AO-WAD: A Generalized Approach for Accessible Design within the Development of Web-based Systems

Adriana Martín[1,2]
[1]Department of Exact Sciences, Caleta Olivia
University of Patagonia Austral (UNPA-UACO)
Santa Cruz, Argentina
[2]GIISCo, Computer Science Department,
University of Comahue (UNCo)
Neuquén, Argentina
e-mail: adrianaelba.martin@gmail.com

Viviana Saldaño[1], Gabriela Miranda[1],
Gabriela Gaetán[1]
[1]Department of Exact Sciences, Caleta Olivia
University of Patagonia Austral (UNPA-UACO)
Santa Cruz, Argentina
e-mails: {vivianas / gmiranda /
ggaetan}@uaco.unpa.edu.ar //

***Abstract*— Web Engineering (WE) methods have evolved to support different concerns during the development process of current Web-based systems, as context-awareness, Business-to-Business (B2B) process modeling, Rich Internet Applications (RIAs) and live-regions or quality factors to improve users' experience. Therefore, developers have conceptual tools to focus on these concerns in advance, but unfortunately, the situation is not the same to early accessibility design. In this paper we provide a briefly overview of our proposal, called Aspect-Oriented Web Accessibility Design (AO-WAD), and generalize its use within some of the best known WE approaches to provide accessibility support through Aspect-Orientation techniques. We embed AO-WAD into OOHDM, UWE and OOWS methods and propitiate an ease understanding through a motivating example.**

Keywords- Web Accessibility; WE Approaches; UI Design; Aspect-Orientation.

I. INTRODUCTION

Nowadays, the advance of the Internet and the emerging technologies associated to the Web are universalizing information systems, allowing access to any connected potential user. The term "Web application" [1] refers to a new family of software applications specially designed due to the high growth of commercial activities on the Internet. These systems are being implemented in very short periods of time, without support of appropriate tools. For this reason, Web applications have low quality and very difficult maintenance. In most cases, development of Web application has been "ad hoc", lacking systematic approach, quality control and assurance procedures. Therefore, there is now great concern about how Web-based systems are developed to promote integrity and quality.

Web Engineering (WE), which is still an emerging discipline, encourages a process driven by systematic approaches to develop high quality Web-based systems. In the last decade, many WE methodological approaches, as Object-Oriented Hypermedia Design Method (OOHDM) [14], UML-based Web Engineering (UWE) [4], Object-Oriented Web Solution (OOWS) [3] and Web Site Design Method (WSDM) [16], have been proposed and evolved to provide support by means of abstract mechanisms that make easier the conceptualization and development of this kind of Web applications.

In contrast, the state-of-the-art shows that there are not many proposals for the early design with accessibility principles in mind and besides, even fewer proposals, provide conceptual tools to fully support accessibility nature to migrate to other WE approaches.

In general, a proposal for including accessibility design within systematic and unified Web development works only in association with a host WE approach. Therefore, there is a high dependence between host's process and deliverables and the proposed conceptual tools to support Web accessibility. The consequences are clear, since failing the design principle "low coupling" hinders embedding and easy connection with other WE approach. For example, Plessers et al. [13] is a well-known proposal that generates annotations for visually impaired users automatically from explicit conceptual knowledge existing during the WSDM [16] design process. The proposal prioritizes accessibility support using a rule-based mapping model to drive accessibility annotations, but by means of WSDM's modeling concepts to which these annotations are tightly bound. On the other hand, Moreno et al. [11] defines several constructs in UML meta-model to support the abstraction of Web accessibility concepts following the standard WCAG [18][19]. Thus, the proposal can be easily implanted into approaches following the MDA paradigm, but at expense of not fully addressing the non-functional, generic and "crosscutting" features of accessibility.

Our proposal for accessible design, called Aspect-Oriented Web Accessibility Design (AO-WAD) [6][7][9], recommends including accessibility concerns systematically within methods for Web application development. AO-WAD is born to join OOHDM [14] prioritizing accessibility at the very beginning of the Web design process. While OOHDM provides the main development framework, Aspect-Orientation ensures handling naturally the non-functional, generic and crosscutting characteristics of the accessibility concern within the framework.

At this point, let us define what is (and what is not) Web accessibility, and why it is a good idea to model its requirements as softgoals to be "satisficed" [2]. In short, Web accessibility is the ability to access the Web. However, you will never be perfectly accessible to everybody. From this point of view and since there is not a simple binary opposition between accessible and inaccessible [15],

ISBN: 978-1-61208-230-1

accessibility requires more loosely defined criteria, as the one proposed in [2] for non-functional requirements.

In this paper, we introduce AO-WAD as an example of having complete commitment to accessibility through Aspect-Orientation techniques without losing generality when developing within WE approaches. Supporting this statement, we develop a motivating example within OOHDM [14], UWE [4] and OOWS [3] as host methods, which are some of the most widespread and mature WE approaches.

This document is organized into eight sections as follows: in Section II we briefly introduce AO-WAD, while in Section III we explain the way our proposal provides accessibility support to OOHDM, UWE and OOWS Web development processes. Then, in Sections IV, V and VI we apply AO-WAD to a motivating example using as hosts these three WE approaches. In Section VII we achieve some insights about including accessibility design within Web development processes applying Aspect-Orientation techniques. Finally, in Section VIII we present the conclusions and future work.

II. AO-WAD IN A NUTSHELL

The model we envisage to deal with accessibility concerns within a WE approach is illustrated in Figure 1 [6]. Step 1 (Figure 1 (1)) manages Web application requirements looking for those that involve accessibility needs. This is because it is at the user's interface level where accessibility barriers finally show, so we are particularly interested in discovering accessibility requirements at the user interface (UI) design. Then, Step 2 (Figure 1 (2)) proposes an early capture of accessibility concrete concerns by developing two kinds of diagrams: the User Interaction Diagram (UID) with accessibility *integration points* [6] and the Softgoal Interdependency Graph (SIG) *template* [6] for Web Content Accessibility Guidelines (WCAG) 1.0. Step 3 (Figure 1 (3)) aids designers making decisions through the abstract UI model (Figure 1 (3.1)), and then, at Step 4 (Figure 1 (4)) toward its implementation through the concrete UI model (Figure 1 (4.1)). Thus, given a user's task, the SIG diagram provides the WCAG 1.0 accessibility checkpoints that "crosscut" the UI widgets (both, abstract and concrete ones; Figure 1 (3.1) and (4.1) respectively), to help to an accessible user experience.

Figure 1 (3) shows that at Step 3, our approach provides a supporting tool to assist developers in the implementation of cases, and on the creation of their corresponding models by using reusable components (for a detailed description of AO-WAD and its contribution to the area of accessible design see [6]).

In the following section, we show how AO-WAD can be implanted to work not only with OOHDM [14], but also with UML-based Web Engineering (UWE) [4] as one of the most popular and recognized Object-Oriented WE approaches.

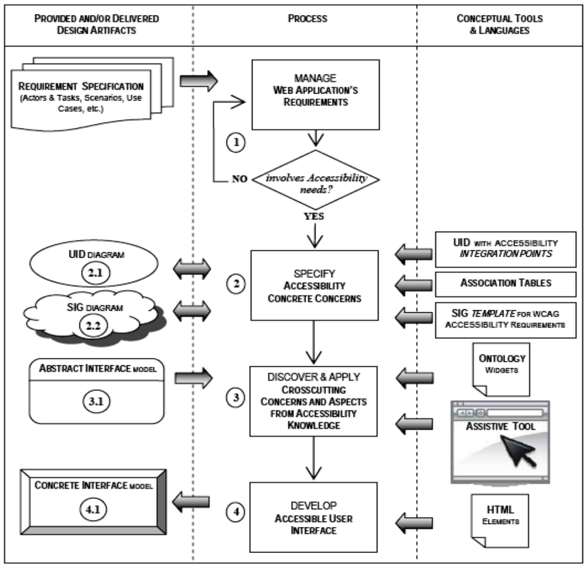

PROVIDED AND/OR DELIVERED DESIGN ARTIFACTS
PROCESS
CONCEPTUAL TOOLS & LANGUAGES
REQUIREMENT SPECIFICATION (Actors & Tasks, Scenarios, Use Cases, etc.)
MANAGE WEB APPLICATION'S REQUIREMENTS
1
NO
involves Accessibility needs?
YES
UID DIAGRAM
2.1
SIG DIAGRAM
2.2
2
SPECIFY ACCESSIBILITY CONCRETE CONCERNS
UID WITH ACCESSIBILITY INTEGRATION POINTS
ASSOCIATION TABLES
SIG TEMPLATE FOR WCAG ACCESSIBILITY REQUIREMENTS
ABSTRACT INTERFACE MODEL
3.1
3
DISCOVER & APPLY CROSSCUTTING CONCERNS AND ASPECTS FROM ACCESSIBILITY KNOWLEDGE
ONTOLOGY WIDGETS
ASSISTIVE TOOL
CONCRETE INTERFACE MODEL
4.1
4
DEVELOP ACCESSIBLE USER INTERFACE
HTML ELEMENTS

Figure 1. An overview of AO-WAD

III. SYSTEMATIC WEB DEVELOPMENT AND ACCESSIBILITY DESIGN

AO-WAD was developed in the spirit of model-driven paradigm to provide accessibility support within WE approaches. WE approaches are generally approaches as model-driven, because they address the different concerns involved in the development of a Web application using the following primary artifacts: (i) separate models (such as content, navigation and presentation), and (ii) model compilers to produce (semi) automated generation of most of the Web application's implementation from the original models [10]. AO-WAD focuses on preserving model-driven principles to enrich these artifacts (UI models and model transformations) with accessibility concerns. Thus, the integration of AO-WAD at the design level is immersed in a Web application development process.

In Section 2, we describe AO-WAD main process and interaction with OOHDM deliverables to model accessibility concerns in an Aspect-Oriented manner during Web developments. Figure 2 summarizes the embedding of AO-WAD within OOHDM Model-Driven Development process. The UID [17] is the conceptual tool used by OOHDM [14] to state transformations between Web application requirements (Use Case model) and the Conceptual, Navigation and UI models. AO-WAD propitiates the same principle between Web applications requirements and accessible UI models. The interaction between OOHDM models links and reinforces accessibility needs by applying two conceptual tools: the UID with *integration points* and SIG template for accessibility. The SIG diagram conveys the accessibility knowledge through WCAG 1.0 operationalizing softgoals [6] required to be applied at UI model. Due to accessibility nature, these accessibility softgoals "crosscut" the UI model more than once causing “crosscutting symptoms”. At this point, AO-WAD proposes to address these symptoms by modularizing softgoals into accessibility aspects. As Figure 2 shows, the deliverable of the process is an accessible and clean design, which means an OOHDM UI model enriched with accessibility concerns but free of “crosscutting symptoms”.

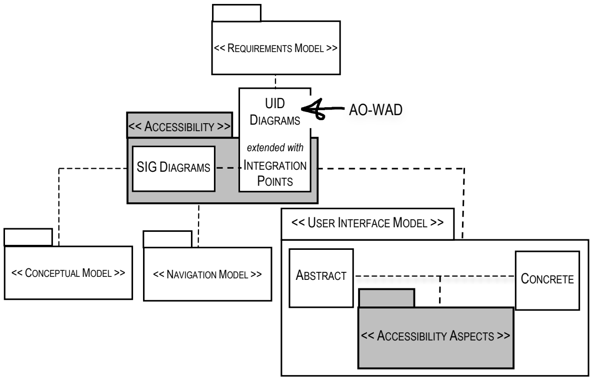


Figure 2. AO-WAD embedded into OOHDM Model-Driven Development process

As another good example of an established WE approach, UWE is based on OMG (modeling and metadata specifications) and uses UML for the analysis and design of Web applications. Figure 3 summarizes the embedding of AO-WAD within UWE Model-Driven Development process. In UWE [4], the Requirements model consists of two parts: (i) use cases of the Web application and their relationships and, (ii) activities describing use cases in detail. In particular, the Activity diagram is the conceptual tool used by UWE to describe more accurately each use case.

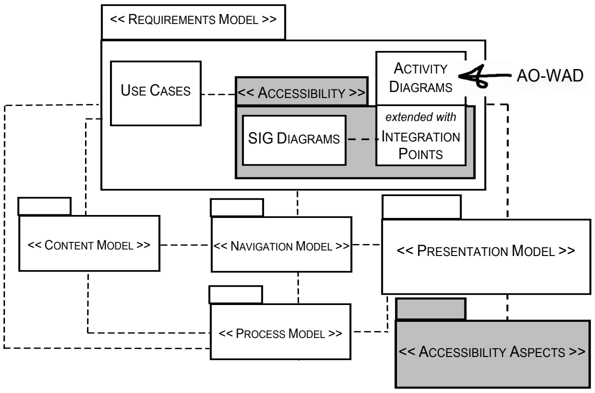


Figure 3. AO-WAD embedded into UWE Model-Driven Development process

UWE uses the Activity diagram to state transformations between Web application requirements and the Content, Navigation and Presentation models. Thus, as Figure 3 shows, AO-WAD embeds into UWE extending the Activity diagrams with integration points and through the SIG diagrams convey accessibility concerns as WCAG 1.0 operationalizing softgoals, which “crosscut” the Presentation model causing “crosscutting symptoms”. At this point and as we explained before, AO-WAD proposes to address these symptoms by modularizing softgoals into accessibility aspects. Figure 3 shows the deliverable of the process is an accessible and clean design, which means a UWE Presentation model with accessibility concerns but free of “crosscutting symptoms”.

OOWS extends an Object-Oriented software production method (called OO-Method [12]), for providing methodological support for Web application development. Figure 4 summarizes the embedding of AO-WAD within OOWS Model-Driven Development process. In OOWS [3], the Requirement model extends the OO-Method Task model to capture not only the structural and behavioral requirements (as happens in non-Web applications) but also navigational requirements using two extra diagrams: (i) a Task taxonomy, which hierarchically specifies the tasks that the users should achieve when interacting with the Web application, and (ii) a Task definition, which describes the interactions that users require from the Web application and the information that is exchanged in each interaction, using UML Activity diagrams and CRC cards [20], respectively. In particular, the OOWS Task definition model identifies and describes interaction points between the user and the Web application, which are very useful for our purpose.

As Figure 4 shows, AO-WAD can focus on this methodological support to embed into OOWS extending in first place, the Task definition model (Activity diagrams and CRC cards) with integration points and then, conveying accessibility concerns through the SIG diagram as WCAG 1.0 operationalizing softgoals. Again, AO-WAD proposes softgoals modularization into accessibility aspects to be injected into the Navigational and Presentation model.

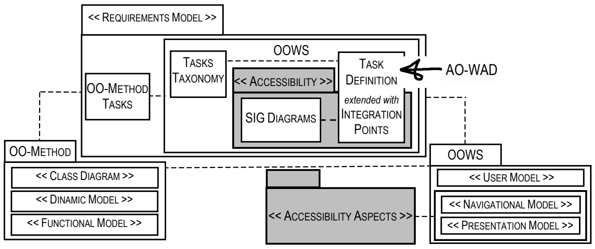


Figure 4. AO-WAD embedded into OOWS Model-Driven Development process

In order to ease understanding of AO-WAD within systematic Web development processes, we develop a motivating example in the following section, working with OOHDM, UWE and OOWS as host WE approaches.

IV. AN ACCESSIBLE UI FOR THE STUDENT’S LOGIN

We describe the embedding of AO-WAD within OOHDM, UWE and OOWS approaches using the following use case specification “Login a User given the User’s ID and Password”:

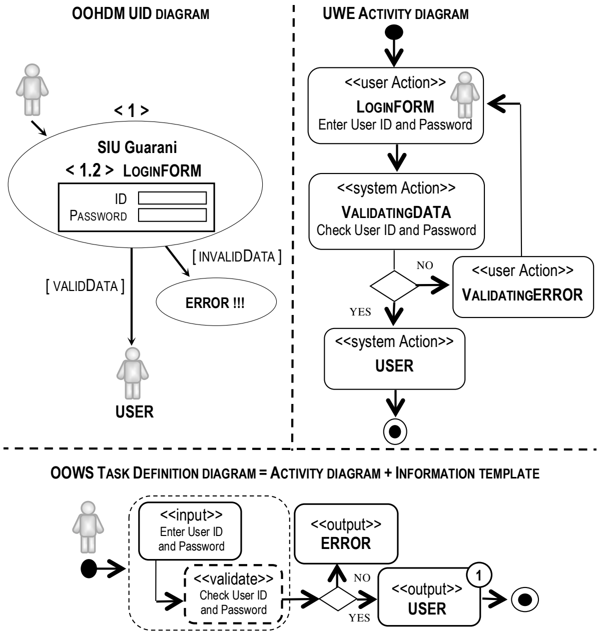
Use Case: Login a User given the Users ID and Password
Brief Description**:** This use case describes the User login
Primary Actor: User

Copyright (c) IARIA, 2012. ISBN: 978-1-61208-230-1

Description **Main Success Scenario**:	
Step	Action
1.	The system requests that the User enters his/her ID and Password.
2.	The User enters his/her ID and Password.
3.	The system validates the ID and Password and logs the User
Extensions:	
Step	Branching Action
3.a	The User enters an invalid ID and/or Password; the application displays an error message; the use case ends.

This example is simple but extremely representative mainly because of two reasons: (i) increasingly, business and government agencies are adopting a Web presence for sales and services to their customers, clients and citizens and, (ii) it clearly explains all of the issues concerning to accessible content that come into play when we think about how people with different capabilities interact with a Web page to input information [15]. The use case above describes the Web application's requirements for the user login and functionality that comprises user-application interaction; as we can see at the first step of the main success scenario, the user is requested by the application to enter his/her ID and Password. Since very often a specification based only on use cases is not enough [17], different kinds of refinement techniques are used to obtain a more detailed specification of functional requirements. OOHDM applies UID technique [17] to model user-system interactions and to specify the information that requires input from the user and choices that allow changes between interactions. UWE follows the principle of using UML whenever possible for specification and refines requirements with Activity diagrams for the main stream of the task to be performed. While OOWS, proposes the Task Taxonomy and Definition models [3] to capture Web application requirements, and in particular, the last one is the key model for specifying the interaction between the user and the Web application. Figure 5, illustrates the UID, the Activity diagram and also the Task Definition model), which provide a more detailed specification to the login use case within OOHDM, UWE and OOWS approaches, respectively. As we can see in Figure 5, an Activity diagram and an Information template implemented with the data technique CRC card, compound the OOWS Task Definition model.

As we already see in Section 2, looking at Step 1, AO-WAD proposes to examine the Web application requirements for the use case above, to identify accessibility concerns during the user-system interaction. It is clear in this specification that the FORM element is the key UI element to help achieve an accessible student's login. Following, in Sections V and VI we focus on modeling issues at Steps 2 (Figure 1 (2)) and 3 (Figure 1 (3)) respectively, as the main steps when implanting AO-WAD within WE approaches.



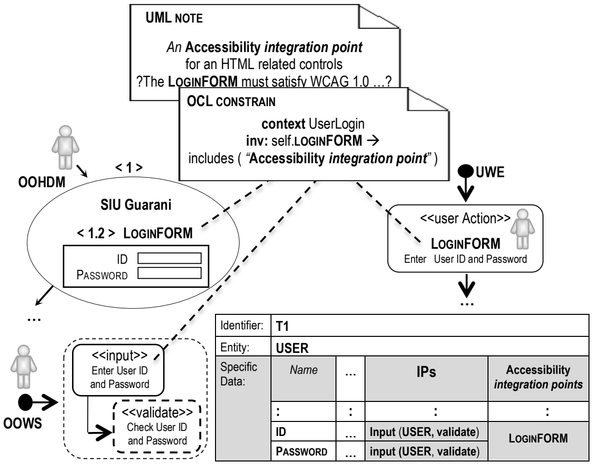
Identifier:	**T1**			
Entity:	**USER**			
Specific Data:	*Name*	*Description*	*Nature*	**IPs**
	Name	Name of the USER	String	**output** (**USER, 1**)
	:	:	:	:
	ID	ID of the USER	String	**input** (**USER, validate**)
	PASSWORD	Password of the USER	String	**Input (USER**, **validate**)

Figure 5. Requirements Specification with UID (left), Activity Diagram (right) and Task Definition Model (bottom)

V. SPECIFYING ACCESSIBILITY CONCRETE CONCERNS

When developing with OOHDM, AO-WAD proposes at Step 2.1 extending the UID diagram with integration points to support an early registration of accessibility concerns. This conceptual tool attaches an accessibility integration point to each one of those UI elements with impact on the dialog required by the use case functionality and modeled by the UID. Looking for the same modeling purpose, AO-WAD Step 2.1 can be also satisfied when developing with UWE, and OOWS extending the Requirements model of these WE approaches. When developing with UWE, the Activity diagram is enriched with accessibility *integration points*. Likewise in OOWS, the Task Definition model provides user-application interaction points (IPs) enabling AO-WAD accessibility *integration point* to be easily attached. Figure 6 illustrates the UID (left), the Activity diagram (right) and the Task Definition model (bottom), extended with an *integration point* that allows an early record of accessibility concerns for the FORM UI element -- i.e. HTML related controls. Also, Figure 6 shows two possible ways of attaching the accessibility *integration points* to these diagrams and model: (i) including a UML Note modeling construct or (ii) defining an Object Constraint Language (OCL) expression. As Figure 6 (bottom) shows, OOWS aids the Activity diagram with an

Information template, whose CRC card can be also extended to reinforce the specification of the accessibility *integration point* for the FORM UI element. As we see, the integration of Step 2.1 proposed by AO-WAD into the Requirement model of WE approaches is straightforward.


UML NOTE
An Accessibility integration point
for an HTML related controls
?The LOGINFORM must satisfy WCAG 1.0 ...?
OCL CONSTRAIN
context UserLogin
inv: self.LOGINFORM →
includes ("Accessibility integration point")
OOHDM
< 1 >
SIU Guarani
< 1.2 > LOGINFORM
ID
PASSWORD
...
UWE
<<user Action>>
LOGINFORM
Enter User ID and Password
...
OOWS
<<input>>
Enter User ID and Password
<<validate>>
Check User ID and Password

Identifier:	T1			
Entity:	USER			
Specific Data:	*Name*	...	IPs	Accessibility *integration points*
	:	:	:	:
	ID	...	Input (USER, validate)	LOGINFORM
	PASSWORD	...	input (USER, validate)	

Figure 6. UID (left), Activity Diagram (right) and Task Definition Model (bottom) with Accessibility *integration points*

Then, AO-WAD proposes at Step 2.2, the specification of accessibility softgoals through a SIG tree. When developing with OOHDM, the SIG diagram is a consequence of instantiating the SIG template taking the UID with integration points as input --i.e the early registration of accessibility concerns for the FORM UI element, shown by Figure 6 (left), which is core to the required functionality. The SIG diagram specifies accessibility operationalizing softgoals to be satisfied for reaching the required WCAG 1.0 level of compliance. Applying the same modeling purpose, the Activity diagram extended with *integration points*, shown by Figure 6 (right and bottom, respectively), provides the required input for developing the SIG diagram within UWE and OOWS. Although the SIG *template* is not a UML specification tool, it can be easily transformed into an XML tree structure and work with other UML diagrams within the philosophy of the model-driven paradigm. Therefore, there are no major problems for including Step 2.2 proposed by AO-WAD during the development process of WE approaches under consideration.

VI. SOLVING ACCESSIBILITY CROSSCUTTING SYMPTOMS

AO-WAD proposes at Step 3, the specification of accessibility aspects to avoid "crosscutting symptoms" resulting from applying accessibility operationalizing softgoals to elements comprising the UI model. At the UI modeling stage, OOHDM delivers an Abstract UI model [14] whose vocabulary is established by the Abstract Widget Ontology extended by AO-WAD [6] to support new elements required by current UI, which are dynamic and with a high degree of complexity. Similarly, UWE delivers a Presentation model [4] from a Meta-model for modeling UI elements. Presentation requirements are specified in OOWS using a Presentation model that is strongly based on the Navigational model and uses the navigational contexts as basic entities to define the presentation properties; working together, these models capture the essential requirements for the construction of Web UI [3].

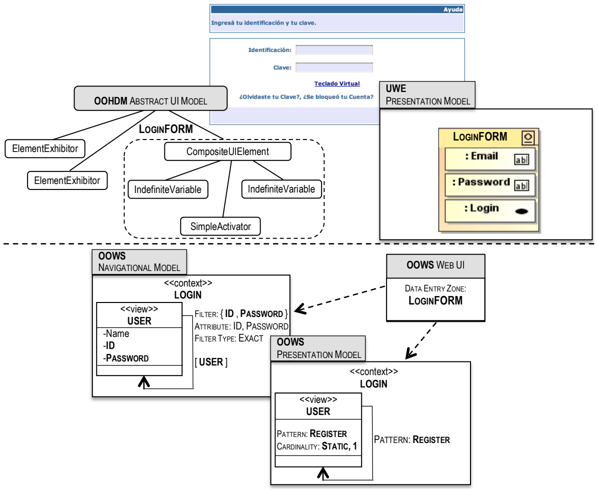


Figure 7. Abstract Interface Model in OOHDM (left), Presentation mModel in UWE (right) and Navigation-Presentation Models and Web UI in OOWS

Figure 7 shows the Abstract UI model delivered by OOHDM (left), the Presentation model provided by UWE (right), and the Navigation-Presentation models and Web UI provided by OOWS (bottom), for the screenshot (top) corresponding to the login example. In first place, AO-WAD recommends discovering "crosscutting symptoms" that manifest when applying accessibility operationalizing softgoals to the UI model --i.e. OOHDM Abstract Interface model, UWE Presentation model and OOWS Navigational-Presentation models and Web UI. These operationalizing softgoals are spread out and intermixed through the components of the login FORM UI element, causing "scattering" and "tangling" symptoms. Then, AO-WAD prescribes eliminating these symptoms through a modularization process that applies aspects to provide accessibility support at the user's technology and layout. Thus, aspects modularize operationalizing softgoals to be satisfied for properly convey the accessibility concerns required by UI elements. As Figure 8 depicts through a pseudo code, Aspect-Orientation provides a mechanism called "weaving", which requires that each aspect must specify "where or how" should be invoked and "what" should be injected into the core --i.e. a concrete UI model.

VII. DISCUSSING WE APPROACHES FROM THE ACCESSIBILITY PERSPECTIVE

We have been working for a while on accessibility [5] and particularly on accessibility design at early stages of Web applications development [6][7][8][9]. Particularly, we have been applying concepts from Aspect-Orientation in association with the WCAG 1.0 document to deal with accessibility concerns within WE approaches.

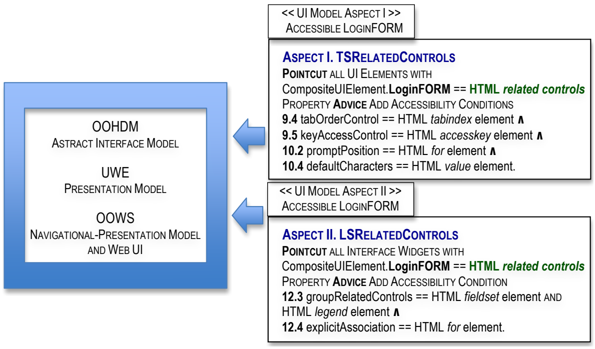

OOHDM
ASTRACT INTERFACE MODEL
UWE
PRESENTATION MODEL
OOWS
NAVIGATIONAL-PRESENTATION MODEL AND WEB UI
<< UI MODEL ASPECT I >>
ACCESSIBLE LOGINFORM
ASPECT I. TSRELATEDCONTROLS
POINTCUT ALL UI ELEMENTS WITH
CompositeUIElement.LoginFORM == HTML related controls
PROPERTY ADVICE ADD ACCESSIBILITY CONDITIONS
9.4 tabOrderControl == HTML tabindex element ∧
9.5 keyAccessControl == HTML accesskey element ∧
10.2 promptPosition == HTML for element ∧
10.4 defaultCharacters == HTML value element.
<< UI MODEL ASPECT II >>
ACCESSIBLE LOGINFORM
ASPECT II. LSRELATEDCONTROLS
POINTCUT ALL INTERFACE WIDGETS WITH
CompositeUIElement.LoginFORM == HTML related controls
PROPERTY ADVICE ADD ACCESSIBILITY CONDITION
12.3 groupRelatedControls == HTML fieldset element AND
HTML legend element ∧
12.4 explicitAssociation == HTML for element.

Figure 8. Specification of Accessibility Aspects conveying Accessibility Concerns

Since the model-driven paradigm provides a good framework to develop for the Web 2.0, we believe that a proposal to somehow improve the users experience should be able to work within any WE approaches. Although AO-WAD is conceived within OOHDM to fully address accessibility features, its application can be generalized to work with other approaches, such as UWE and OOWS methods. The process proposed by AO-WAD (Figure 1) can be normalized to handle accessibility concerns through two conceptual tools: the accessibility *integration points* and SIG *template* techniques. These tools are core to AO-WAD generalization since they provide the required support to manage accessibility concerns within any WE development processes. The accessibility *integration points* technique provides early registration of accessibility concerns, while the SIG *template* technique allows instantiation for specifying concrete WCAG operationalizing softgoals to be applied [6]. These diagrams can be easily implanted into WE Requirements models, such as UIDs in OOHDM, Activity diagrams in UWE and Task Definition model in OOWS (Figures 5 and 6). Then, crosscutting symptoms are solved by the modularization of WCAG operationalizing softgoals into accessibility aspects to enrich WE Navigational and Presentation/UI models, such as Abstract UI in OOHDM, Presentation model in UWE and, Navigational-Presentation models and Web UI in OOWS (Figures 7 and 8). So, a first step in the normalization of AO-WAD for its generalization can be synthetized as follow: (1) extending requirements with accessibility *integration points*, (2) specifying the SIG diagram and, (3) modularizing WCAG operationalizing softgoals into accessibility aspects to be injected. Finally, since AO-WAD is developed to work with the model-driven paradigm, we would like to highlight advantages/disadvantages of this paradigm and how benefits/affects AO-WAD. On one hand, applying systematic and unified model-driven approaches brings the benefit of having full documentation and automatic application generation at the expense of introducing some bureaucracy into the development process. Since our proposal suggests the early treatment of the accessibility concerns through models, we may still be influenced by this reality and its disadvantages --i.e., time and cost consuming, complexity, learning effort, etc. On the other hand, using models and taking advantages of an iterative and incremental development process to deal with accessibility concerns, allows: (i) going back from UI models to Navigation models to look for alternatives in the navigation path, (ii) assessing the need and relevance of these alternatives to the functionality under develop, and (iii) going forward from Navigation models to UI models to check the accessibility of the UI related to these alternatives. Thus, the accessibility of all the alternative navigation paths that may compromise the desired functionality can be evaluated within AO-WAD.

AO-WAD supports accessible Web applications design by embedding Aspect-Orientated techniques into WE development approaches to proper address the non-functional, generic and "crosscutting" features of the accessibility nature.

VIII. CONCLUSIONS AND FUTURE WORK

The application of the model-driven paradigm to the domain of Web development has resulted in well-known WE approaches, which can be particularly useful because of the continuous evolution of Web 2.0 applications, technologies and platforms. The new generation of Web applications must offer user interfaces that enhance the experience and access to all Web users. In this context, we believe that WE approaches provide suitable models to carry with the improvements required by the application under development. In this paper we briefly introduce AO-WAD, which provides complete support to accessibility concerns by enriching WE models. Following OOHDM, UWE and OOWS processes, in this work we focus our efforts on the generalization of AO-WAD. We show that AO-WAD is flexible enough to be embedded within any WE approach, and therefore this can be a starting point that propitiates industry adoption.

As future work, we will continue working to complete the normalization of AO-WAD and validate its generalized use to systematic developing of accessible Web applications. Since, UWE and OOWS approaches provide tools for partial/full automating their design and/or implementation stages, we will analyze also the interaction of AO-WAD with these tools.

REFERENCES

[1] Baresi L., Garzotto F., and Paolini P. From Web Sites to Web Applications: New Issues for Conceptual Modeling. ER'2000 Workshop on Conceptual Modeling and the Web, LNCS 1921.Springer-Verlag, 2000.

[2] Chung, L. and Supakkul, S.: Representing FRs and NFRs: A Goal-Oriented and Use Case Driven Approach. SERA (2004) doi:10.1007/11668855_3

[3] Fons, J., Pelechena, V., Pastor, O., Valderas, P., and Torres, V. Applying the OOWS Model-Driven Approach for Developing Web Applications. The Internet Movie Database Case Study. In: Rossi, G., Pastor, O., Schwabe, D., Olsina, L. (eds.) WE, pp. 65-108. Springer (2008)

[4] Koch, N., Knapp, A., Zhang, G., and Baumeister, H.: UML-based Web Engineering: An Approach based on Standards. In: Rossi, G., Pastor, O., Schwabe, D., Olsina, L. (eds.) WE, pp. 157–191. Springer (2008)

[5] Martín, A., Cechich, A., and Rossi, G.: Comparing Approaches to Web Accessibility Assessment. In: Calero, C., Moraga, M.Á., Piattini, M. (eds.) Handbook of research on Web information systems quality, pp. 181–205. Information Science Reference, Hershey (2008)

[6] Martín, A., Rossi, G., Cechich, A., and Gordillo, S. Engineering Accessible Web Applications. An Aspect-Oriented Approach. World Wide Web Journal, 13(4), 2010, 419-440 doi:10.1007/s11280-010-0091-3

[7] Martín, A., Mazalú, R., and Cechich, A. Supporting an Aspect-Oriented Approach to Web Accessibility Design. ICSEA (2010), Francia, doi:10.1109/ICSEA.2010.10

[8] Martín A., Cechich, A., and Rossi, G. Accessibility at Early Stages: Insights from the Designer Perspective. W4A (2011), India, doi: 10.1145/1969289.1969302

[9] Mazalú, R., Huenuman, F., Martín, A., and Cechich, A. AO - WAD: A Supporting Tool to Aspect-Oriented Web Accessibility Design. ASSE (2011), Argentina.

[10] Moreno, N., Romero, J., and Vallecillo, A. An Overview of Model-Driven Web Engineering and the MDA. In: Rossi, G., Pastor, O., Schwabe, D., Olsina, L. (eds.) Web Engineering: Modelling and Implementing Web Applications. pp. 109-155. Springer-Verlag, London (2008)

[11] Moreno, L., Martinez, P., and Ruiz, B. A MDD Approach for Modeling Web Accessibility. WOST (2008), USA, doi:10.1.1.163.9478

[12] Pastor, O., Gómez, Insfrán, E., and Pelechano, V.: The OO-Method Approach for Information Systems Modeling: From Object-Oriented Conceptual Modeling to Automated Programming. Inf. Syst. 26(7): 507-534 (2001)

[13] Plessers, P., Casteleyn, S., Yesilada, Y., De Troyer, O., Stevens, R., Harper, S., and Goble C.: Accessibility: A Web Engineering Approach. WWW (2005) doi:10.1145/1060745.1060799

[14] Rossi, G. and Schwabe, D.: Modeling and Implementing Web Applicactions with OOHDM. In: Rossi, G., Pastor, O., Schwabe, D., Olsina, L. (eds.) WE, pp. 109–155. Springer (2008)

[15] Thatcher, J., Burks, M., Heilmann, Ch., Henry, S., Kirpatrick, A., Lauke, P., Lawson, B., Regan, B., Rutter, R., Urban, M., and Waddell, C.: Web Accessibility - Web Standards and Regulatory Compliance. Friendsof ED, USA (2006)

[16] De Troyer, O., Casteleyn, S., and Plessers, P.: WSDM: Web Semantics Design Method. In: Rossi, G., Pastor, O., Schwabe, D., Olsina, L. (eds.) WE, pp. 303–351. Springer (2008)

[17] Vilain, P., Schwabe, D., and Sieckenius de Souza, C.: A Diagrammatic Tool for Representing User Interaction in UML. UML (2000) doi:10.1007/3-540-40011-7_10

[18] W3C: Web Content Accessibility Guidelines 1.0. (WCAG 1.0). http://accessibility.w3.org/TR/WAI-WEBCONTENT/ (1999). Accessed 15 April 2009

[19] W3C: Web Content Accessibility Guidelines 2.0 (WCAG 2.0). http://accessibility.w3.org/TR/WCAG20/ (2008). Accessed 25 January 2010.

[20] Wirfs-Brock, R., Wilkerson, B., and Wiener, L. Designing Object–Oriented Software. Prentice–Hall, 1990.