user-centered conceptual design of the WA and covering its content, navigation and presentation aspects.

The Model-Driven Engineering (MDE) paradigm is applied successfully by a number of web engineering methods. The most well known approach to model driven engineering is the Model Driven Architecture (MDA) defined by the Object Management Group (OMG). As an example, the UWE [5] method and its variant UWE4JSF which has the JavaServer Faces as target development platform [6] follow the MDA principles and use OMG standards such as the QVT transformation language [10].

These methods share with the forward design phase of our redesign approach the use of meta-models and models transformations towards automatic code generation. WebML [2] follows an MDE approach for mapping its modelling elements onto the components of the MVC Model 2 architecture, which can be transformed into components for different platforms. WebML differs from our and other considered approach because its process is an MDE one but not MDA. Similar to our approach, WebML uses the MVC pattern for its Platform Independent Models (PIMs). Our choice of adopting MVC as architecture for the PIM logical model guarantees the availability of a wide range of open-source and commercial technology frameworks to choose from, for the implementation stage, such as J2EE, .Net and PHP.

Finally, to the best of our knowledge, there is no semi-automatic and model-driven redesign approach for WAs that integrates a reverse engineering phase with a forward phase, both based on a design methodology specific for WAs (such as UWA), and with the second phase taking as input the models recovered by the first phase.

III. The Redesign Process

The proposed redesign process consists of a reverse engineering phase and a forward model-driven design phase. The first phase relies on REUWA, [1] a reverse engineering approach which allows the abstraction of UWA conceptual models from existing WAs. The forward phase takes as input the recovered UWA models and enables to refine, evolve, and transform them into a low level design model that supports re-implementing the application adopting the MVC architectural pattern and the JavaServer Faces (JSF) technology framework.

Tool support is provided for both phases which are semi-automatic and require the user intervention mostly to drive the different process activities and validate the produced results. In particular, the reverse engineering phase is supported by the REUWA tool [1] and the forward design phase by the UWAMDD tool [3].

A. The UWA Web Design Methodology

The UWA design framework allows to specify the design of a WA by means of three main models: the Information Model, the Navigation Model, and the Presentation Model [15]. Additional UWA models include: the Transaction Model, to model the business processes the application is intended to support; the Operation Model, used to specify the elementary operations the application will provide to its users; the Customization Model, specifying, by customization rules, how the application can be adapted to different usage contexts.

The Information Model comprises two sub-models: the Hyperbase model and the Access Structures model. The first describes the contents of the applications in terms of base information classes (Entities), their structure (Components and Slots) and their relationships (Semantic Associations). The Access Structures model defines subsets (Collections) of the application contents, each based on a selection criterion derived from a specific information access user goal. The Navigation model assembles elementary information elements (slots from one or more entities, association centers and collection centers) into reusable units of consumption (Navigation Nodes) and defines navigation contexts (Navigation Clusters) by grouping nodes and defining navigation paths through them (using Navigation Links). The Presentation model specifies how the application is organized in pages, which are the sub-components of each page (Publishing Sections and Publishing Units), and which node is published in each publishing unit.

Each UWA model describes a specific aspect of the designing application from the perspective of the final user, thus specifying how the application will be perceived by its users, rather than how it will be implemented.

An example of UWA models, including a portion of the Information, Navigation and Publishing models is reported in Figure 3.

B. The Reverse Engineering Phase

REUWA is a semi-automatic reverse engineering approach defined to abstract UWA conceptual models from existing WAs. Here, we shortly synthesize this approach, while more details can be found in [1].

REUWA enables the recovery of the following UWA models:

- Information model
- Navigation model
- Publishing model

For a given WA the process is carried out by analysing a significant amount of client side HTML pages downloaded from the application by means of a Web crawler.

1) UWA Information Model Abstraction: It is recovered by analyzing the HTML pages of the WA to abstract Entities, Semantic Association and Collections.

UWA Entities are identified by searching for groups of related attributes (i.e. keywords) in the client side HTML pages (static and dynamically generated) of the WA. A group of keywords involved in the same user input or output operation and included in the same HTML form or

output report is considered as a possible group of Slots characterizing a UWA Entity. The rationale behind this assertion is that the set of data items that a user enters into an input form, or that are shown to a user by an output report, usually represents a concept of interest for the user in the domain of the application. Similar considerations apply to groups of keywords characterizing a set of cloned client pages, i.e. a group of client pages characterized by the same HTML control structure but different content. In this case keywords can be identified by considering labels associated to content items (such as text, images, multimedia objects, etc.), text appearing in table headings, titles appearing in page sections, etc.. From each group of cloned client pages a HTML page Template is produced. This template has the same control component and the same set of keywords that are common to all the pages in the set of cloned pages. Each identified keyword is candidate to be a UWA Slot and the keywords in a group are candidate to be an Entity Component. Edit distance metrics and clustering techniques have been defined to identify groups of cloned pages and extract from them associated groups of keywords.

A candidate Semantic Association is assumed to exist between pairs of Entities having some Slots in common. If different Entities are shown in the same HTML page, a candidate Association between them is also considered to exist. Semantic Associations are also derived from hyperlinks connecting pages showing different Entities, mainly when a Slot is used as an anchor to set the hyperlink.

The identification of UWA Collections is mainly based on (*i*) the usage of a table where each row reports a different instance of a given Entity or Association; (*ii*) a list of hyperlinks pointing to pages showing different instances of the same Entity.

At the end of each of the above steps, a validation activity is manually carried out by a human expert knowledgeable of the application domain to define a validated set of UWA Entities, Semantic Association, and Collections associated to the considered application.

2) UWA Navigation Model Abstraction: It is carried out by identifying Nodes and Clusters for the analyzed application. Nodes are identified by associating them to structural sections in the pages of the application, displaying /requiring information from/to the user. The client pages related to Entities, Associations, and Collections are selected and analyzed to: (*i*) identify which attributes of each Entities, Associations, or Collections are referred in the page; (*ii*) associate a Node to each group of attributes; (*iii*) identify hyperlinks connecting Nodes in the same page or in different pages. Links between nodes are used to identify Navigation Clusters. A list of Nodes and their organization into Clusters is the result of this step. Each Node and each Cluster is assigned a unique name derived from the elements of the Information Model they are associated to.

3) UWA Publishing Model Abstraction: It is abstracted by identifying Publishing Pages (PPs), Publishing Sections

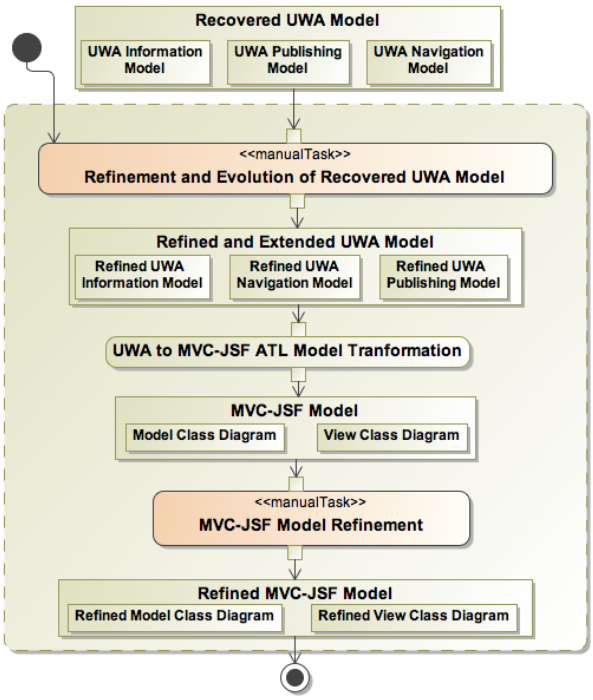

Figure 1: The forward design phase of the redesign process

(PSs) and Publishing Units (PUs) from the set of templates obtained during the phase of Information Model recovery. A PP is associated to each template contributing to the identification of at least an Entity. To identify PSs and PUs associated to each PP, a PS is assumed to include only one PU and a PU is associated to each of the Nodes recovered in the Navigation Model. By tracing the association between Nodes and templates it is possible to associate PUs to PPs.

C. The Forward Design Phase

The model-driven forward design phase of our redesign approach is depicted in Figure 1. In this phase, the UWA models recovered with REUWA are first refined and evolved to meet possible new requirements for the application, and afterwards they are transformed, through an automatic step, to produce a low level design model, named MVC-JSF model, which enables re-implementing the application adopting the MVC architectural pattern and the JSF technology.

An example of the generated MVC-JSF design model is reported in Figure 3 (right side). The model consists of two diagrams: a Model Class Diagram (MCD) and a View Class Diagram (VCD). The first is defined to represent the Model component of the MVC architecture, and the latter to represent the components View and Controller. The MCD defines the classes with methods and attributes (implemented by means of JavaBeans in JSF) corresponding to the content types (UWA Entities) of the application; the VCD represents the presentation layer of the application and describes the structure of pages in terms of visual

Table I: Mapping between UWA and UML-MVC modeling primitives

UWA Conceptual Model	UWA modeling Concept	MVC Component	UML-MVC Modeling Element
Information Model	Entity Type	Model	Class into MCD
	Slot	Model	Private attribute and associated set and get methods into MCD
	Semantic Association	Model	Association between Classes into MCD
	Association Center	Model	Private Attribute and associated methods into MCD
	Collection Type	Model	Class into MCD associated to the Collection Center Private Attribute and associated methods added to the Class involved in the collection
	Navigation Node	View	Class into VCD
Navigation Model	Navigation Cluster	View	Navigation Links between Classes into VCD
	Publishing Unit	View	Class into VCD
Publishing Model	Section	View	Class into VCD aggregating classes originated from Publishing Units
	Page	View	Class into VCD aggregating classes originated from Publishing Sections
	Link	View	Navigation Links between Views of the VCD

```
rule Collection2ModelClass {
from
  c : UWA!ConnectibleElement
    (c.oclIsTypeOf (UWA!Entity)
    and c.container -> notEmpty())
    to m2 : MVC!ModelClass (
    name <- 'IME_' + c.name ),
    collectionsClass : MVC!ModelClass
    (name <- 'IMC_' + c.name + '_Collections')
}
```

Figure 2: An example ATL tranformation rule

and interaction elements, and the navigation between pages through the Controller component.

The automatic transformation of a UWA conceptual model into the MVC-JSF model is based on a set of well defined mapping rules between the modeling primitives of the respective metamodels and the implementation of these rules with the Atlas Transformation Language (ATL). In brief, the UWA Information model maps to the MCD, while the UWA Navigation and Publishing Models merge into the VCD. Associations between the visual user interaction elements of the View pages and the attributes/methods of the MCD are derived from the Navigation Model. Table I reports an excerpt of the defined mapping rules, while Figure 2 reports the implementation of one of them with the ATL language (the rule refers to the transformation of UWA Collections).

IV. Case Study

In order to validate our approach, we applied it to redesign FilmUp.it (http://www.filmup.it), an Italian information portal which provides showtimes, reviews and multimedia contents on movies.

A. Reverse Engineering FilmUp.it

To initiate the redesign process, an instance of the FilmUp.it Web site was mirrored and around 13000 client-side HTML pages were downloaded. A preliminary similarity analysis was performed on these pages and 4500 of them were selected to cover the different sections of the Web site, with an average of 200 pages per section. According to the REUWA approach [1], these pages were analyzed using clone detection and clustering techniques to first identify clusters of similar pages (by considering their HTML structure), obtain a HTML page template from each cluster, and abstract from these templates UWA Entities, Semantic Associations and Collections.

The analysis identified 17 Entities (i.e., *Cinema*, *Producer*, *Actor*, *Biography*, *TV-Guide*, *Review*, *Newsv*, *Curiosity*, *Quiz*, *Trailer*, *Filmography*, *Character*, *Opinion*, *Poster*, *SoundTrack*, *Film*, *PhotoGallery*) and 53 Semantic Associations. Using the information on the identified clusters of pages and the abstracted UWA Entities and Semantic Associations, the analysis proceeded to identify UWA Navigation Nodes and Navigation Clusters (Structural, Association and Collection Clusters) which form the Navigation Model of the application. An Association Cluster was recovered for each validated Semantic Association. For each UWA Entity, a UWA Structural Cluster was generated (looking at the nodes generated from the clusters of pages containing that UWA Entity). The same was done for Collection Clusters. Finally, a simplified version of the UWA Publishing Model for FilmUp.it was recovered by associating a UWA Publishing Page to each cluster of pages, and by associating a Publishing Section and a Publishing Unit to each identified Navigation Node. To aggregate Sections and Units to Publishing Pages the information on which cluster of pages (or HTML page template) originated which Nodes was considered.

Figure 3 shows an excerpt of the recovered UWA models. In the upper part of the figure, three UWA Entities (*Film*, *Actor*, and *Trailer*) are shown along with their internal Components and Slots. Just below these Entities and on the left side, a UWA Semantic Association between the Entities *Film* and *Trailer* is showed along with the Association Center. On the right, the Collection *Trailers* modeling the list of incoming trailers is showed. In the middle of the diagram, two Navigation Clusters are showed: (*i*) the one related to the Semantic Association between *Film* and *Trailer*, (*ii*) and that associated to the Collection of *Next Trailers*. In the first of the two Clusters, the internal node labelled as *Film-Trailer* defines the selection of slots to be used as preview information in the navigation from a *Film* to its *Trailer*).

B. Generating the MVC-JSF Design of FilmUp.it

Figure 3 shows, on the right side, a portion of the MVC-JSF model automatically generated for the FilmUp.it WA by following the process described in Section 1 with the tools supporting this phase and taking as input the UWA models recovered by the RE phase.

The generated MCD includes the classes *IME_Actor*, *IME_Film*, and *IME_Trailer*, derived from the corresponding UWA Entities, and the associations between them, each

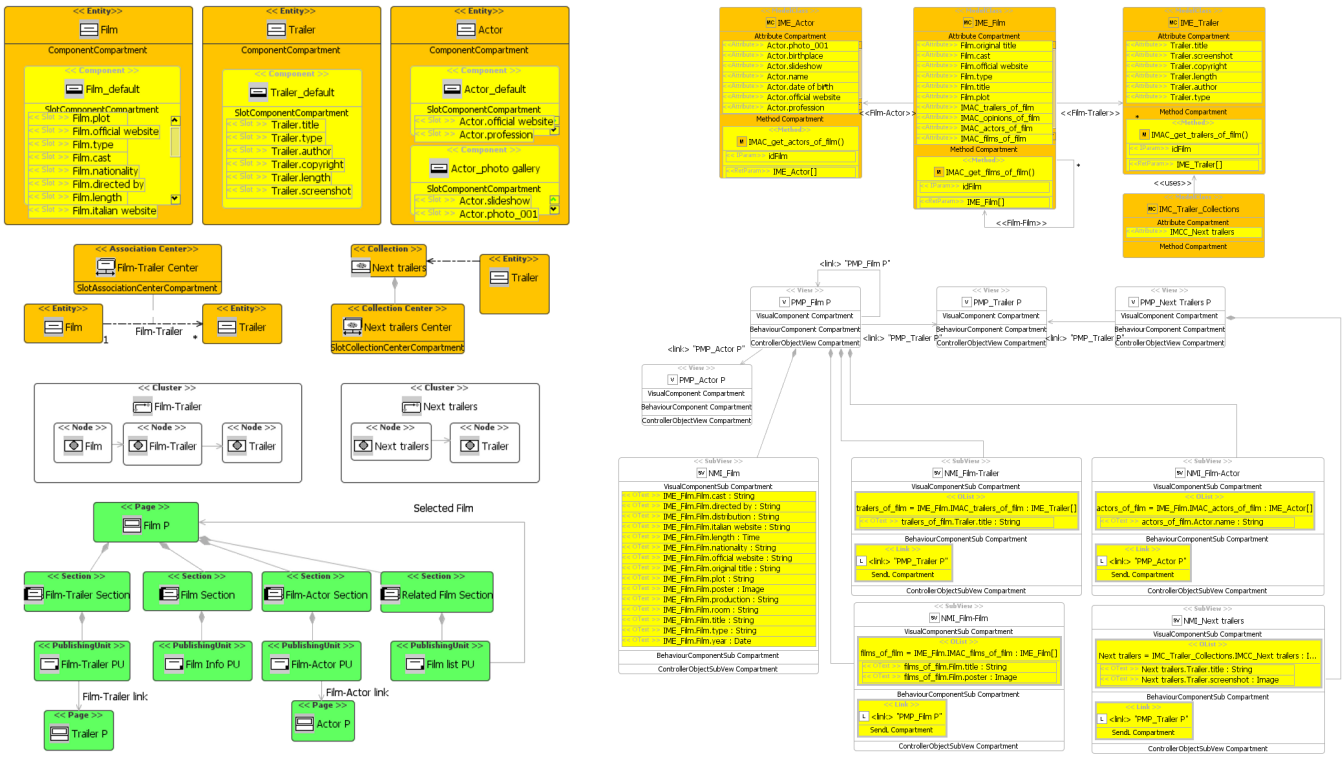

Figure 3: An excerpt of the UWA recovered models (left) and of the MVC-JSF models (right) for FilmUp.it

derived from a UWA Semantic Association. At the implementation level the three UML classes will result in three different JavaBeans with their own attributes and methods derived from the UWA Slots of the corresponding Entities. The prefixes IME, IMAC, etc., in the name of each class link it to the UWA conceptual model element from which the class was derived. As an example, the IME prefix in the name of the *IME_Film* UML class indicates that this class was generated from the *Film* Entity *Film* of the UWA Information Model. The VCD, below the MCD in Figure 3, represents the different views and sub-views of the redesigned application. Each view is represented as a UML class with the stereotype <<view>> and, similarly to classes in the MCD, it includes in its name a prefix that establishes a link to the UWA model element from which the view was derived. In the compartment associated to each View, the items identified by the <<oText>>, <<oList>>, <<iText>> and <<link>> stereotypes, relate directly to the attributes and or methods of the class in the MCD. In this way the Controller component of the MVC architecture is embedded in the diagram VCD. At the implementation stage, each of the views will be implemented by a JSF or JSP page with its visual and interaction elements.

C. Discussion of Results and Limitations of the Approach

The validation of the UWA models and the MVC-based design was carried out by comparing these models with the ones (assumed as a gold standard) produced by two software engineers expert of the application domain, the UWA methodology, and the MVC architectural pattern, that analyzed the WA and produced the (new) design 'by hand'. Due to space constrains, we can not provide details about this validation. However, the differences between the gold standard and the results produced by our semi-automatic redesign process were very limited: all the UWA Entities found by the experts were identified by REUWA, just few Semantic Associations (not more than 5%) identified by REUWA were rejected by the experts. Very few differences were found in the UWA Navigation Models extracted by REUWA and those defined by the experts. Similar considerations applied to the differences among the MVC-design produced by the experts and our approach. It is worthwhile to note that the time spent by the experts was about 4 times the one needed to apply the whole semi-automatic redesign approach by one software engineer, not included in the team of experts. Moreover, to assess the usefulness of the proposed approach we experimented to add two new Collections to the Web site: the *Most Visited Films* and the *Last Reviewed Films* (both not available in the original WA). By using our approach it was just a matter of extending the UWA recovered model in order to satisfy the new requirements. The MVC-JSF design model was then regenerated automatically by the transformation engine with no additional (nor manual) effort required. Also in this case the results were compared with those obtained by carrying out the same modifications on the original legacy WA: the time needed to fulfill the task was about 8 times greater than the time with the semi-automatic approach. This was due both to the increased comprehensibility of the WA

thanks to the availability of the recovered documentation, and to the greater flexibility of the new design and the tool support. This showed that the quality of a legacy WA can be actually improved by the proposed approach.

Another interesting point is on the technological side. While actually only a single MVC-based design targeting JSF platform has been defined, more platform specific models can be defined in order to migrate the application from a platform to another with very reduced effort.

The validation of the approach also highlighted some limits related to both the reverse engineering and the forward design processes. Concerning the reverse process, a limit is related to the impossibility to identify Entities, in an automatic way, when keywords are not detected in the analysed pages, by increasing the needed effort and requiring manual intervention. The metamodel underlying the forward design phase also need further improvements since it has been defined by specifically targeting the JSF implementation framework. We aim to evolve it in order to cover a wider spectrum of development platforms.

V. Conclusions

This paper presented an approach for the semi-automatic model-driven redesign of existing WAs. The approach relies on a reverse engineering phase to abstract UWA conceptual models from a WA by analyzing the client side HTML pages. The recovered models are used to produce a low level design model to re-implement the application adopting a MVC architectural pattern and the JSF technology. The main and new contribution of the proposed approach is that it applies a Web engineering methodology, such as UWA, both for the reverse engineering phase and the following forward design phase, and it integrates the two phases. As such it allows the semi-automatic redesigning of an existing WA, by recovering its embedded know how (the "as-is" conceptual model), and producing a new design based on the MVC architectural pattern, respectful of the original WA contents, relationships, and navigation among them, that can be further refined to improve the overall WA quality.

Future work will be devoted to improve the MVC-based model by supporting more implementation technologies and target platforms. Moreover experiments will be carried out to keep separated some common WA concerns (e.g., persistence, access control, customization and business logic) both in the modeling and in the code generation steps by applying MDD in the context of Aspect Oriented Software Development.

References

[1] M. L. Bernardi, G. A. D. Lucca, and D. Distante. The re-uwa approach to recover user centered conceptual models from web applications. *International Journal on Software Tools for Technology Transfer*, 11(6):485–501, 2009.

[2] S. Ceri, P. Fraternali, and A. Bongio. Web modeling language (webml): a modeling language for designing web sites. *Comput. Netw.*, 33(1-6):137–157, 2000.

[3] D. Distante, P. Pedone, G. Rossi, and G. Canfora. Model-driven development of web applications with uwa, mvc and javaserver faces. In *Proc. of the 7th International Conference on Web Engineering (ICWE'07)*, pages 457–472, Como, Italy, 2007. Springer Berlin / Heidelberg.

[4] R. J. E. Gamma, R. Helm and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[5] N. Koch and A. Kraus. The expressive power of uml-based web engineering. In *IWWOST'2002: Proc. of 2nd International Workshop on Web Oriented Software Technology*. Springer Verlag, 2002.

[6] C. Kroiss, N. Koch, and A. Knapp. Uwe4jsf: A model-driven generation approach for web applications. In *ICWE '09: Proc. of the 9th International Conference on Web Engineering*, pages 493–496, Berlin, Heidelberg, 2009. Springer-Verlag.

[7] M. P. L. Prechelt, B. Unger-Lamprecht and W. Tichy. Two controlled experiments assessing the usefulness of design pattern documentation in program maintenance. *IEEE Trans. Softw. Eng.*, 28(6):595–606, 2002.

[8] G. A. Di Lucca, A. R. Fasolino, F. Pace, P. Tramontana, and U. D. Carlini. Comprehending web applications by a clustering based approach. In *IWPC'02: Proc. of the 10th International Workshop on Program Comprehension*, page 261, Washington, DC, USA, 2002. IEEE Computer Society.

[9] G. A. Di Lucca, A. R. Fasolino, and P. Tramontana. Reverse engineering web applications: The ware approach. 16(1-2):71–101, 2004.

[10] OMG. Object management group (mof,mda,xmi,qvt,uml) - http://www.omg.org/.

[11] K. I. R. Sridaran, G. Padmavathi. A survey of design pattern based web applications. *Journal of Object Technology*, 8(2):61–70, 2009.

[12] F. Ricca and P. Tonella. Understanding and restructuring web sites with reweb. In *IEEE MultiMedia*, volume 8, pages 40–51, Los Alamitos, CA, USA, 2001. IEEE Computer Society.

[13] D. Schwabe and G. Rossi. An object oriented approach to web-based applications design. In *Theor. Pract. Object Syst.*, volume 4, pages 207–225, New York, NY, USA, 1998. John Wiley & Sons, Inc.

[14] S. Tilley, D. Distante, and S. Huang. Web site evolution via transaction reengineering. In *WSE'04: Proc. of the 6th IEEE International Workshop on Web Site Evolution (WSE)*, pages 31–40, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[15] UWA Project Consortium. *Deliverable D7: Hypermedia and Operation design: model and tool architecture*. UWA Project Consortium, 2001.

[16] UWA Project Consortium. Ubiquitous web applications. In *eBusiness and eWork Conference 2002*, 2002.

[17] J. Vanderdonckt, L. Bouillon, and N. Souchon. Flexible reverse engineering of web pages with vaquista. In *WCRE '01: Proc. of the 8th Working Conference on Reverse Engineering (WCRE'01)*, page 241, Washington, DC, USA, 2001. IEEE Computer Society.