2. ACCESSIBILITY WITHIN WE APPROACHES

2.1 Web Accessibility

Generally speaking, in the World Wide Web (WWW), where users have the freedom to choose what best meets their expectations, the quality of a user interface (UI) can make the difference between maintaining the Web site competitiveness (or not) within its domain --e.g. e-Business and B2B[10], e-Education (e-Teaching and e-Learning), e-Government, GIS[11] (GeoWeb, Web Mapping and Web GIS), etc., and even compromise the Web site survival.

In May 2006 foreword by Molly Holzschlag said [41]:

"…Berners-Lee's vision has always had to do with the human side of the Web. After all, it's not machines that use the Web, but people… Accessibility is not about disabilities; rather, it's about people getting to shared information that the vision of the Web has made manifest…"

Web Accessibility is dedicated to achieving the access to the Web by everyone, regardless of their permanent or temporary disabilities, age-related problems, generational gaps, personal skills and preferences, culture and developed education, etc. While it is true that Web Accessibility emerged initially to help accessing the Web to people with disabilities, currently there is no doubt about the spectrum of benefits that Accessibility provides to the universe of Web users. In this thesis, we have chosen not to provide several definitions of Web Accessibility, as is usually done to describe its

[10] Business to Business (B2B) also known as e-Biz, is the exchange of products, services, or information between businesses rather than between businesses and consumers.

[11] A Geographic Information Systems (GIS) is a system of hardware and software used for storage, retrieval, mapping, and analysis of geographic data. GeoWeb consists of location-aware Web technologies usually manifested on the WWW; Web Mapping then refers to those online applications that permit users to view or create maps on a Web platform, usually with limited or no GIS analysis; while Web GIS then refers to GIS that use Web technologies as a method of communication between the elements of a GIS.

scope and contributions (these definitions are all available at the Internet[12]). Instead, we prefer to introduce Table 2.1 that clearly shows how Accessibility can help all users to face accessing the Web at different life situations; after all, we all have different skills and abilities.

Table 2.1: Web Accessibility benefits the entire universe of Web Users

Disability	**People with Disability**	**People "without Disability"**
Vision	Blinds	Users who are driving in the dark…
Low vision	Low-vision Users	Users who are using a device with a small display...
Hearing	Deafs	Users who are in forced silence (library) or using music players with headphones…
Low hearing	Low-hearing Users	Users who are in noisy environments…
Motor impaired	Motor impaired Users due to illness or traumatic injuries (permanent or temporary)	Users who are wearing tight clothes, protective clothing, overalls, workware… Users on a moving and/or unstable vehicle --e.g. a train...
Cognitive impaired	Users who are limited in their abilities to process and memorize information, to take decisions, to learn, to performe intellectual tasks.	Users who are tired, fatigued, distracted, worried, sleepy, drunk...
Communicational impaired	Users having difficulties to understand linguistic and textual.	Users who have no knowledge of the language, slogans or symbols...

The Word Wide Web Consortium (W3C) is one of the main referents of Web Accessibility and has worked for more than ten years in the development of a standard called Web Content Accessibility Guidelines (WCAG[13]), which is considered a benchmark for most of the laws on Information Technology and Communication worldwide. The WCAG has two documents, the WCAG 1.0 [45] and the WCAG 2.0

[12] W3C (2005) definition at http://www.w3.org/WAI/intro/accessibility.php; ISO/TS 16071 (2003) definition at http://www.iso.org/iso/catalogue_detail.htm?csnumber=30858; Hull (2004) definition at http://ausweb.scu.edu.au/aw05/papers/refereed/arora/paper.html; Fourney and Carter (2006) definition at http://userlab.usask.ca/papers/IEA06DF-JC.pdf; etc.

[13] WCAG overview at http://www.w3.org/WAI/intro/wcag

[46], whose stable specifications were released in 1999 and 2008 respectively. Since their longstanding presence in the Accessibility arena, the WCAG 1.0 has provided the basis for the promulgation of other Accessibility standards and legislation in several countries. For example, this is the case for the US Section 508 [38], the UK PAS 78 [34] and the Italian Legislation on Accessibility [40]. Currently, the migration process from WCAG 1.0 to WCAG 2.0 of these standards and legislation is taking place. In Argentina, Web Accessibility is an issue that has been recently included in the State's agenda. The legislation 26.653 called “Guía de Accesibilidad para Sitios Web del Sector Público Nacional[14]”, which adheres to WCAG 1.0 document, was approved by Resolution 69/2011 on June 27th 2011. In August 2011, Argentina became a member of the W3C[15]. We will return on WCAG and its documents in Section 4.6, and then also in Section 7.3.1 where we will explain how we carry out the migration of our proposal.

Since 1999, when the first W3C Accessibility document was released, a number of tools and approaches have emerged and are available to support Web developers evaluating Accessibility of existing Web applications. However, Accessibility has not yet gained enough recognition as a crucial non-functional requirement such as other quality factors. This situation may be due to several reasons, but probably, it had much to do with the way Accessibility was first introduced to Web developers --i.e. by showing only its side committed with disability. This lack of knowledge within developer’s community, prevented them from getting involved with the cause, and as a consequence, the work has been addressed mostly by Accessibility specialists and entities engaged with disability. As we shall see next in Section 2.2, the status is worse from a design perspective, since it is a fact that there are not many efforts considering Accessibility at early stages of the development process.

At this point, we would like to perform some considerations concerning to the relationship between Accessibility and Web development stages. As we already said in Chapter 1, Web Engineering (WE) focuses on stages, which create and exploit domain models, to face the development life cycle of Web applications. Almost every mature

[14] Access to Public Information by Law 26.653 at http://www.infoleg.gov.ar/infolegInternet/anexos/175000-179999/175694/norma.htm

[15] Argentina became a member of the W3C at http://www.puntogov.com/nota.asp?nrc=2641

WE method proposes the following five stages, each one delivering its respective model: requirements, conceptual, navigation, user interface and implementation. In the best cases, Accessibility is submitted to user interface (UI) codification and implementation stage. In most cases, Accessibility is addressed when the application is already fully developed, and in consequence the process of making this application accessible involves significant redesign and recoding, which may be considered outside the project's scope and budget [22].

Finally, when we talk about Web Accessibility, we must specify the target of the Accessibility efforts since to establish the client-server Web relationship, several components are required. This means that Web Accessibility depends on these components working together and improvements in specific components could substantially improve Web Accessibility. Thus, for example, we can evaluate the Accessibility of the following components: (i) User agents, client devices or assistive technologies, such as PCs and notebooks, cell phones, iPods and iPads, screen readers[16], screen magnifiers[17], braille keyboards[18], PDAs, etc., (ii) Web browsers, such as Safari, Mozilla Firefox, Internet Explorer, Opera, etc., (iii) Authoring tools –i.e. software that helps creating Web sites[19], (iv) Web pages --i.e. the content, structure, presentation and layout of Web documents and (v) Web navigation --i.e. how the Web user moves from one Web page to another when traveling through the cyberspace. The W3C-WAI provides valuable standards to improve the Accessibility of these components[20] that are

[16] Software for the visually impaired users that reads the contents of a computer screen, converting the text to speech.

[17] A screen magnifier is software that interfaces with a computer's graphical output to present enlarged screen content.

[18] Portable units used to take notes using the Braille system; quite often use chording techniques (key combinations), but some units are designed with a traditional keyboard.

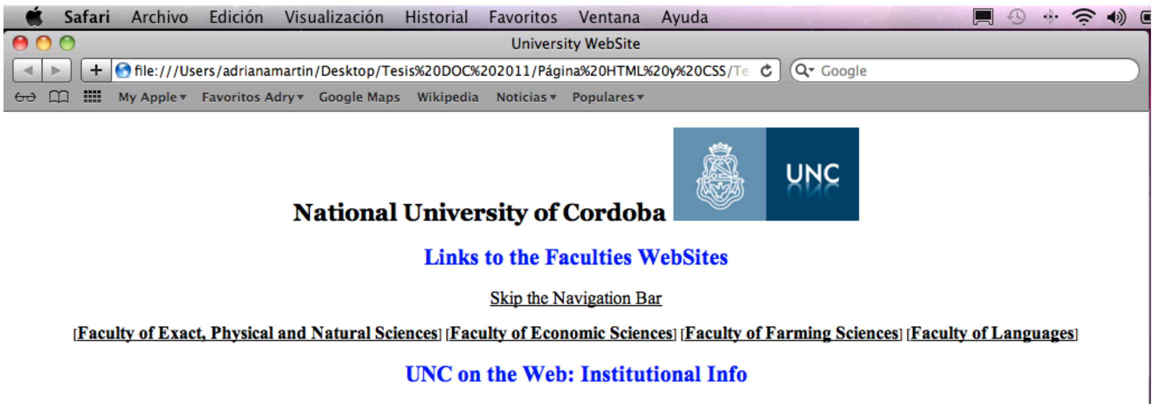
[19] A list of some Authoring tools and their comparison at http://www.edb.utexas.edu/minliu/multimedia/Compare%20Web%20Authoring%20Tools.pdf

[20] W3C-WAI guidelines and techniques at http://www.w3.org/WAI/guid-tech.html

called “Essential Components of Web Accessibility”[21]. As examples of these standards, we already mentioned the WCAG documents [45] [46], which are focused on explaining how to make accessible the Web content component and, the User Agents Accessibility Guidelines (UAAG) [48] document[22], which provides guidelines for designing user agents that lower barriers to Web accessibility for people with disabilities. As we are especially interested in developing accessible Web applications, our work focuses its efforts on designing user interfaces (UI) by applying the WCAG recommendations to propitiate better access to content, help navigation and improve the user experience while interacting with the application.

2.2 Proposals for Developing Accessible Web Applications

This section reviews the most relevant proposals that aim to consider the Accessibility concern in at least, some of the stages of the development life-cycle. To provide a more complete description and also to perform a more thorough analysis of these proposals, in Section 2.2.1 we introduce a case study that we use to apply each one of them.


Safari Archivo Edición Visualización Historial Favoritos Ventana Ayuda
University WebSite
file:///Users/adrianamartin/Desktop/Tesis%20DOC%202011/Página%20HTML%20y%20CSS/Te
Google
My Apple Favoritos Adry Google Maps Wikipedia Noticias Populares
UNC

National University of Cordoba

Links to the Faculties WebSites

Skip the Navigation Bar

[Faculty of Exact, Physical and Natural Sciences] [Faculty of Economic Sciences] [Faculty of Farming Sciences] [Faculty of Languages]

UNC on the Web: Institutional Info

Figure 2.1: A simplified University home page example

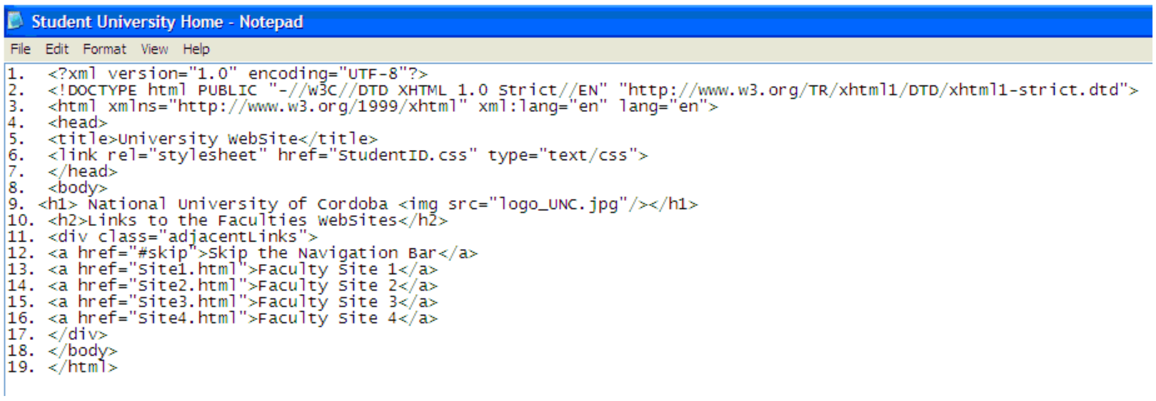
[21] W3C-WAI: strategies, guidelines, resources to make the Web accessible to people with disabilities at http://www.w3.org/WAI/intro/components.php

[22] UAAG overview and UAAG 2.0 working draft at http://www.w3.org/WAI/intro/uaag.php

2.2.1 Providing a Student of his/her Faculty Site

In this section we present the typical situation faced by a college student when looking for his/her respective Faculty site. Let us assume that the student enters the home page of the University of which depends the desired Faculty and this home page has the appearance illustrated in Figure 2.1.

As we can see in Figure 2.1 the page offers the student a set of related links to the Faculties that make up the University. The name of each Faculty is an anchor the student can use to browse to his/her Faculty site. Since links are navigation mechanisms that create a set of paths a user may take through a site, it is very important to keep a consistent style of presentation for links, as for every interface of components relevant to the interaction interface-functionality. Thus, taking into account Accessibility recommendations for links will allow users to locate and skip navigation mechanisms more easily to find important content. This helps people with learning and reading disabilities but also makes navigation easier for all users. Predictability will increase the likelihood that people will find information at your site, or avoid it when they so desire [45]. Returning to the University home, Figure 2.2 illustrates the corresponding HTML code for this page example.



```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
3.  <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
4.  <head>
5.  <title>University WebSite</title>
6.  <link rel="stylesheet" href="StudentID.css" type="text/css">
7.  </head>
8.  <body>
9. <h1> National University of Cordoba <img src="logo_UNC.jpg"/></h1>
10. <h2>Links to the Faculties WebSites</h2>
11. <div class="adjacentLinks">
12. <a href="#skip">Skip the Navigation Bar</a>
13. <a href="Site1.html">Faculty Site 1</a>
14. <a href="Site2.html">Faculty Site 2</a>
15. <a href="Site3.html">Faculty Site 3</a>
16. <a href="Site4.html">Faculty Site 4</a>
17. </div>
18. </body>
19. </html>
```

Figure 2.2: The HTML code for the University home page example

As we can see at lines 12, 13, 14, 15 and 16 of Figure 2.2, a set of five HTML *a* elements is defined for a "skip" option and four Faculties, and they are enclosed with an HTML *div* element at lines 11 and 17 of the styling *class* "adjacentLinks". Following, we use this simple example to discuss the way the five approaches cited at this chapter work for improving more accessible user interface designs.

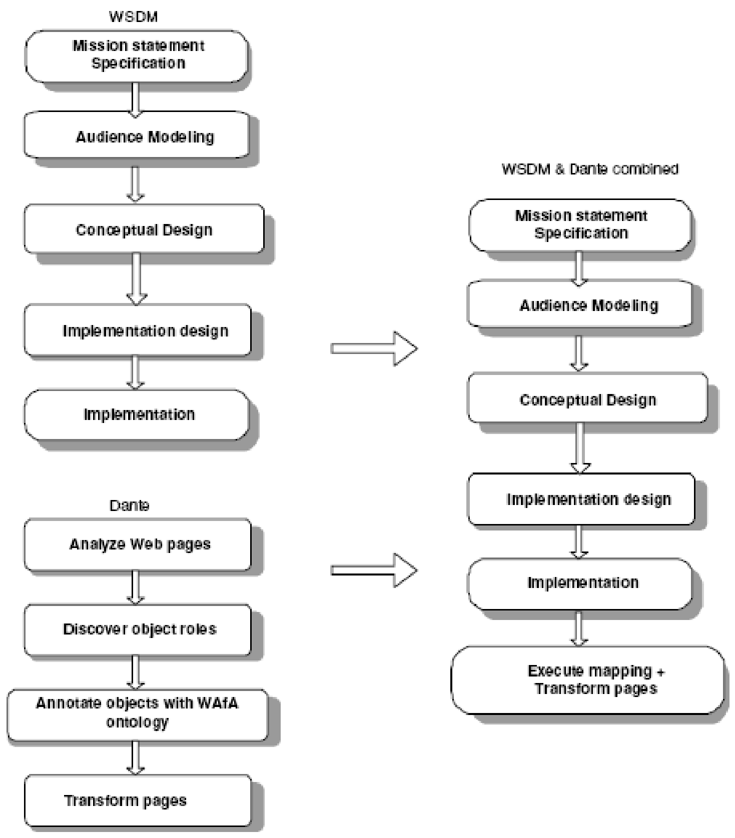


Figure 2.3: The WSDM with Dante from [51]

2.2.2 Automatic Annotations for Accessibility

The main goal in Plessers et al. [35] is to generate annotations for visually impaired users automatically from explicit conceptual knowledge existing during the design process. The approach integrates the Dante [52] annotation process into the Web Site Design Method (WSDM) [13] that allows Web sites and Web applications to be developed in a systematic way. The annotations are generated from explicit conceptual knowledge captured during the design process by means of WSDM's modeling concepts. These WSDM's modeling concepts, used in the different phases, are described using the WSDM OWL[23] ontology. To generate code the authors establish a transformation process that takes the conceptual design models as input and generates a set of annotations as a consequence. The transformation process consists of two annotation steps: authoring and mobility, which resemble the original annotation

[23] OWL Web Ontology Language at http://www.w3.org/TR/owl-ref/

process of the Dante approach. The difference is that the authoring annotation in Dante is manual and based on the HTML source code of the Web site. The integration of the Dante [52] annotation process into the Web Site Design Method (WSDM) [13] is graphically illustrated by Figure 2.3 [51].

As we can see in Figure 2.3 the transformation to an accessible design, takes place at the "Execute mapping + Transform pages" step, where a mapping between WSDM and Dante ontologies applies. The WSDM key models where transformation takes place are the WSDM site structure model and the WSDM presentation model, both outputs of the WSDM Implementation Design phase.

that is annotated with concepts from the Dante's WAfA[24] ontology, a relationship between the concepts in the WSDM ontology and the WAfA ontology is established. By using these mapping rules,

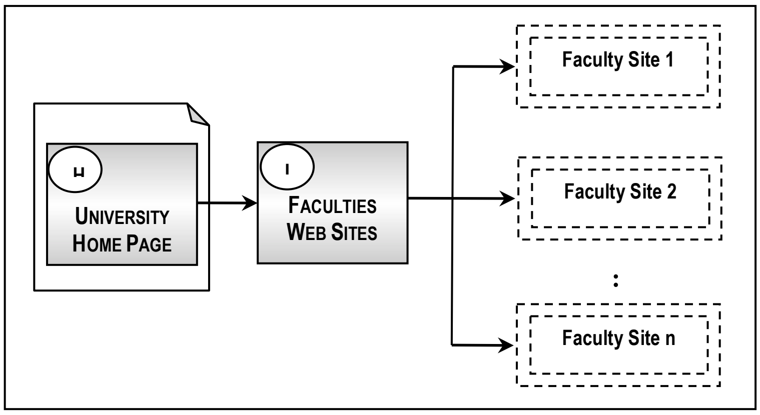


Figure 2.4: Part of the WSDM site structure model for the University home page example

Now, applying this proposal for developing the page example of Section 2.2.1, Figure 2.4 shows part of the WSDM site structure model. As we can see in Figure 2.4, we enrich this model of the University home page with navigational aid links --i.e. the home link and the landmark link components represented by means of the symbols "H" and "L" respectively. From home, the landmark link component offers a list of links that the student may choose when browsing to his/her Faculty Web site.

Figure 2.5 provides the WSDM presentation model as a page template for the University home page example, where the navigational aid links "H" and "L" from

[24] Web Authoring for Accessibility (WAfA) at http://augmented.man.ac.uk/ontologies/wafa.owl

Figure 2.4 are graphically highlighted in grey. Having these WSDM key models, the transformation process consists of two steps: (1) *Authoring Annotation transformation* which uses the information specified in the WSDM models and the Dante's WAfA ontology to generate the authoring annotation and, (2) *Mobility Annotation transformation* which uses the output of the previous transformation as well as the WSDM models to extend the authoring annotation with mobility annotation to improve Accessibility.

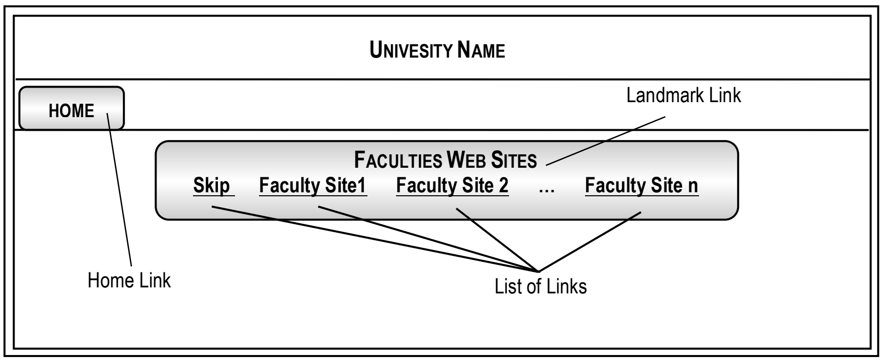


Figure 2.5: The WSDM presentation model for the University home page example

Following, we will explain the transformation process for the University home page example taking into account the WSDM models of Figures 2.4 and 2.5:

(1) *Authoring Annotation transformation.* This process uses the mapping rules between modeling concepts defined in the WSDM ontology and authoring concepts from the WAfA ontology. The "list of text links" at the page example, can be represented by the *List* concept (at WSDM ontology) and by the *NavigationalList* concept (at the WAfA ontology), but this is not a straightforward one-to-one mapping. So, assuming the set **C** as the set of all WSDM modeling concepts and the set **I** as the set of all instances of these modeling concepts, Figure 2.6 shows the corresponding mapping rule for the "list of links" to the Faculties web sites at the University page example. To avoid confusion while applying this rule, the WSDMs concepts are prefixed with "wsdm" and the WAfA concepts with "wafa". The *NavigationalList* WAfA concept is given in bold, followed by its meaning (in italic), an informal explanation of the mapping rule and finally, a formal definition using first-order predicate logic.

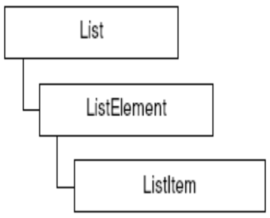
WSDM ONTOLOGY CONCEPT	WAFA ONTOLOGY CONCEPT	MAPPING RULE BETWEEN WSDM AND WAFA ONTOLIGIES
List 	**NavigationalList**	**wafa:NavigationalList**: *A "list of links".* The annotation can be generated for wsdm:List where all list elements have a wsdm:Link defined upon them [35]. $\forall$ i $\in$ I, $\exists$ y $\in$ I: wsdm:List(i) $\wedge$ ($\forall$ x $\in$ I: wsdm:hasChild(i, x) $\wedge$ wsdm:ListItem(x) $\wedge$ wsdm:hasNavigationReference(x, y) $\wedge$ wsdm:NavigationReference(y)) $\rightarrow$ **wafa:NavigationalList(i)**

Figure 2.6: Mapping rule for the "list of links" at the University home page example

(2) *Mobility Annotation transformation.* This process re-uses the mapping rules provided by the Dante approach [52], adjusting them to interact with the WSDM models instead of the HTML code of the Web page. Taking the output of the previous transformation as well as the WSDM models, we extend the *NavigationalList* authoring annotation with mobility annotation to improve Accessibility. Figure 2.7 provides the mapping rule [35] for mobility annotation transformation that applies to objects authoring annotated as a *NavigationalList.* All the links in the list are text links corresponding to the Faculties' names for whose Web sites access are allowed to students. As the mapping rule from Figure 2.7 shows, the *NavigationalList* authoring concept must be annotated with the *DecisionPoint* and *NavigationPoint* mobility concepts, while the *TextLink* authoring concept (required because all the links in the list are text links) must be annotated with *NavigationPoint* and *TravelMemory* mobility concepts. As a consequence, the *NavigacionalList,* where all the links in the list are *TextLink,* must be annotated with *DecisionPoint*, *NavigationPoint* and *TravelMemory* mobility concepts.

MOBILITY ANNOTATION TRANSFORMATION
NavigationalList $\downarrow$ **NavigationalList** $\rightarrow$ DecisionPoint $\wedge$ NavigationPoint **TextLink** $\rightarrow$ NavigationPoint $\wedge$ TravelMemory **NavigacionalList** $\wedge$ **TextLink** $\rightarrow$ DecisionPoint $\wedge$ NavigationPoint $\wedge$ TravelMemory $\downarrow$ **DecisionPoint** $\wedge$ **NavigationPoint** $\wedge$ **TravelMemory**

Figura 2.7: Mapping rule for the *NavigationalList* at the University home page example

A *DecisionPoint* is a choice point where alternative paths of browsing are possible; while a *NavigationPoint* provides a possible route and the user exercises some control by choosing to follow or not to follow it; finally, a *TravelMemory* holds information about where the user has been and provides means to get back there. For the particular case of the University home page example, these mobility concepts will offer a student a point from where it is possible to choose a Faculty name, browse to its Web site and also get back from there to the University home page. We must to keep in mind that authoring and mobility concepts are from WAfA ontology, so the application of the rule for the Pleasers proposal [35], looks like shows Figure 2.8.

$$\boxed{\begin{array}{c}
\forall\ i \in \text{I: wsdm:String}(i)\ \vee \\
(\exists\ x, y \in \text{C:} \\
\text{wsdm:ObjetcChunkReference}(i) \wedge \text{toProperty}(i, x) \wedge \text{rang}(x,y) \wedge \text{wsdm:String}(y)) \\
\rightarrow\ \text{Text}(i) \\[6pt]
\forall\ i \in \text{I: } \textbf{wafa:NavigationalList(i)} \wedge \\
(\forall\ x \in \text{I}, \exists\ y \in \text{I:} \\
\text{wsdm:hasChild}(i,x) \wedge \text{wsdm:ListItem}(x) \wedge \text{wsdm:hasChild}(x,y) \wedge \text{Text}(y)) \\
\rightarrow\ \textbf{wafa:DecisionPoint} \wedge \textbf{wafa:NavigationPoint} \wedge \textbf{wafa:TravelMemory}
\end{array}}$$

Figure 2.8: The Pleaser et al. [35] proposal for the University home page example

The botton-half of the rule is a direct translation of the original rule, applied to the objects annotated as a NavigationalList where all wsdm:ListItems are text elements. The top-half of the rule formally defines a text element [35]. For further details of this proposal, we refer the reader to [35].

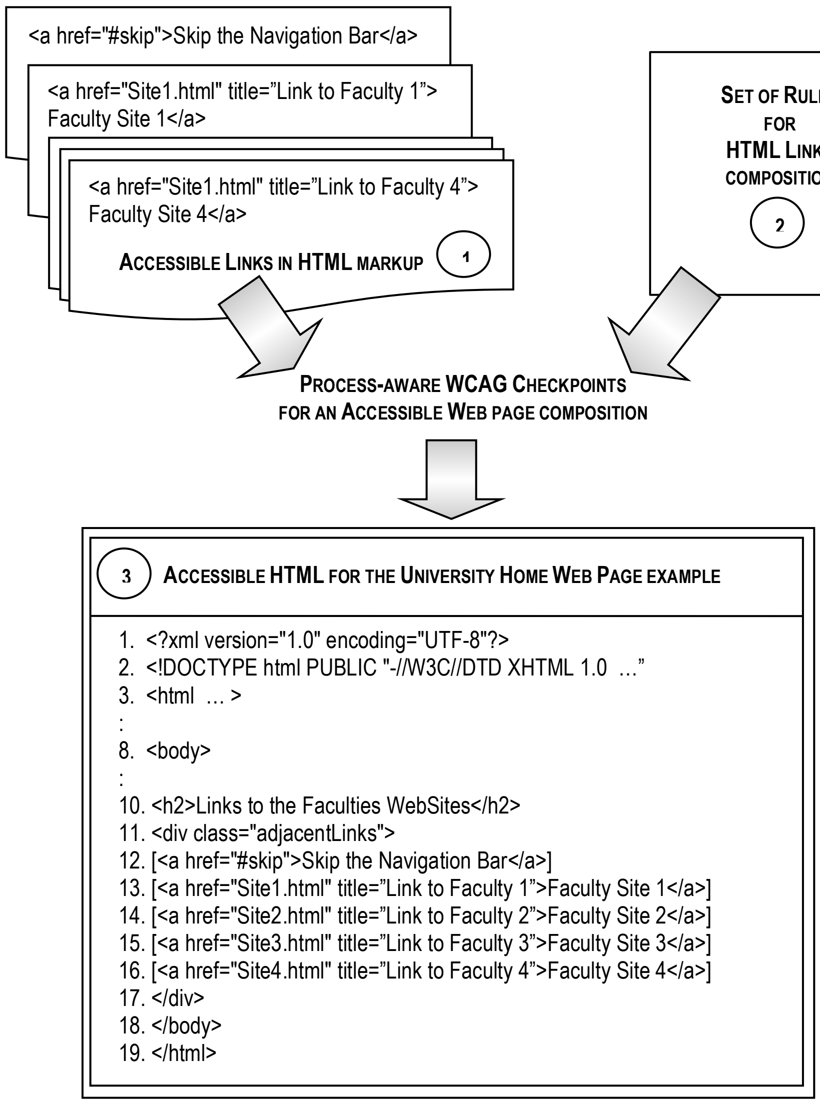
2.2.3 Rules for an Accessible Composition

The work by Centeno et al. [9] presents a set of rules that, in a Web composition process, a design tool must follow in order to create accessible Web pages. These rules are formalized with W3C standards like XPath25 and XQuery26 expressions, defining conditions to be met in order to guarantee that Accessible chunks of Web pages are safely compound into a page that also results Accessible. The authors also propose

[25] W3C XML Path Language at www.w3.org/TR/xpath

[26] W3C XML Query Language at www.w3.org/TR/xquery

using the “Web-Composition Service Linking System” (WSLS) [20] as Accessibility enabled authoring tool that makes this task feasible, and focus on how this tool incorporates Accessibility into the process of generating new Web contents. The XPath and XQuery expressions spot HTML nodes and attributes having Accessibility problems. This work proposes to properly manage these spot elements by an authoring tool, so that the author’s attention can be directly brought to these barriers in a semi-automated edition process.



<a href="#skip">Skip the Navigation Bar</a>
<a href="Site1.html" title="Link to Faculty 1"> Faculty Site 1</a>
<a href="Site1.html" title="Link to Faculty 4"> Faculty Site 4</a>
ACCESSIBLE LINKS IN HTML MARKUP
1
SET OF RULES FOR HTML LINKS COMPOSITION
2
PROCESS-AWARE WCAG CHECKPOINTS FOR AN ACCESSIBLE WEB PAGE COMPOSITION
3
ACCESSIBLE HTML FOR THE UNIVERSITY HOME WEB PAGE EXAMPLE

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  ...”
3.  <html  ... >
:
8.  <body>
:
10. <h2>Links to the Faculties WebSites</h2>
11. <div class="adjacentLinks">
12. [<a href="#skip">Skip the Navigation Bar</a>]
13. [<a href="Site1.html" title="Link to Faculty 1">Faculty Site 1</a>]
14. [<a href="Site2.html" title="Link to Faculty 2">Faculty Site 2</a>]
15. [<a href="Site3.html" title="Link to Faculty 3">Faculty Site 3</a>]
16. [<a href="Site4.html" title="Link to Faculty 4">Faculty Site 4</a>]
17. </div>
18. </body>
19. </html>
```

Figure 2.9: The Centeno et al. [9] proposal for the University home page example

The WSLS approach follows the AOSD separation of concerns principle to decompose complexity and control Accessibility over six distinguished categories: Data, Presentation, Navigation, User, Interaction, Process and Communication. The six elements are mediated by a service control function. Beyond the advantage of the reuse aspect of these components, separation of concerns facilitates also being compliant to the underlying guidelines [9].

Figure 2.9 resumes graphically the proposal at Centeno et al. [9] applied to the page example of Section 2.2.1. As highlighted in Figure 2.9 (1), given $S1 to $S5 compoundable pieces of HTML markup (also called HTML snippets), each one represents an accessible link to a Faculty of the student's University. The composition of these accessible chunks of Web pages, must follow some rules in order to create an accessible "list of links" at the University home page. The proposal provides a set of rules that are focused on formalizing the conditions to be met so that accessible HTML snippets can be safely compound into a page that also results accessible from the WCAG point of view. As shown in Figure 2.9 (2), from the set of rules provided by the proposal, we select for the page example only those rules for HTML links composition. For example, rule 10.5 establishes "provided that all $S1's and $S2's links have non-consecutive links (some printable text between links), their composition could have consecutive links without such printable characters if a $S2's link appears just in front of $S1's link" [9]. This condition for rule 10.5 ("non-consecutive links") is formalized with a combination of XPath and XPointer as depicted in Figure 2.10 Since this formalization is somewhat difficult for those unfamiliar with XPath and XPointer, the next row of Figure 2.10 summarizes its meaning in simpler terms to facilitate its reading; remember that "a" represents an HTML *a* element that is used to define links.

```
$S2//a = () or
(pos($S1,$position)/preceding::a = () or
string-length(normalize-space((end-point(pos($S1,$position)/preceding::a[last()])
/range-to(pos($S1,$position)))/text())) > 0 or
string-length(normalize-space((start-point($S2)/range-to($S2//a[1]))/text())) > 0)
and
((pos($S1,$position)/following::a = () or
string-length(normalize-space((end-point(pos($S1,$position))/range-to(pos($S1,$position)
/following::a[1]))/text())) > 0 or
string-length(normalize-space((end-point($S2//a[last()])/range-to(end-point($S2)))/text())) > 0))
```

$S2**//a = ∅ ∨ (printable characters before first **$S2's link ∧ printable characters after last **$S2's** links)

Figure 2.10: XPath + XPointer pre-conditions for avoiding consecutive links without printable non-linkable characters between them [9]

Meanwhile, rule 13.1 establishes "there should be no links sharing both a text and a title but pointing to different targets; provided $S1 and $S2 have no such ambiguous links there exist a functional dependency such that for every pair of (link's contents, link's title) only a single target may be found in both $S1 and $S2. In that case, we should also make sure that no link in $S1 is similarly described in $S2 (and pointing to a different target), or vice-versa; if so, an ambiguity would be introduced in the composed result" [9]. This condition for rule 13.1 ("clear links") is formalized with XPath as depicted in Figure 2.11.

```
(every $a1 in $S1//a satisfies $S2//a[text() = $a1/text() and @title = $a1/@title and
@href != $a1/@href] = ()) and
(every $a2 in $S2//a satisfies $S1//a[text() = $a2/text() and @title = $a2/@title and
@href != $a2/@href] = ())
```

Figure 2.11: XPath pre-condition for avoiding ambiguous links [9]

Returning to Figure 2.9, given $S1 to $S5 HTML snippets corresponding to Faculty links and rules 10.5 and 13.1, a process-aware WCAG checkpoints takes place for Web page composition to deliver an accessible "list of links" at page example. As we can see in Figure 2.9 (3), the "list of links" conform rules 10.5 and 13.1 responding respectively to the statements "non consecutive links" --i.e. printable characters between links where included, and "clear links" --i.e. title's, target's and content's links are properly specified, to avoid students get confuse while browsing his/her University home page example. For further details of this proposal, refer to [9].

2.2.4 Adaptation to tackle Crosscutting Concerns

Casteleyn et al. [6], focus on how to extend an application with new functionality without having to redesign the entire application. The work states that since creating a Web application has become an increasingly complex task, various design issues like device-dependence, privacy, security, Accessibility, localization, personalization, etc. have become extremely relevant to the application performance. To add new functionality, the authors propose to separate additional design concerns and describe them independently. By using a component-based implementation, they show how to extend a Web application to support additional design concerns at the presentation generation level. Furthermore, they demonstrate how an aspect-oriented approach can support the high-level specification of these (additional) design concerns at a conceptual level.

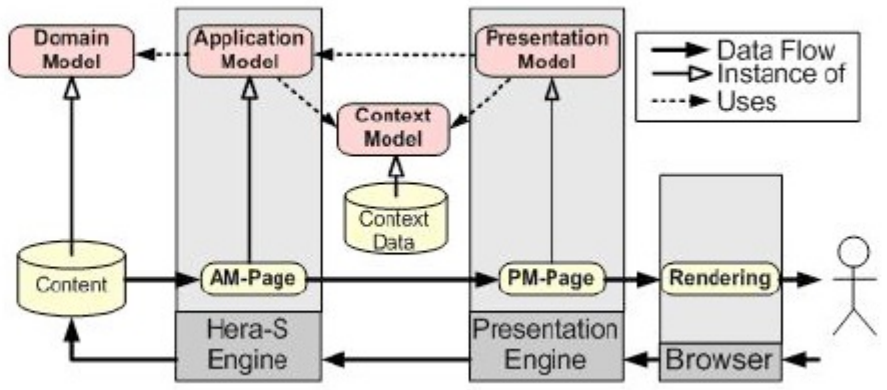


Figure 2.12: Hera-S architecture [8]

The work firstly illustrates how to add adaptation to an existing Hera-based Web application [23], using a component-based implementation. To do so, they apply the Generic Adaptation Components (GAC) approach [16] provided by the AMACONT[27] project. Niederhausen et al. introduce further work over this foundation [32] that proposes an aspect-oriented view on adaptation engineering within the AMACONT framework. By separating the specification of adaptation from the underlying application in the form of so-called adaptation aspects, this work proposes to add new or modify existing adaptation concerns on demand. The authors also present an extension of their graphical authoring tool AMACONTBuilder [15]. This extension allows Web engineers to intuitively incorporate adaptation aspects into Web applications. Casteleyn et al. latest implementation [7] [8] proposes a Semantic-based Aspect-oriented adaptation approach materialized in the form of a domain specific language, which the authors called Semantic-based Aspect-oriented Adaptation Language (SEAL)[28]. It is presented in the context of a Web Information System (WIS) design method, Hera-S, which combines the popular open source Resource Description Framework (RDF)[29]

[27] System Architecture for Multimedia Adaptive WebCONTent at http://www-mmt.inf.tu-dresden.de/Forschung/Projekte/AMACONT/index_en.xhtml

[28] SEAL BNF specification at http://wise.vub.ac.be/downloads/research/seal/SEALBNF.pdf

[29] W3C RDF/XML syntax specification at http://www.w3.org/TR/REC-rdf-syntax/

called Sesame [5] and the rich modeling capabilities of Hera [23], a model-driven approach for engineering Web applications based on semantically structure data. They choose Hera-S because: (i) it naturally builds on Semantic Web data and, (ii) it was conceived with adaptation in mind. An illustrative overview of Hera-S architecture is shown in Figure 2.12 from [8]. Basically, the architecture receives data from the actual source, which conforms to the Domain Model (DM). The Application Model (AM) is instantiated according to the context data provided by the Context Model (CM), resulting in so-called Application Model Pages (AMPs). The authors devised their own custom-made aspect language SEAL to provide adaptation support in the context of Hera-S. By using SEAL's syntax, which is based on BNF notation, they show their adaptation engineering perspective applying pointcuts and advices expressions.

```
:UniversityUnit a ams:NavigationalUnit ;
ams:hasInput [ a ams:Variable ;
ams:varName "U";
ams:varType uncdb:University] ;

ams:hasAttribute [
rdfs:label "UniversityName" ;
ams:hasQuery
"SELECT N1 FROM {$U} rdf:type {uncdb:University};
rdfs:label {N1}"] ;

ams:hasSetRelationship [
rdfs:label "Faculties" ;
ams:refersTo :FacultyUnit ;
ams:hasQuery
"SELECT F FROM {$U} rdf:type {uncdb:University};
uncdb:unversityFaculty {F}"
].

:FacultyUnit a ams:NavigationalUnit ;
ams:hasInput [ a ams:Variable ;
ams:varName "F";
ams:varType uncdb:Faculty] ;

ams:hasAttribute [
rdfs:label "FacultyName" ;
ams:hasQuery
"SELECT FN FROM {$F} rdf:type {uncdb:Faculty};
rdfs:label {FN}"
].
```

Figure 2.13: The Hera-S AM for the University home page example

To demonstrate the practicality of their proposal, they apply and integrate SEAL in the HydraGen engine[30] (an implementation generation tool for Hera-S developed externally by the University of Eindhoven).

Now, applying this proposal for developing our University home page example of Section 2.2.1, a Hera-S Application Model (AM) using Turtle RDF notation[31] would include the statements shown in Figure 2.13.

An Hera-S Application Model (AM) is specified by means of navigational units (denoted by ams:NavigationalUnit and called shorthand: units). A unit can be used to represent a page and it is a primitive that (hierarchically) groups elements (called attributes) that will together be shown to the user. The type of a unit (denoted by ams:varType) refers to a domain data and the specification of this type is done by using the namespace-prefix from the Hera-S Domain Model (DM). Our Hera-S AM example bellow, consists of two units, *UniversityUnit* and *FacultyUnit*, which are of the type uncdb:University and uncdb:Faculty respectively (in this case this namespace-prefix from our Hera-S DM stands for "Universidad Nacional de Córdoba Data Base"). Both units are navigational units of Hera-S AM, each one representing a particularly grouping of information. For example, the *UniversityUnit* contains one attribute (denoted by ams:hasAttribute) representing the university's name and a set of navigational relationships (denoted by ams:hasSetRelationship) from *UniversityUnit* to *FacultyUnit*. Note that the ams:SetRelationship "refersTo" the *FacultyUnit*, which specifies what exactly to show for every faculty. Since a unit will mostly correspond to (a) specific domain concept(s), one or several content elements are needed in order to instantiate the unit. For example, in the *UniversityUnit* the output of the SeRQL queries (denoted by ams:hasQuery) provides a university name and a number of members which will be used respectively to instantiate the UniversityName and the Faculties of the *UniversityUnit*.

Now, by using the domain specific language SEAL it is possible to apply the Casteleyn et al. proposal [8], to provide aspect-oriented adaptation support in the context of Hera-

[30] Hydragen: An implementation of Hera-S at http://wwwis.win.tue.nl/~ksluijs/material/Singh-Master-Thesis-2007.pdf

[31] W3C-Turtle at http://www.w3.org/TeamSubmission/turtle/

S for the University home page example of Section 2.2.1. As Figure 2.14 shows, we have instantiated the adaptation requirement to stand for the Accessibility requirements of adjacent links. The *adaptation a*spect is composed of a pointcut and an advice; while pointcut expressions select exactly those elements from the Application Model (AM) where adaptation concerns need to be applied. Advices specify exactly what needs to be done to the element(s) selected in the pointcut [8]. Back to our example of Section 2.2.1, the pointcut in Figure 2.14 selects sets of relationships --i.e. consecutive links, which originate from a(ny) University unit and target a(ny) Faculty unit. The advice is conditioned to users using a "screen-reader" device. As we explained above, in Hera-S the user's context is captured by the Context Model (CM) and with Hera-S notational conventions, referencing this user's context is done using a "cm:" -prefix.

Adaptation REQUIREMENT: *for users using a screen-reader avoid consecutive links and clearly identify the target of each one of them.*

Adaptation ASPECT:

POINTCUT: type SetRelationship and from uncdb:University and to uncdb:Faculty

ADVICE: if (cm:userDevice.type = "screen-reader") {

ADD attribute containing hasLabel "Faculty Name", hasQuery "SELECT FN FROM {$F} rdf:type {uncdb:Faculty}; rdfs:label {FN}";

ADD rdf:plainLiteral "[" and "]" surrounding;

};

Figure 2.14: Aspect-oriented adaptation using SEAL for Accessibility requirements of the University home page example

Firstly, the advice adds an AM attribute to the relationships selected in the pointcut showing the faculty name with the label "Faculty Name" and the corresponding query, if the user's device is a "screen-reader". Secondly, the advice also uses plain RDF(s)[32] to add square brackets surrounding the relationships selected in the pointcut.

Although, this approach is primarily focused on adapting an existing Web application, we include it because the approach proposes to add relevant design concerns, like

[32] W3C-RDF:PlainLiteral: A data type for RDF Plain Literals at http://www.w3.org/TR/rdf-plain-literal/#Syntax_for_rdf:PlainLiteral_Literals

Accessibility, in an aspect-oriented manner and, it is representative of other similar works in the adaptation field, like [1] [37]. For further details of this proposal, we refer the reader to [6] [7] [8].

2.2.5 User Needs through Personas

By using existing ‘‘best practices of software engineering’’ for Accessibility purposes, the approach by Zimmermann & Vanderheiden [53] presents a methodology for accessible design and testing to capture functional requirements. The approach defines a new way to use proven tools of software engineering, like use cases, scenarios, test cases, guidelines and checkpoints, for Accessibility purposes; and to relate them to each other, thus facilitating automation as much as possible. The resultant methodology or process model for accessible design and testing consist of: (i) capturing Accessibility requirements in a way that makes them tangible and comprehensible, through use cases and the technique of user profiling “personas” [53], (ii) making Accessibility requirements concrete through scenarios and guidelines for accessible design, (iii) manual and automatic testing based on test cases and Accessibility checkpoints that are derived from guidelines, and (iv) complementary user testing and expert reviews, thus evaluating intermediate and end results, and continuously improving the overall process model.

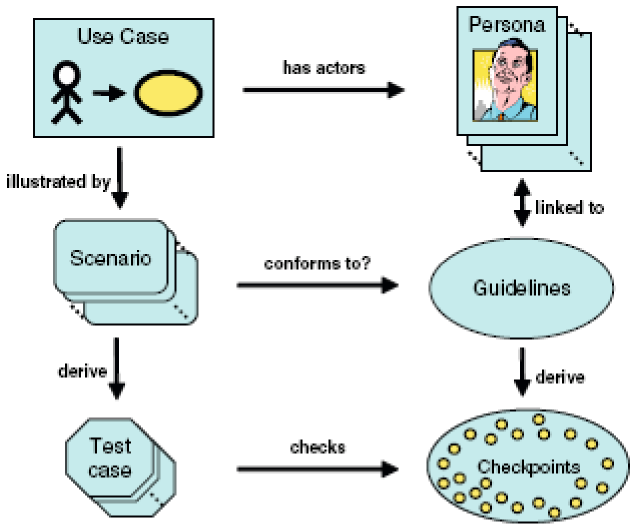


Figure 2.15: Components of the integrated approach and their relationships [53]

In this way for design projects that are employing a use case driven methodology, this approach allows to incorporate accessible design into the existing processes rather than having to add Accessibility as a new process [53]. Figure 2.15 from [53] shows how basic design tools as use cases, scenarios and test cases are linked to personas, guidelines and checkpoints respectively for Accessibility purpose.

Figure 2.16 shows the process model for accessible design and testing by Zimmermann & Vanderheiden [53] applied to our University home page example of Section 2.2.1 and using WCAG 1.0 Accessibility guidelines.

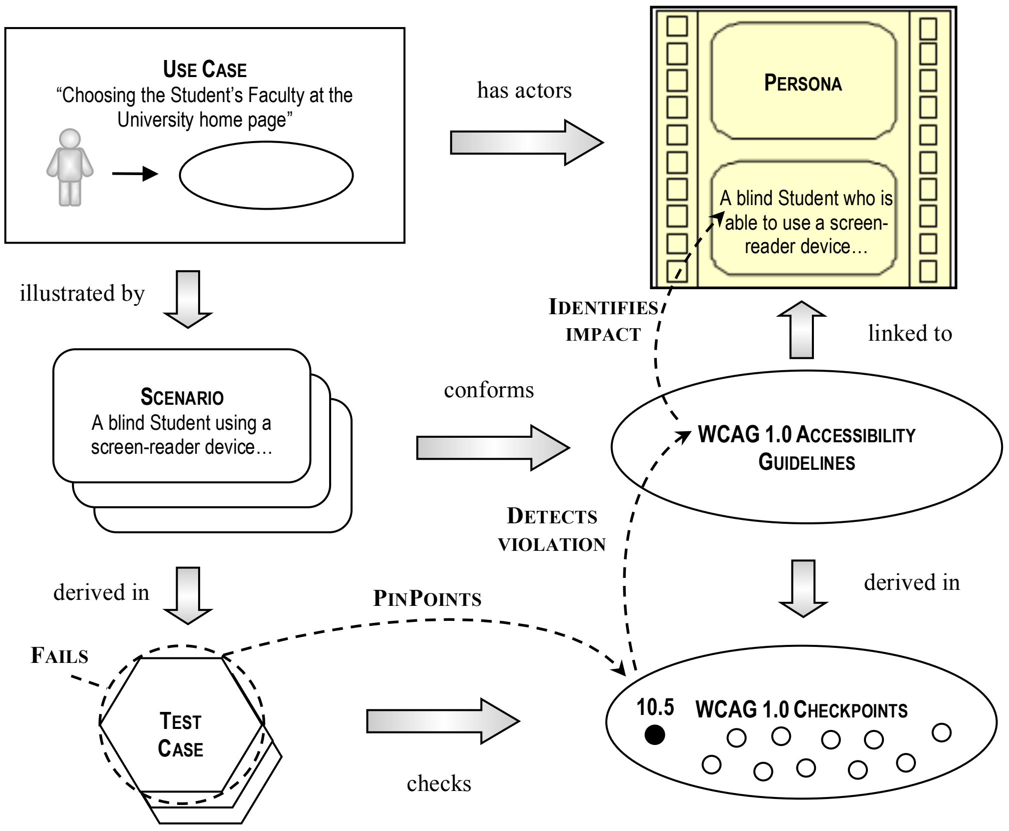

USE CASE
"Choosing the Student's Faculty at the University home page"
has actors
PERSONA
A blind Student who is able to use a screen-reader device…
illustrated by
IDENTIFIES IMPACT
linked to
SCENARIO
A blind Student using a screen-reader device…
conforms
WCAG 1.0 ACCESSIBILITY GUIDELINES
DETECTS VIOLATION
derived in
PINPOINTS
derived in
FAILS
TEST CASE
checks
10.5
WCAG 1.0 CHECKPOINTS

Figure 2.16: The Zimmermann & Vanderheiden [53] proposal for the University home page example

Figure 2.16 shows a situation where a test case is failing because an Accessibility requirement for adjacent links is not met. In this case, the proposed model makes it possible to pinpoint to a particular checkpoint that is causing the failure (10.5 checkpoint), and trace it back to a particular guideline that is violated (guideline 10 from WCAG 1.0). This allows identifying a particular persona (a blind Student) who despite being able to use a screen-reader will not be able to access the application because of the Accessibility barrier identified by the test case failure. The model presented here is not only useful for fixing the Accessibility problems, but also provides a context to the developers for understanding the consequences of failure [53]. For further details of this proposal, we refer the reader to [53].

2.2.6 Model-Driven Development with AWA

Accessibility for Web Applications (AWA) [29] [30] offers a domain specific methodological framework for the development of accessible Web applications. The AWA framework provides: (i) a specific Accessibility process (which can be adopted by other processes), indicating activities, artifacts and their sequence in the different phases of integrating Accessibility criteria, and (ii) the support for modeling and using techniques provided by Web Engineering (WE) methods as well as Model-Driven Development (MDD), the focus of this work.

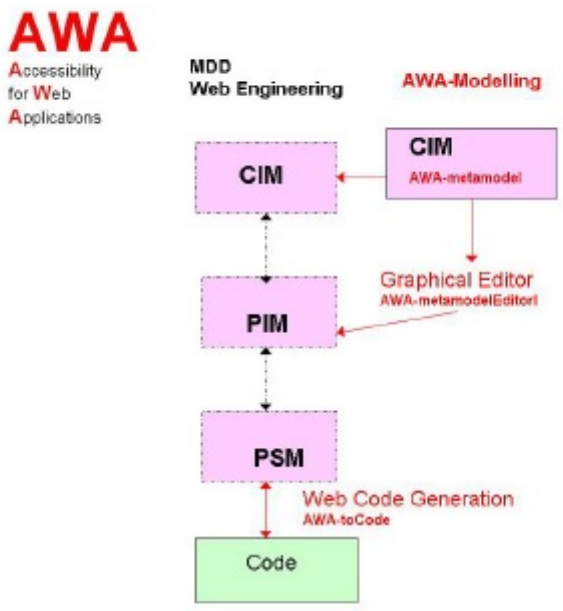

AWA
Accessibility for Web Applications
MDD Web Engineering
AWA-Modelling
CIM
CIM AWA-metamodel
Graphical Editor AWA-metamodelEditor
PIM
PSM
Web Code Generation AWA-toCode
Code

Figure 2.17: AWA for MDA development process [29]

As shown in Figure 2.17, the strategy in AWA consists of providing a Computational Independent Model (CIM), called domain specific AWA-Metamodel, which can be used to build Platform Independent Models (PIMs) and Platform Specific Models (PSMs) for accessible applications within WE methods. The authors provide an AWA-toCode resource and the strategy is based on a transformation Model-to-Text (M2T) to generate code from PSMs. In this work, they also announced that they have developed a CASE support for metamodeling, using the Ecore plugin from the Eclipse Modeling Framework (EMF)[33] [29].

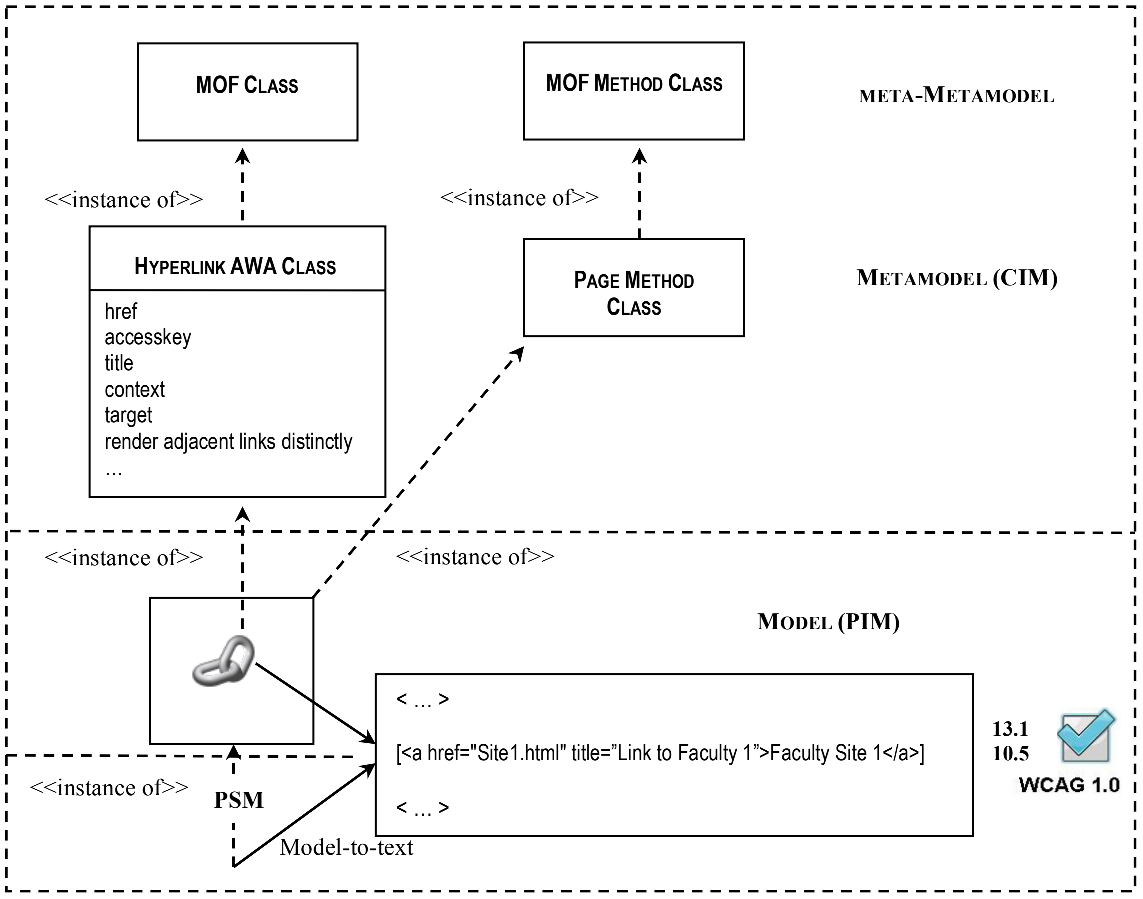


Figure 2.18: The Moreno et al. [29] proposal for the University home page

Figure 2.18 shows AWA for Model Driven Architecture (MDA)[34] applied to the Hyperlink concept required by our University home page example of Section 2.2.1. Here several constructors have been defined in the MetaObject Facility (MOF)[35] to support the abstraction of Web Accessibility concepts. The diagram develops the concept of hyperlink that includes required attributes to enable the hyperlink to meet the

[33] EMF overview at http://help.eclipse.org/indigo/index.jsp?topic=/org.eclipse.emf.doc/references

[34] OMG-MDA overview at http://www.omg.org/mda/

[35] OMG-MOF specification at http://www.omg.org/mof/

WCAG standard, such as the title attribute. This attribute contributes to satisfy the 13.1 checkpoint of WCAG 1.0 that establishes “Clearly identify the target of each link”. To continue with the example of Section 2.2.1, Moreno et al. [29] do not consider the 10.5 checkpoint of WCAG 1.0 as a property for the link/hyperlink concept. Although, notice that as we have done at the hyperlink AWA class in Figure 2.10, it is possible to include the “render adjacent links distinctly” attribute, to enable meeting this Accessibility requirement, if the presence of adjacent links makes it necessary.

A graphic element representing a hyperlink (MOF meta-object) has been defined in the AWA-Editor, and may be included in the PIM models, which contain knowledge provided by the AWA-Metamodel necessary for the Web code generation in the final phase [29]. For further details of this proposal, we refer the reader to [29] [30].

In this Chapter we presented Accessibility in the context of some WE approaches. We reviewed and applied in a case study five different proposals [35] [9] [6] [53] [30] that consider this quality factor in the development process of Web applications.

After introducing background (Chapter 3) and our proposal (Chapter 4), we will apply it (Chapter 5) and we will come back to the approaches summarized here to compare them to our proposal (Chapter 6).