MAIS
Multichannel Adaptive Information Systems

Fondo investimenti per la Ricerca di Base (FIRB)
Programma Strategico Tecnologie abilitanti per la Società della Conoscenza - ICT

R. 7.1.4. – Rapporto Conclusivo

Florian Daniel, Maristella Matera

Abstract

This document summarizes the main results achieved within the task 7.1. In particular, it illustrates a methodological final framework defined for the development and the deployment of context-aware Web applications and services, and also summarizes the main features of the developed tools.

Data	Novembre 2005
Tipo prodotto	
Stato	Bozza
Unità responsabile	Polimi
Unità coinvolte	PoliMI, Cefriel, Engineering
Autore da contattare	Maristella Matera
Versione	0.1

Table of Contents

1. Introduction

More and more users are asking for services and content that are highly tailored to their devices and, more generally, to their specific contexts of interaction. Accordingly, the development of suitable information systems presents new requirements to application designers. In order to help designers cope with such demands, in this document we provide some guidelines to support them in the development of multichannel, adaptive Web information systems. As shown in Figure 1, these guidelines cover several aspects of design and development, which have been faced and solved during the project, and have been combined to form a methodological framework.

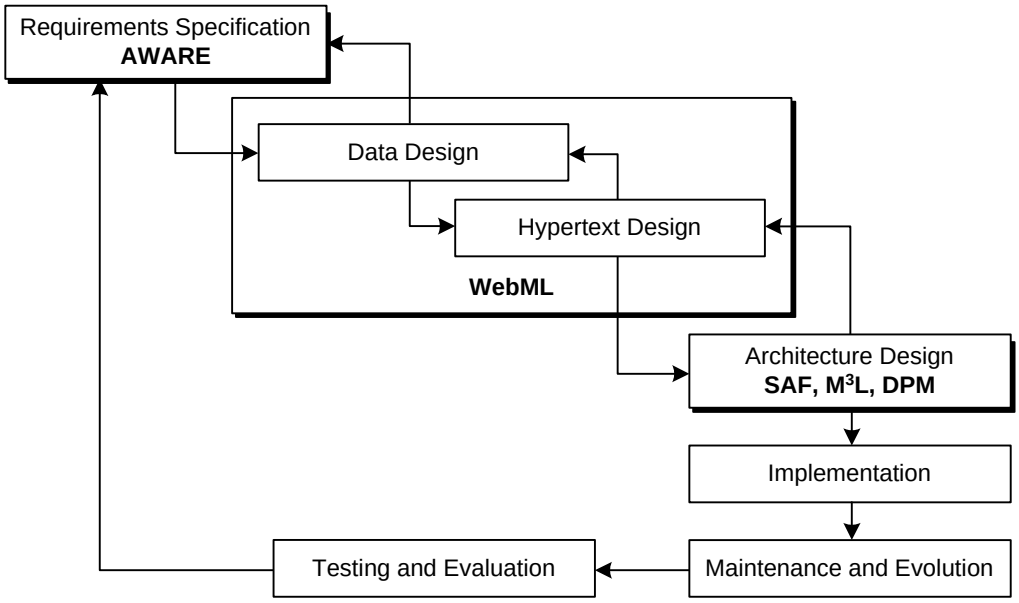

Figure 1. Software life cycle, with special focus on adaptive Web applications

The framework includes a number of coordinated activities as follows:

- Requirements for Web applications are formalized by means of the AWARE method (Section 2), and serve as input to the subsequent design tasks.
- Data and hypertext design is based on the WebML method, extended appropriately to adaptive applications (Section 3).
- Sections 4 and 5, finally, put the methodological results on a technological basis, by outlining the multichannel and multimodal deployment architectures and implementations developed, namely SAF, M3L, and DPM.

2. Modeling the Requirements of Multichannel Applications

AWARE (Analysis of Web Application REquirements) is a requirements engineering model which recognizes the central role of all the relevant stakeholders in a project and their goals in eliciting, analyzing, and specifying requirements for an interactive application, as in traditional goal-based requirements engineering approaches [Antón96, DFL91, MCY99]. Goal-oriented requirements engineering assumes that the "why" of the stakeholders' requirements has to be sought and documented, in order to highlight and keep track of the reasons behind the requirements and the design decisions. Lack of space prevents us from fully illustrating the methodology (for which we refer to [BP04]). Here we recall the key concepts of AWARE, inviting the reader to grasp the essence of the method.

2.1. Understanding Stakeholders, Users and Their Goals

Stakeholders are those people who have an interest in the success of an application or may have knowledge relevant to it and visions related to its success. The stakeholders include, of course, the clients who fund the development of the application, but may also include other company representatives, marketing managers, and sponsors, as well as decision makers, opinion makers, or domain and content experts external to the organization. Some of these stake-holders have personalized goals with respect to the application to be built (see the examples in Figure 2), in the sense that they have a direct interest in its successful deployment and use (e.g., the client or her/his representatives). Others may not have goals but can project their visions on the application: thanks to their knowledge or expertise in the field, they can share their perspective and opinion on the project (e.g about the content, the technology, the communication strategy, the users, their needs, and so on). The users are an important category of stakeholders. They can be described in terms of the personal characteristics of archetypal visitors (also called "person"). Personal characteristics are chosen along any dimension that analysts consider relevant to the design. For an application related to tourism in a national park, the relevant persons may be first-time users, experienced users, children, parents, people between 15 and 18 years old, people over 30 years old, people with fast connections, people with slow connections, people who are not familiar with the technology, people with visual disabilities, hearing-impaired individuals, students, foreign tourists, first-time-visitor tourists, etc. Persons may be defined along any orthogonal dimension (e.g., site knowledge, family relationship, level of disability, age, domain expertise, or occupation) that contains user characteristics. A user profile aggregates a meaningful set of multidimensional characteristics, which tentatively describe a potential visitor to the application.

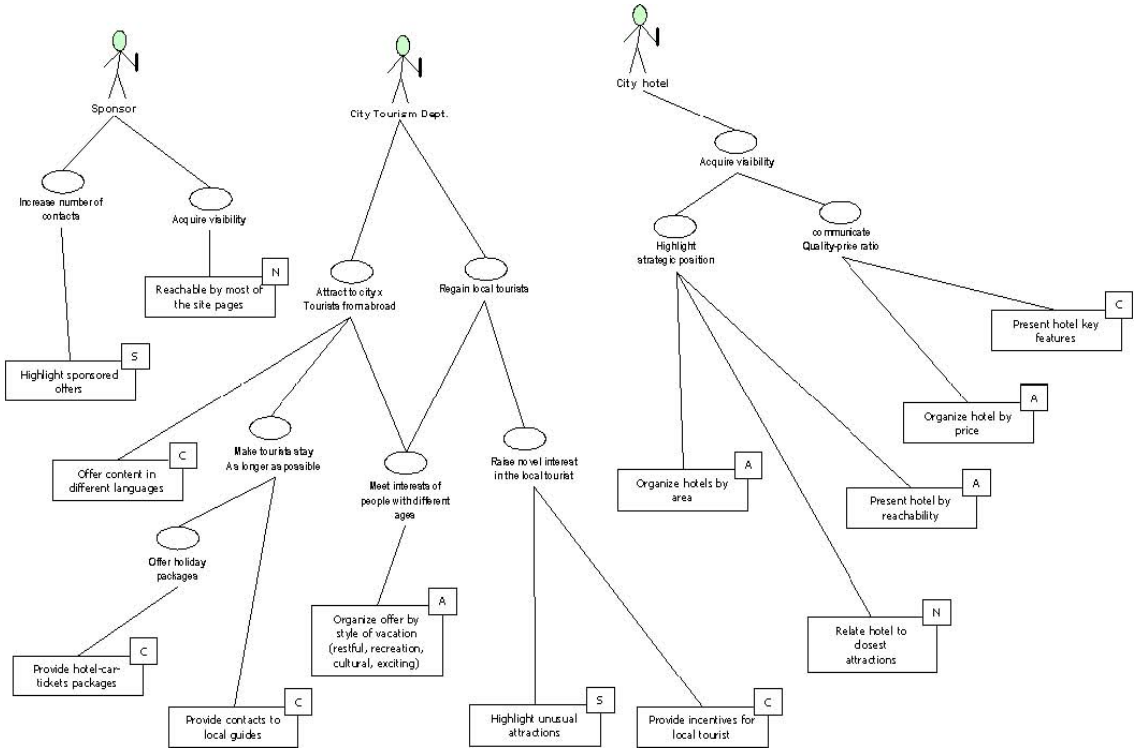

Figure 2. Example of the main goals of stakeholders

The requirements analysis should reason carefully about the user's goals, which should be plausible motivations for visiting the application, or the objectives of their interaction. The user's goals may vary in granularity from low-level, specific information seeking ("find the opening hours of the park on day X"), called functional goals, to higher-level, open-ended, ill-defined needs or expectations ("decide whether the city is worth visiting"), called soft goals [BM03]. Goal identification should also allow one to define the overall purpose of combining different communication channels, the definition and selection of the types of channel (mobile phone, PDA, interactive TV, kiosk, Website, etc.), the role played by each channel in the communication strategy, and the specific goals envisioned for each channel.

2.2. Analyzing Goals and Using Scenarios

AWARE adopts a refinement process to pass from all the stakeholders' (including the users') high-level goals to subgoals and, eventually, to application requirements. The raw material gathered during elicitation may consist of an unstructured mix of very high-level goals, pieces of design, examples of other sites, design ideas and sketches, design decisions, and detailed requirements. This first set of raw material must somehow be organized in order to be usable by analysts, and fed into design. The analysis of such material may be guided by the following lines of inquiry:

- What does a given high-level goal mean? How can it be clarified? Often, in fact, the goals of the users and of the main stakeholders are too vague, abstract, or generic, posing obstacles to

devising operational indications for the designers. For example, if a goal of a main stakeholder is to “attract new tourists”, analysts should inquire “What does it mean specifically?” To make tourists stay longer in the territory? To bring new people to the territory? From which countries?

- What are the possible (realistic) ways to satisfy a high-level goal? Analysts should elicit possible subgoals which may contribute to accomplish the long-term goal. For example, for the goal “How can we convince people to stay longer in the surroundings of the city?”, a possible subgoal might be to “highlight a variety of tours and sequences of visits to attractions lasting one week or more” or “explore tourism retention strategies useful for achieving the high-level goal”.

Refinement involves a decision-making process which is crucial for the definition of the communication strategy that will be implemented in the application. This is the activity in which analysts make the most important strategic decisions about the application. The refinement process also applies to user goals. Goal refinement for user goals should ask: “How might a user with this profile want to accomplish her/his goal?” For example, the user goal “planning a visit to city X” may be decomposed into a number of subgoals, such as “know what are the must sees of the city”, “decide on a suitable hotel where to stay”, or “see interesting hotspots near the hotel”.

To facilitate the elicitation and refinement process, user scenarios may complement goal analysis. Scenarios are commonly recognized as powerful drivers for goal-based approaches. Scenarios may take the form of narrative descriptions (also defined as “stories about use”) of circumstances in which the application is used by a user with a specific profile. The essential ingredients of a scenario are a user profile and a user goal. When a plausible story which combines such a profile and such a goal is constructed, a scenario emerges which describes a success story of use of the application. Scenarios can assist analysts in discovering new requirements, exemplifying goals, revealing new goals, and facilitating the communication of the requirements to the stakeholders.

2.3. Defining and Organizing Requirements

For each channel, which includes not only the type of device but also the characteristics of the context of use (as defined in MAIS), the requirements for content-intensive interactive applications are expressed in natural language and their level of detail is negotiated between analysts and the design team. The requirements are not aimed at capturing all the functionality of the application, but only at those crucial features needed by designers to shape the user experience and by stakeholders to agree on initial specifications. To organize the requirements set and to facilitate the subsequent design activity, AWARE classifies the requirements according to the aspects of the design for which they have an implication. The AWARE requirement taxonomy includes (among other things) the following

dimensions [BP04]: Content, Structure of Content, Access Paths, Navigation, Presentation, Operation, Accessibility and Adaptation.

The requirements give coarse-grained, semi-structured indications to designers. We propose a model, named IDM (Interactive Dialogue Model), as an innovative conceptual tool to facilitate the transition between requirements and detailed application design. After a first (incomplete and provisional) set of requirements, designers may need to give a coherent shape to the user experience in terms of possible dialogues that the user may be engaged in, to properly support the scenarios envisioned during the requirements analysis. In this dialogue-based perspective, the in-the-large structure of the application takes the shape of a dialogue generator, and the user may activate one or more dialogues within a limited range of possibilities.

IDM enables us to communicate, document, and take decisions about the following concerns: What is the overall content? What is the overall organization of the content? How can the user access the content and browse through the various pieces? What are the operations/transactions available to the user?

Through a few simple, intuitive primitives (based on theories about dialogue, and linguistic theories), IDM enables designers to define the overall patterns of the communication and interaction dialogue before digging into details that depend on technical issues. IDM also embeds methodological support for anticipating the description of context-aware services at requirements/design time, such as the impact of the user's location on the dynamics of the dialogue, and the provision of high-level rules describing the behaviour of the application in response to changes in the user context. As described below, IDM schemas can also be easily mapped onto lower-level design languages such as WebML.

IDM	WebML - Data	WebML - Hypertext
Topic	Entity	/
Dialogue Schema	/	Site View
Dialogue Act	Attributes/Entity/Relations	<Cluster of> Pages/Areas
Introductory Act	Access Schema	Content Unit/Index Chain
Group Strategy	/	Navigation Pattern
Transition Act	Interconnection Schema	Navigation Pattern
Transition Strategy	/	Navigation Pattern
Operation Act	/	Content Management Pattern
Structural Strategy	/	Navigation Pattern

Figure 3. Mapping of IDM onto WebML primitives. The primitives. The "/" symbol indicates the lack of corresponding primitives in one of the two WebML models

3. Design of Multichannel Adaptive Web-Based Applications

The design of the front-end ofthe application leverages a conceptual-modeling approach based on the adoption of WebML (Web Modeling Language) [CFBCCM02]. WebML is a visual language for specifying the content structure of a Web application and the organization and presentation of contents in one or more hypertexts. The design process starts with the specification of a data schema, expressing the organization ofthe content of the Web application. The WebML data model uses Entity–Relationship primitives. The WebML hypertext model then allows one to describe how content, previously specified in the data schema, is published in the application hypertext. The overall structure of the hypertext is defined in terms of site views, areas, pages, and content units. A site view is a hypertext, designed to address a specific set of requirements. Several site views can be defined on top of the same data schema, for serving the needs of different user communities, or for arranging the composition of pages to meet the requirements of different access devices such as PDAs, smart phones, and similar appliances. A site view is composed of areas, which are the main sections of the hypertext and comprise, recursively, other subareas or pages. Pages are the actual containers of information delivered to the user; they are made of content units, which are the elementary pieces of information extracted from the data sources by means of queries, and published within pages.

Content units and pages are interconnected by links to constitute site views. Links can connect units in a variety of configurations, yielding to composite navigation patterns. Besides representing user navigation, links between units also specify the transportation of some information that the destination unit uses for selecting the data instances to be displayed.

Some WebML units also support the specification of content management operations. They allow the creating, deleting or modifying of an instance of an entity (through the create, delete and modify units respectively), or the adding or dropping of a relationship between two instances (through the connect and disconnect units, respectively). Recently, WebML has also been extended to model invocations of Web services; in this context, application data can be derived from external data sources as well [BCFAB05]. For a more complete presentation of WebML and its visual notation, the reader is referred to [CFBCCM02].

3.1. WebML and Context-Awareness

The overall design process for context-aware applications can follow the activity flow typically used for conventional Web applications. However, some new issues must be considered in modeling and exploiting the context, in order to achieve adaptive behaviour.

During data design, the user and context requirements can be translated into three different subschemas complementing the application data (see Figure 4):

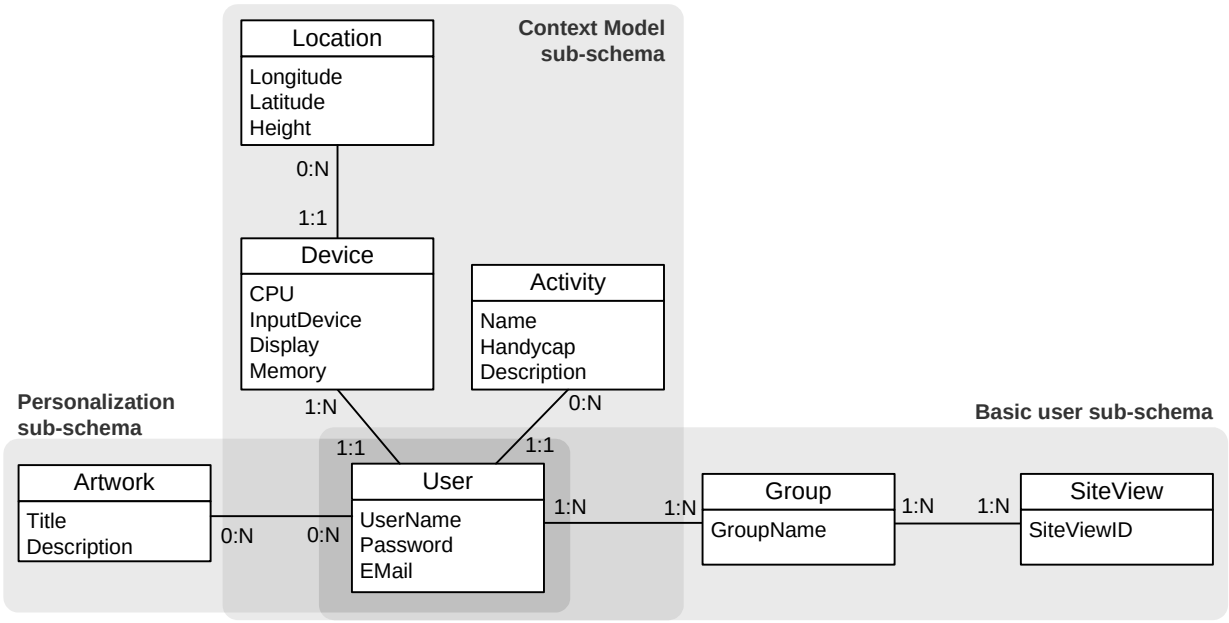

Figure 4. Three subschemas representing context data

The user sub-schema, which clusters data about users and their access rights to application data. In particular, the entity user provides a basic profile of the application's users, the entity Group allows access rights for group of users to be managed, and the entity SiteView allows users (and user groups)to be associated with views over the application's data source. In the case of adaptive context-aware applications, users may require different interaction and navigation structures, according to the varying properties of the context.

- The personalization subschema, which consists of entities from the application data, associated with the User by means of relationships expressing user preferences for some entity instances. In general, relationships defined between the entity User and any other entity of the application data support the personalization of the content of the entity with respect to the identity of the user. For example, the relationship between the entities Artwork and User in Fig. 1.4 allows the selection and the presentation to the user of the artworks s/he likes most.
- The context sub-schema, which includes entities such as Device, Location and Activity, that describe particular properties of the context that are considered by the application in order to provide adaptivity. Context entities are connected to the entity User to associate each user with her/his (personal) context.

Such a context representation is consistent with the MAIS context model and slightly extends it to take into account some requirements specific to WebML (e.g., site view specification).

During hypertext design, adaptive functional requirements are considered to augment the application's front end with reactive mechanisms. As illustrated in Figure 5, our basic assumption is that context-awareness is a property to be associated only with some pages of an application, and not necessarily the application as a whole. Location-aware applications, for example, adapt "core" contents to the position of a user, but typical "access pages" (including links to the main application areas) might not be affected by the context of use.

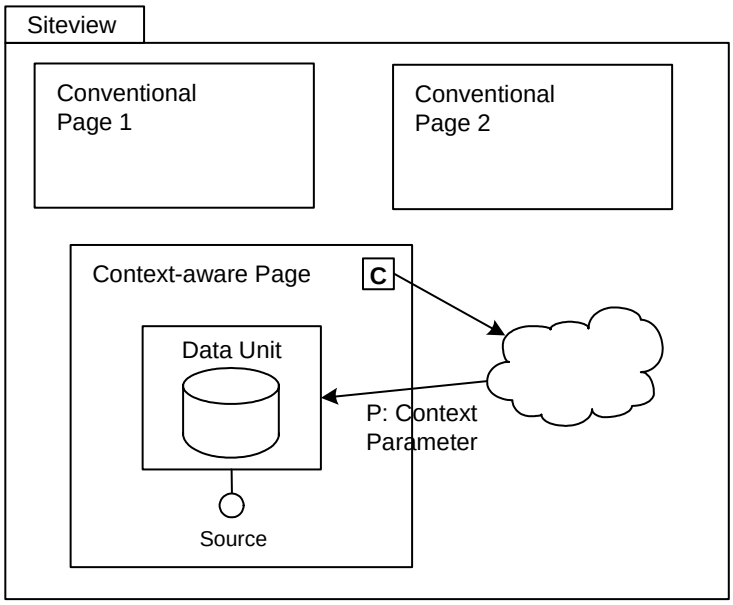

Figure 5. Coarse hypertext schema highlighting context-aware pages. Context-aware pages are labeled with a C and are associated with a context cloud

We therefore tag adaptive pages with a C-label (standing for "Context-aware")to distinguish them from conventional pages. This label indicates that some adaptivity actions must be associated with the page. During application execution, such actions must be evaluated prior to the computation of the page, since they can serve to customize the page content or to modify the predefined navigation flow. As shown in Figure 5, adaptivity actions are clustered within a context cloud. The cloud is external to the page, and the adaptivity actions that it clusters are kept separate from the page specification. The aim is to highlight the two different logics derived from the roles played by pages and context clouds: while the former act as providers of content and services, the latter act as modifiers of such content and services.

In order to continuously evaluate the state of the context and the executing page's adaptivity actions, the C-pages must be provided with autonomous intervention capabilities. In the absence of dedicated

push mechanisms[1], such capabilities can be achieved by periodically refreshing the viewed page and giving the adaptive logic of the application the possibility to intervene in the application itself before rendering the actual response. Where no push mechanisms are available, this polling mechanism provides a valuable "simulation" of the required active behavior. Such mechanism could also be managed by a middleware architecture, such as the MAIS reflective architecture.

3.2. Specifying Adaptivity Actions

The design of context clouds assumes a central role for context-aware applications. Context clouds are associated with the page by means of a directed arrow, i.e., a link, exiting the C-label. This link ensures communication between the page logic and the cloud logic, since it can transport parameters derived from the content of the page, which may be useful for computing the actions specified within the cloud. Vice versa, a link from the cloud to the page can transport parameters or, in general, values computed by the adaptivity actions, which might affect the adaptivity of page contents with respect to a new context.

In order to support the specification of adaptivity actions in the context cloud, WebML has been extended through some new visual constructs that refer to a number of dimensions:

1 **Acquisition and management of context data.** These actions may consist of:
   - Acquisition of fresh context data, provided by means of device-or client-side-generated URL parameters. A new GetURLParameterunit has been introduced to support the retrieval of parameters generated at the client side and communicated to the application by appending "parameter–value" pairs to an HTTP request's query string. Once fresh context parameters have been retrieved, the values previously stored in the data source are replaced accordingly. This operation is specified by means of WebML content management units.
   - Acquisition of context data from the context model. The execution of adaptivity actions may require the retrieval of context data already stored in the application data source, without requiring any visualization. A GetDataunit has therefore been introduced. Similarly to WebML content units, it specifies the retrieval of values from the data source, according to a selector condition. Differently from content units, it does not publish the retrieved values in a page.
2 **Condition evaluation.** The execution of some actions may depend on the evaluation of some conditions. The pattern that recurs most often consists of evaluating whether the context has

[1] The standard HTTP protocol underlying most of today's Web applications implements a strict pull paradigm.

changed, and hence triggering some adaptivity actions. The evaluation of conditions is specified through two control structures, represented by the If and Switch operation units that have been recently proposed for extending WebML for workflow modeling [BCCFM03].

3 **Page content adaptivity.** Parameters produced by context data acquisition actions and by condition evaluation can be used for page computation. They are sent back to the page by means of a link exiting the context cloud and going to the page. The result is the display of a page where the content is filtered with respect to the current context.

4 **Navigation adaptivity.** The effect of condition evaluation within the context cloud can be an automatic, i.e., context-triggered navigation causing redirection to a different page. The specification of context-triggered navigation just requires a link exiting the context cloud to be connected to pages other than the cloud's source page.

5 **Adaptivity of the whole hypertext structure.** In order to deal with coarse-grained adaptivity requirements, for example due to the user changing his/her device, role, and/or activity within a multichannel, mobile environment, a switch to a different site view might be needed. Therefore, a ChangeSiteView unit has been introduced, which takes the identifiers of a target site view and of a target page as input. In order to support "contextual" switching, the input link also transports parameters characterizing the current state of interaction, such as those representing selections made by the user, session data (e.g., the object identifiers of the user and group), and parameters characterizing the current context, retrieved through the data acquisition cycle performed most recently.

6 Adaptivity of presentation properties. In order to support more fine-grained adjustments to the application's appearance, a ChangeStyleunit has been introduced, for representing run-time modification of the presentation properties of the style sheet coding.

3.3. An Example of Context-Aware Design with WebML

In the MAIS project, we have experimented both with the methodology and with the extension of the model by means of a prototype application providing context-aware tourist information [M04]. Figure 1.6 shows a fragment of the application design, illustrating some of the extensions presented above[2]. Starting from the Sight Details page, the schema states that, on the first automatically triggered access

[2] For more details about the visual notation of the WebML extensions for context-awareness, the reader is referred to [CDM03].

to the page, the context cloud is accessed and the operations included in it are performed [3]. Hence, the user's Latitude and Longitude are retrieved from the request parameters by two GetUrlParameterunits. The retrieved values are used by the GetDataunit Get Sight to identify a suitable Sight for the current user's position. Then, the object identifier (OID) retrieved by the GetDataunit is checked by the If unit. If the OID value obtained is not null, the corresponding Sight is shown in the Sight Details page (content adaptation); otherwise, the previously retrieved Latitude and Longitude are used to get a city map from an external Web service. The invocation of an appropriate service could be managed by the MAIS platform through the Concrete Service Invoker. The user is then redirected to the City Map page, and the perceived effect is that of an automatically performed navigation action. Figure 1.6 also models the City Map page as a context-aware page sharing its context cloud with the Sight Details page. Therefore, as soon as an automatic refresh of the CityMap page occurs, the shared context cloud is triggered again and the application is adapted to the user's new position.

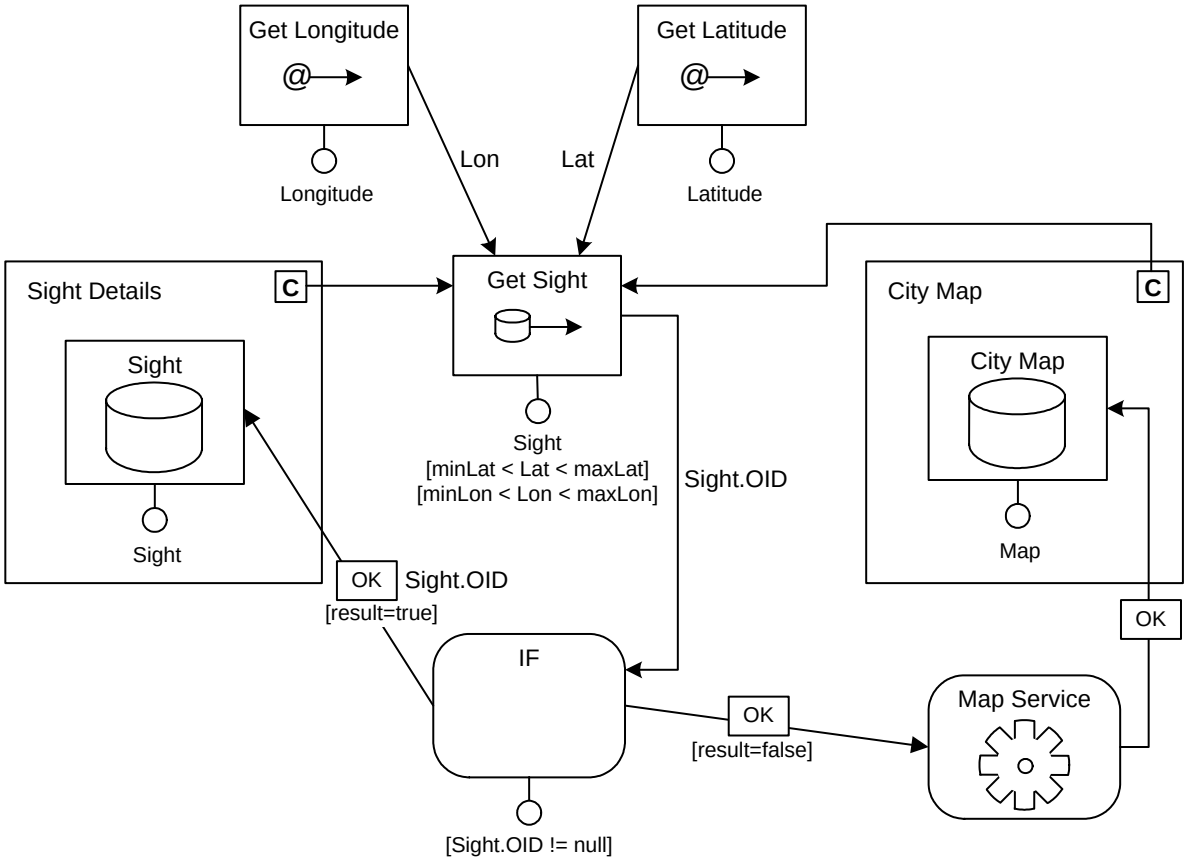

Figure 6. A schema exemplifying some of the WebML extensions introduced for context-awareness retrieved by the Get Data unit is checked by the If unit.

[3] For the sake of simplicity, the cloud is not explicitly represented here; however, it consists of five operation units positioned outside the two pages.

4. Multimodal Deployment of Adaptive Applications

To support the deployment, execution, and delivery of adaptive, multimodal applications, designed using the methodology proposed in Section 3, two frameworks have been developed: the first is focused on adaptive, context-aware applications, and the second deals with multimodal delivery of hypertexts. The two frameworks will be described below.

4.1. Context-Awareness

A framework called SAF (Situation Aware Framework) has been developed to allow the simple design, delivery, and execution of context-aware Web applications. The kinds of adaptation supported by SAF are:

- layout: adaptation of the disposition of objects in the page space;
- presentation: adaptation of color scheme, font type, and font size;
- entity instance selection: selection of a specific instance of an entity;
- attribute selection: selection of the attributes of an entity to be shown.

In SAF, situation awareness is managed through a declarative approach. An SAF situation–aware service can be modeled with WebML, using properties that express declaratively the adaptation behavior that the platform must perform. In SAF, the adaptation behavior of the platform can be determined using two approaches:

- Explicit: the service designer must explicitly declare the kind of adaptation desired using properties or rules, as described above.
- Implicit: this kind of adaptation is performed automatically by the framework according to a set of predefined rules. If the user is running, for example, the layout can be switched to one column mode and the font size can be set to the largest size available. The rules managing this process are defined in the framework.

Figure 7 shows a brief description of the architecture of the framework and of its components. The left-hand side of the figure describes the design-time components:

- The context manager must design the context model, and define the set of properties (and values) used to describe the environment.
- The service designer creates the service description using the extended WebML model and explicitly defines the adaptation behavior of the system.

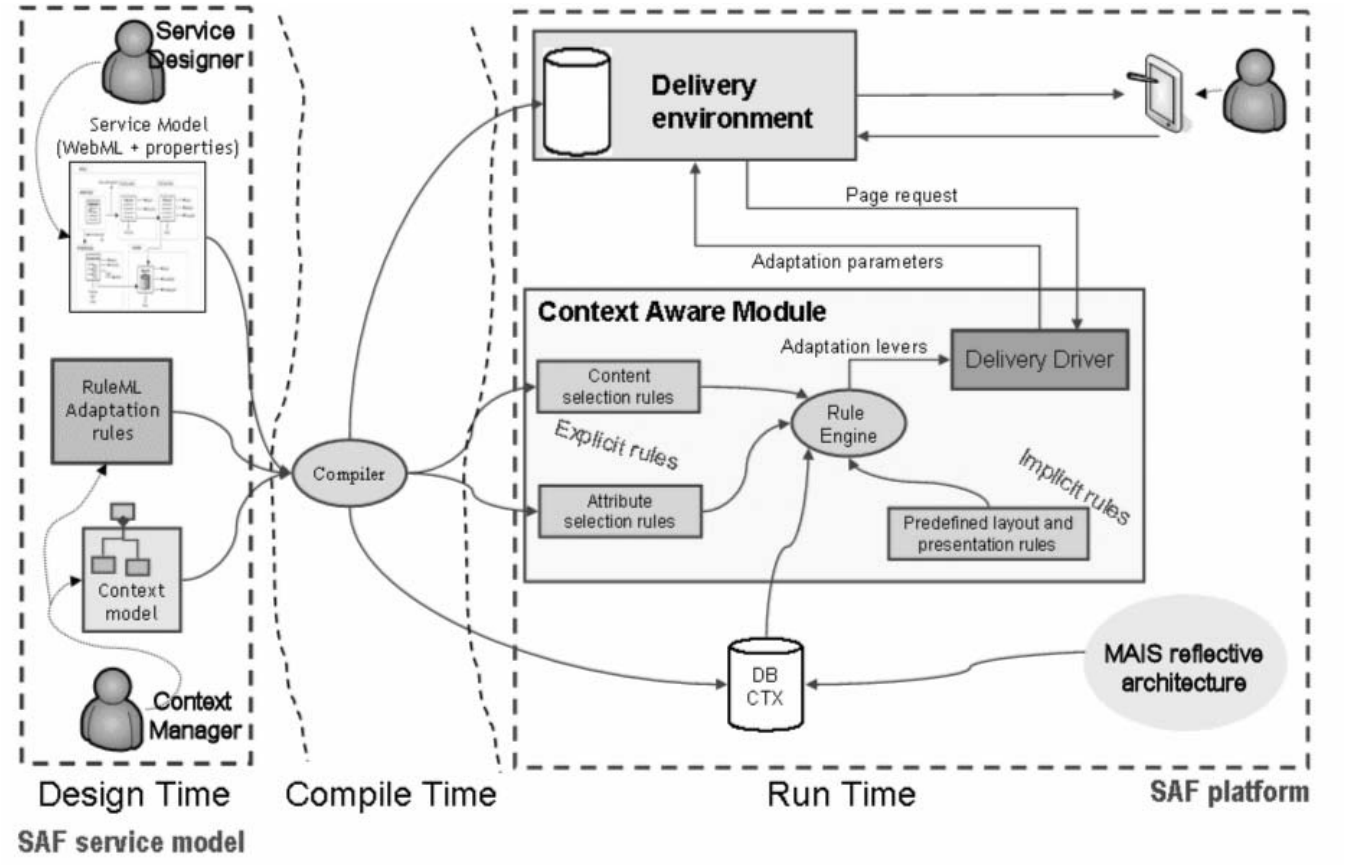

Figure 7. SAF Architecture

The right-hand side of the figure shows the run-time components. The context-aware module shown is responsible for taking the decisions about the adaptation actions to be performed. According to rules that have been defined, the status of the context, and the events that have occurred, the rule engine decides which adaptation actions to perform and uses a driver to set up the delivery environment configuration.

4.2. The M3L Multimodal Framework

The M3L framework is designed to deliver multimodal content using, both for input and for output, vocal and visual interaction modes. These modes has been chosen considering the capabilities and characteristics of the devices currently available on the market. The underlying design model of the M3L multimodal framework is based on an extension of the WebML language. To make a multimodal-enabled WebML service, additional information is needed. For every component of the page, it is necessary to describe the interaction modes to be used, both for input and for output. A WebML design is enriched with properties associated with the various WebML units. These properties are translated into M3L using specific attributes.

The M3Lmultimodal framework is able to manage different devices at the same time synchronizing them to offer a coherent view of the same service on the various channels supported. The user feels that he/she is interacting with a single integrated service even if information is transmitted and delivered through several physically different channels. The proposed solution allows services to be written no more than once, through the use of a multimodal specific markup language (which gives us the name M3L).

M3L is defined as a set of XHTML modules: the "multimodal"(used to structure the M3L document) and "M3L Forms"(specific to user input) modules. The two modules, together with the XHTML framework, constitute the M3L language. M3L conforms to the XHTML Host language specification.

The two modules define new elements and attributes used to manage input (especially for data collected by form) and output synchronization. Attributes are defined to allow the developer to choose the best interaction mode for output data. These attributes allow service developers to select a preferred (or compulsory) delivery or input mode. The out attribute allows one to specify which modes can be used to deliver the content of an element to the user. This attribute is made available to any tag that contains information that must be presented to the user. The mode attribute, on the other hand, specifies the modes that a user can use to input data. It is associated with form fields and may have three possible values: "text" to indicate that the user can use a keyboard, "voice" to indicate that the user may use her/his voice, and "all" to say that every known input mode may be used. If, for example, the tag <p>has its out attribute set to "visual", the text contained will be delivered only through the visual mode (via a screen, for example).

Figure 8 displays the main components of the architecture of the M3L framework.The multimodal integrator is the core of the multimodal framework. It manages the overall operation logic of the system and integrates the inputs coming from the various connected channels and modes. The integrator determines the outputs to be sent to the user and manages the synchronization betweenthe channels.

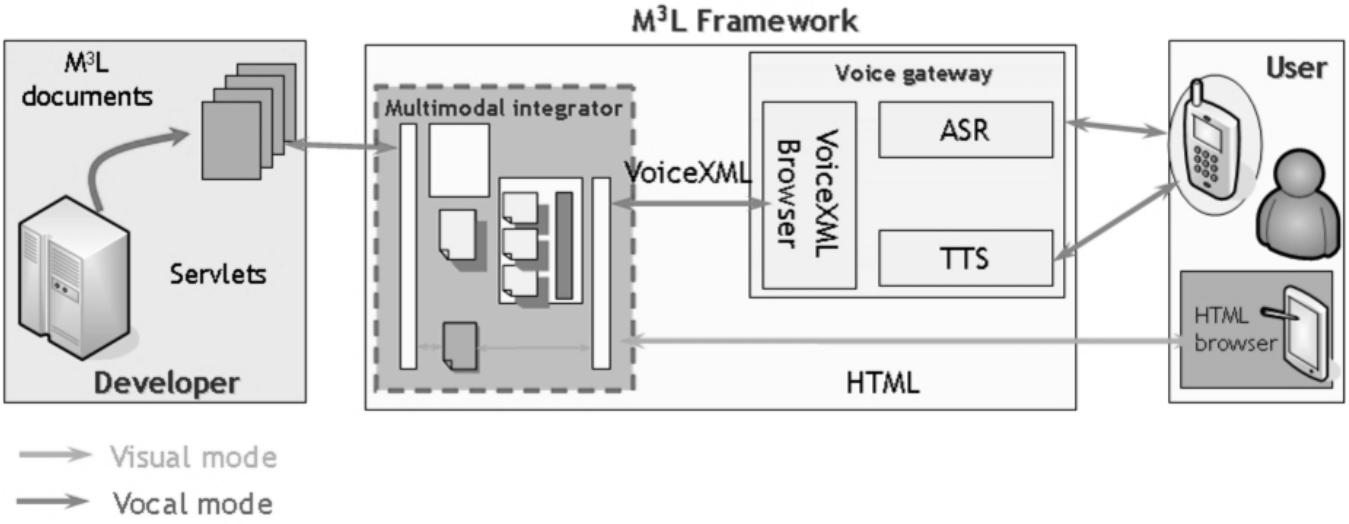

Figure 8. M3L Framework Architecture

The M3L repository is the container for the multimodal services and content. The voice server is the component that allows vocal communication between the user and the service. It receives VoiceXML documents generated by the multimodal integrator for communication with phones, interprets them, and manages the vocal interaction withthe user. A TTS (text-to-speech) unit is used to generate the speech provided to the user, and an ASR (Automatic Speech Recognition) unit is used to manage the user's speech and to collect input data. The voice server enables the vocal interaction, allowing the transmission of voice over ordinary PSTN or GSM networks. As an alternative, it is possible to send the VoiceXML file directlyto the client, as long as he/she has a suitable vocal browser.

5. Multichannel Delivery Environment

5.1. Dynamic Presentation Manager

The Dynamic Presentation Manager (DPM) is a software module for adaptive presentation of information depending on the delivery environment. The adaptation is based on the current operative context. At run-time, when the user gets to a particular page, it will be possible to personalize and customize the presentation of the information, in a way that depends on the current context in which the user is involved. In this way, a context-aware application is generated. In MAIS, the DPM is used for adapting pages designed with WebML and generated through the WebML code generator. Therefore, the architectural pattern used in our approach is based on the MVC (Model View Controller)design pattern, where the DPM module is located within the View layer, providing us with the advantages obtained by using this pattern. A more specific architecture that underlines the functional components of the DPM module is shown in Figure 9. Input data, in the form of situational data and application data, represent a new, large set of data called context data. In our approach, the context is based on three main entities: the user profile (subject), hardware and software device features (tool), and application data (object).

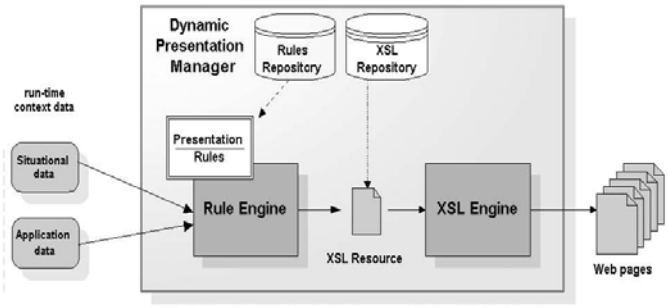

Figure 9. DPM architectural schema

An example of situational might be screen resolution =1024×768, battery level = high, memory amount =512 MB, and CPU power =1GHz; another possible example might be marital status = married, educational qualification = degree, and mother tongue =English. Application data are produced from the business logic and depend on the specific application domain. This data can be adapted, in terms of content, by other external modules using the information contained within the user profiles. The DPM adaptive presentation does not include the adaptation of content but involves only "look and feel" and layout aspects.

The Rule Engine component uses some rules called presentation rules to determine the appropriate XSL (eXtensible Stylesheet Language) file to be passed to the XSL Engine component. Presentation rules are written using the JESS (Java Expert Shell System) language. They are structured as condition

→ action. In our work, the term condition is replaced by a particular context instance and the term action is replaced by the selection of an XSL file. To select the appropriate XSL file, the presentation rules use the situational data and application data provided as input.

The XSL Engine component contains an XSL transformer, which performs a stylesheet transformation using the XSL file selected by the Rule Engine and the application data, serialized in a convenient form. The final result of this transformation is an adapted page that will be presented to the final user, in an appropriate markup language (i.e., (X)HTML or WML). The Rules Repository is the source of the presentation rules, and the XSL Repository is the source of the XSL files.

5.2. An Example of Adaptive Presentation

We have implemented a simple prototype in the MAIS project to validate our multichannel adaptive approach. At design time, using a tourist scenario, we developed the data model, the hypertext model, and the presentation model by means of WebML. Starting from this set of models, we then generated the page code with the WebML code generator. During the page generation, exploiting a functionality of the generator itself, we added a set of properties to the JSP page source code in order that these properties would be caught by the DPMto generate the adaptive presentation. At the end, we integrated the DPM with the WebML run-time platform in order to deliver information on several different channels. We accessed the system from three different devices: from a PDA, from a PC, and from a mobile phone, as shown in Figure 10. We implemented presentation rules to fit the tourist scenario, and using an XSL file for each channel/device, we produced the following results:

- replacement of widgets;
- resizing of the page fonts,
- adjustment of the page layout.

Widget replacement is important for improving the presentation on various channels and for devices with limited hardware resources (i.e., mobile phones in this case). The effects of this replacement are, for example, that images on devices with small screen dimensions are replaced by textual items, and GUI elements such as combo boxes, buttons, and window menus are replaced by lists of items.

Page font resizing is used to present information with an increased font size. In our work, this type of resizing is used to present more easily readable information either in the case of disabled users or to aid the user when in movement. Adjustment of the page layout is used to emphasize aspects of presentation and customization.

To obtain the results described above, we used several context instances structured in terms of the device configuration, such as screen resolution; the protocol being applied; the physical state of the device, such as battery level; and user activity (i.e., user in movement or not).

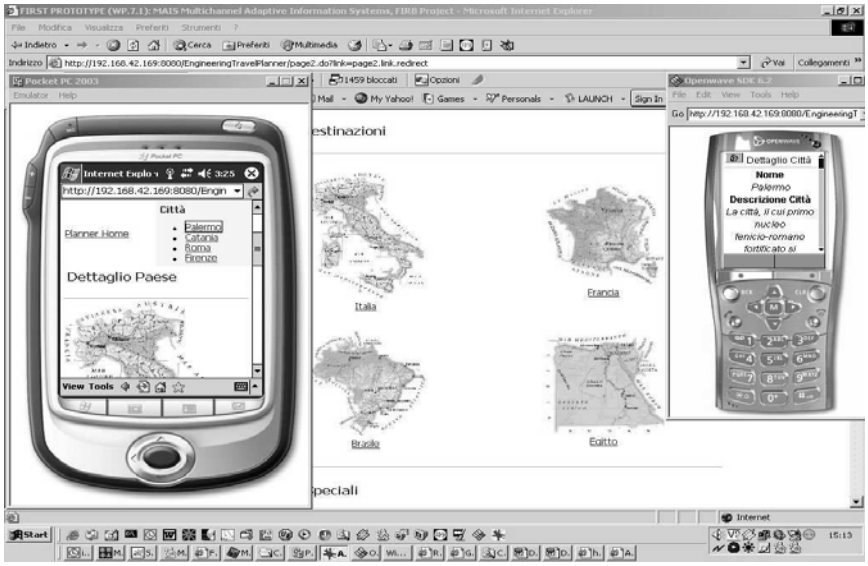

Figure 10. Multidevice access from various devices: PDA, PC, and mobile phone.

6. Conclusions

This document has presented the general framework that has been produced as the result of the 7.1 activity. The framework, consisting of a methodology and some supporting tools, has been defined thanks to a tight integration of the competencies and the efforts of the involved working units, namely PoliMI, Cefriel and Engineering. It extends the results individually achieved by each unit in a way that they can be fruitfully exploited in a comprehensive environment supporting all the design and development phases of multi-channel, context-aware Web applications. As such, we believe that the project objectives have been fully fulfilled.

In the following we list the most relevant publications, illustrating the achieved results:

1. S. Ceri, F. Daniel, M. Matera, F. M. Facca. “Model-driven development of context-aware Web applications”. To appear in ACM TOIT, 2007.
2. S. Ceri, M. Matera, F.Rizzo, V. Demaldé. Designing Data-Intensive Web Applications for Content Accessibility using Web Marts. To appear in CACM.
3. S. Ceri, F. Daniel, and F. M. Facca. Modeling Web Applications reacting to User Behaviors. To appear in Computer Networks, special issue on Web Dynamics. Elsevier.
4. S. Ceri, F. Daniel, M. Matera. Extending WebML for modeling multi-channel context-aware Web applications. Proceedings of International Workshop on Multi-channel and mobile information systems, co-located with WISE 2003, Roma, Dec. 2003.
5. M. Riva, M. Legnani; “An approach to multimodal and ergonomic nomadic services”, in J. Krogstie, E. Lawrence, B. Pernici (eds) Proceedings of IFIP TC8 Working Conference on Mobile Information Systems (Mobis2004), Oslo, Norway, September 2004.
6. D. Bolchini, S. Ceri, F. Daniel, G. Giunta, M. Legnani, L. Mainetti, M. Matera et al. Methods and Tools for the Development of Adaptive Applications. In Mobile Information Systems, B. Pernici (ed.) Springer Verlag, 2006.

References

[Antón96] Antón, A.I. Goal-Based Requirements Analysis, in Proceedings of the 2nd International Conference on Requirements Engineering RE '96, 1996.

[BM03] Bolchini, D. Mylopoulos, J., From Task-Oriented to Goal-Oriented Web Requirements Analysis, in Proc. International Conference on Web Information System Engineering WISE'03, Rome, Italy, 2003.

[MCY99] Bolchini, D. Paolini, P., Goal-Driven Requirements Analysis for Hypermedia-intensive Web Applications, in Requirements Engineering Journal, Special Issue RE03, Springer, 2004.

[BCFAB05] M. Brambilla, S. Ceri, P. Fraternali, A. Bongio, R. Acerbis. Model-driven design of service enabled Web applications. Proc. of SIGMOD 2005, pp. 851-856, ACM Press, 2005.

[BCCFM03] M. Brambilla, S. Ceri, S. Comai, P. Fraternali,I. Manolescu. Specification and design of workflow-driven hypertexts. Journal of Web Engineering, 1(2), 2003.

[CDM03] S. Ceri, F. Daniel, M. Matera. Modello per la specifica di siti Web adattativi. MAIS- Technical Report R7.1.1

[CFBCCM02] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera. Designing data-intensive Web applications. Morgan Kaufmann, 2002.

[DFL91] Dardenne, A., Fickas, S., van Lamsweerde, A., Goal-Directed Concept Acquisition in Requirements Elicitation, in Proceedings IWSSD'91, Como, 1991.

[KKP01] A. Kobsa, J. Koenemann, W. Pohl. Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. The Knowledge Engineering Review, 16(2), 2001.

[KPRS03] G. Kappel, B. Proll, W. Retschitzegger, W. Schwinger. Customization for Ubiquitous Web Applications – A Comparison of Approaches. International Journal of Web Engineering and Technology, January 2003.

[MB04] L. Mainetti, D. Bolchini. Requirements Engineering Methodology for Multichannel and Multimodal Interactive Applications. MAIS Technical Report, Nov. 2004.

[M04] MAIS. First release of a tool for the design of the personalized, multchannel Web applications, Technical Reprot MAIS P7.1.1., July 2004.

[MCY99] Mylopoulos, J., Chung, L., Yu, E., From Object-Oriented to Goal-Oriented Requirements Analysis, Communications of ACM 42 (1) (1999) 31-37.