The SERENOA Project: Multidimensional Context-Aware Adaptation of Service Front-Ends

Javier Caminero[1], Mari Carmen Rodríguez[1], Jean Vanderdonckt[2], Fabio Paternò[3], Joerg Rett[4], Dave Raggett[5], Jean-Loup Comeliau[6], Ignacio Marín[7]

[1] Telefónica R&D, Spain; [2] Université catolique de Louvain, Belgium; [3] Consiglio Nazionale delle Ricerche, Italy; [4] SAP AG, Germany; [5] GEIE ERCIM, France; [6] W4 S.A., France; [7] Fundación CTIC, Spain

Email: fjcg@tid.es, mcrg@tid.es, jean.vanderdonckt@uclouvain.be, fabio.paterno@isti.cnr.it, joerg.rett@sap.com, dsr@w3.org, jean-loup.comeliau@w4global.com, ignacio.marin@fundacionctic.org

Abstract

The SERENOA project is aimed at developing a novel, open platform for enabling the creation of context-sensitive Service Front-Ends (SFEs). A context-sensitive SFE provides a user interface (UI) that allows users to interact with remote services, and which exhibits some capability to be aware of the context and to react to changes of this context in a continuous way. As a result, such UI will be adapted to e.g. a person's devices, tasks, preferences, abilities, and social relationships, as well as the conditions of the surrounding physical environment, thus improving people's satisfaction and performance compared to traditional SFEs based on manually designed UIs. The final aim is to support humans in a more effective, personalized and consistent way, thus improving the quality of life for citizens. In this scenario, we envisage SERENOA as the reference implementation of a SFE adaptation platform for the 'Future Internet'.

Keywords: adaptation, interfaces, ontologies

1. Introduction

Today's SFEs are often designed with the assumption that they are going to be used in a single, static, predefined context of use. Such a context usually involves an able-bodied user who is using a typical set of input and output devices (e.g., a desktop/laptop computer) and who is interacting in a stable environment (e.g., home, office). Any deviation from this assumption may significantly hamper users' effectiveness — not because of any inherent barrier to interaction, but because of a mismatch between the effective context(s) of use (and its evolution over time and space) and the static, predefined context of use.

The SERENOA project[1] acknowledges that in the 'Future Internet', end users will evolve in multiple, dynamic, on-demand contexts of use, this constantly requiring a higher level of adaptation of their SFEs to fit their purpose and better address their needs and wishes for these different contexts of use. The wide availability of different computing devices and interaction resources makes this desire even stronger as the aspiration for executing the same interactive system on these different platforms is expressed, while minimizing the changes in the user interface (UI) across these platforms. There are evidences, that also industrial applications can benefit from contextual awareness, adaptation to user preferences and advanced interactional modalities as shown on the example of warehouse picking in (Ali, S., Rett, J., Lewandowski, A., 2011).

More and more, end-users want to share the same user experience independently of where and how they connect. Furthermore, adaptation to the dynamic changes produced in the environment is often not even considered by existing solutions. Analysts predict that companies are striving to provide access to their business application from all kinds of devices[2] (desktops, smartphones and tablets…). Regulations are pushing public actors to allow the disabled users to access their systems.

Currently, this diversity of needs and demands is either ignored or addressed in one of two ways: a manual UI redesign for a new context or a transcoded UI for this new context. For instance, an external assistive technology can provide a mechanism for adapting UIs to users with motor or perceptual impairments. These approaches are clearly not scalable since new contexts of use constantly appear: new users come in with different capabilities, with even more varying abilities and preferences, new devices constantly enter the market, and people evolve in environments that cannot be anticipated.

SERENOA proposes a semi-automatic adaptation of UIs involving the end user in two major ways: users can intervene in the adaptation (e.g. by controlling, suggesting, accepting/rejecting adaptations, requesting better adaptations) and the system can learn from users (e.g. by

[1] http://www.serenoa-fp7.eu/

[2] According to Gartner's Top Predictions for 2011, "By 2014, 90% of organizations will support corporate applications on personal devices" and "by 2013, 80 percent of businesses will support a workforce using tablets."

observation, by sensing, by machine learning). The adaptation will be guided by a multi-dimensional space which expresses important dimensions and the respective levels in which adaptation must be considered.

2. Guiding Principles for Context Sensitive SFEs

The SERENOA's approach towards a solution for multidimensional context-aware adaptation of SFEs is driven by the following principles:

- **New concepts, languages, runtimes and tools** are needed to support multidimensional context-aware adaptation of SFEs.
- An "**Intelligent Runtime**" should take care of adapting SFEs to the particularities and variations imposed by the target context of use in its multiple dimensions.
- **Keeping Humans in the Loop.** This principle is twofold. On the one hand, end users should be able to provide feedback or even guide the adaptation process according to their preferences or previous experiences with the system. On the other hand, authors, developers and engineers should be able to guide the adaptation process according to their experience and domain knowledge. In both cases, the system will be able to learn from humans, yielding to an open, adaptive system.
- **Open adaptiveness.** A system is open-adaptive "if new adaptation plans can be introduced during runtime" (Oreizy, Peyman, et al., 1999).
- **Covering the full adaptation lifecycle.** SFEs should be able to automatically detect information from the *context of use* (and its possible changes over time), to represent it through run-time models, to manipulate these models, and to use them in order to support a full adaptation life-cycle that will result into feedback loops (coming from end-users) in order to inform any future adaptation.

3. Overall Approach

The aim of SERENOA is to create a novel computational framework for adaptation. Such a framework will be supported by a new set of enabling technologies for context-sensitive SFEs. Such SFEs will be capable of reacting to changes in the multiple dimensions of the context of use in order to accommodate or take advantage of such changes, thus supporting human users in a more effective, personalized and consistent way.

Figure 1 depicts, from a scientific and technical perspective, the main elements (and their interrelationships) of the proposed solution. Thus, the two main domains to conduct research and development work are the computational framework domain and the technology domain. The former is devoted to devise the main scientific principles and languages. The latter is concerned with the development of the enabler technologies, their runtime support and tools.

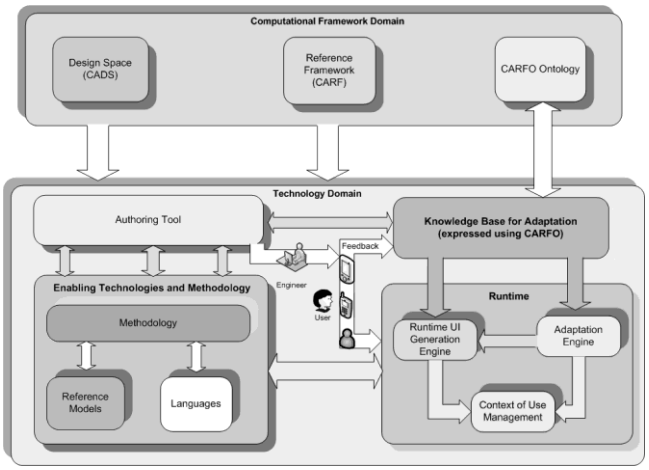

Figure 1: SERENOA's proposal towards a solution for supporting SFE adaptation

Research work targeted to devise a **computational framework** will be constituted by:

- A Context Aware Reference Framework (CARF) identifying the relevant abstraction layers for the description of SFEs sensitive to the multiple dimensions of the context of use. To this aim, a useful initial approach is the CAMELEON Reference Framework (CRF) (Calvary, Gaëlle, et al., 2003) issued by the FP5 CAMELEON project.
- A Context Aware Design Space (CADS) identifying the relevant design options and how they can vary to accommodate different scenarios and requirements. The CADS will support the whole adaptation life-cycle by allowing each step being performed by the user, the system, a third party or any combination of them.
- A Context Aware Reference Framework Ontology (CARFO) for Multi-Dimensional Adaptation of SFEs which will provide the semantic description, classification and relationships between the set of different concerns involved in advanced adaptation logic for SFEs. Such ontology will be the main formalism supporting the representation of adaptation knowledge.

The **technology domain** will be concerned with:

- The enabling technologies and agile methodology for context-sensitive SFEs, in accordance with the CARF and CADS.
- The knowledge base for representing advanced adaptation logic and expressed by means of the CARFO.
- A set of tools to support the various adaptations required by the adaptation rules, including the adaptation engine.
- A set of user interface generators to obtain implementations for various devices.
- The authoring tool to support the design and development of context-sensitive applications.

The main enabling technologies will be composed by reference models and languages. The former will be aimed at representing both the context of use and the different layers (and their interrelationships) that take part in an adaptive service front-end (tasks and domain, abstract UI, concrete UI, modalities, mappings, transformations, …). The languages that are under development are:

- An Advanced Service Front End – Description Language (ASFE-DL), for describing interactive applications at both abstract and concrete levels;
- An Advanced Adaptation Logic – Description Languages (AAL-DL), for describing the possible adaptation rules triggered by contextual changes and the associated effects in the interactive applications.

For the ASFE-DL, the project is expanding upon existing UI description languages such as MARIA (Paternò, F., Santoro, C. and Spano, L.D., 1999) and UsiXML (Limbourg, Q., et al., 2005). They cover UI description and thus will have to be widened with adaptive-oriented concepts to cover the needs of SERENOA. MARIA provides abstract and concrete languages. The concrete ones are refinements of the abstract language for various target platforms, which support different interaction modalities: graphical desktop, graphical mobile (including the possibility of touch-based interaction), vocal, and graphical and vocal compositions.

Together, the reference models and languages will be proposed for adoption in the open standards arena, particularly in the W3C working group (recently launched (W3C, 2011)) originated by the W3C's Model-Based UI XG (W3C, 2005). The methodological aspects will be enforced by an agile methodology that will cover the complete life cycle of advanced context-aware adaptation relying on the reference models and languages.

The "Adaptation Knowledge Base" will contain (in a machine understandable format) all the necessary knowledge to perform multidimensional adaptation of SFEs. It will include all the rules, conditions and reactions to be performed in response to variations in the context of use. The population of this "Adaptation Knowledge Base" is a critical issue in the project. It is planned to use different knowledge sources, such as existing adaptation tools, user communities (for instance the accessibility community) or domain experts. Such a knowledge base will be dynamic in the sense that it will incorporate new knowledge coming from end user or developer feedback.

The development of the runtime infrastructure will be concerned with implementing the adaptation engine and the runtime support for different platforms or toolkits.

The adaptation engine will be composed of a number of adapters that are under development. The adapters are able to parse logical UI descriptions and transformation rules in order to generate logical descriptions of UIs adapted to the target devices. The logical UI descriptions can be obtained through either specific authoring environments or reverse engineering tools, which take existing implementations and build automatically the corresponding logical descriptions.

Our aim is to support adaptation for various interaction modalities. Indeed, one adapter under development supports transformation of graphical web pages into vocal application, which are then implemented in VoiceXML. The adaptation transformation for vocal devices is composed of various phases: it first processes the graphical logical descriptions of the original web page by removing elements that are unimportant and complicate the overall conversion into a vocal description. Then, the graphical elements are mapped onto vocal ones having the same effects, and lastly, a main vocal menu is introduced in order to facilitate access to the parts of interest. We deemed useful to consider also the vocal interaction given the technological evolution in this area (see for example SIRI for iPhone or Google voice), which has made it suitable for various scenarios.

For the connection of the runtime modules, the considered framework (Cavazza, Marc, et al., 2010) is a hub-based message-passing framework using XML formatted messages over plain text sockets.

An authoring tool will help designers, engineers and web authors to create easily context-sensitive SFEs. Such tool will support the editing of interface descriptions according to the ASFE-DL and adaptation rules according to the AAL-DL, and will simplify the development of interactive adaptive SFEs for different platforms, which may also use different interaction modalities, e.g. graphics, voice, touch etc.

The SERENOA approach and results will be illustrated and validated during the unfolding of the project by three scenarios depicting how rule adaptation techniques can apply to different functional domains.

Lastly, to make sure that the R&D work remains consistent with the market expectations in terms of context aware adaptation of SFEs, an Industrial Advisory Board (IAB) will be held yearly. During the IAB, actors representing the industry are invited to discuss the SERENOA project results and to gather constructive feedback.

4. Computational Framework

As mentioned above, the computational framework of SERENOA is structured by three main components: the CARF, the CADS, and the CARFO.

The Context-Aware Design Space (CADS) is a graphical representation of relevant dimensions of adaptation. These dimensions, as well as their respective granularity levels, are represented by the orthogonal axis. Adaptive applications can be analyzed and compared by means of CADS.

CADS benefits include its extensibility (since additional dimensions and granularity levels can be incorporated), its flexibility (since dimensions can be included or removed, aiming a more focused analysis), exploratory (since it provides a unified view of all dimensions and their coverage levels simultaneously), and it is descriptive (since all dimensions and their granularity levels are well defined).

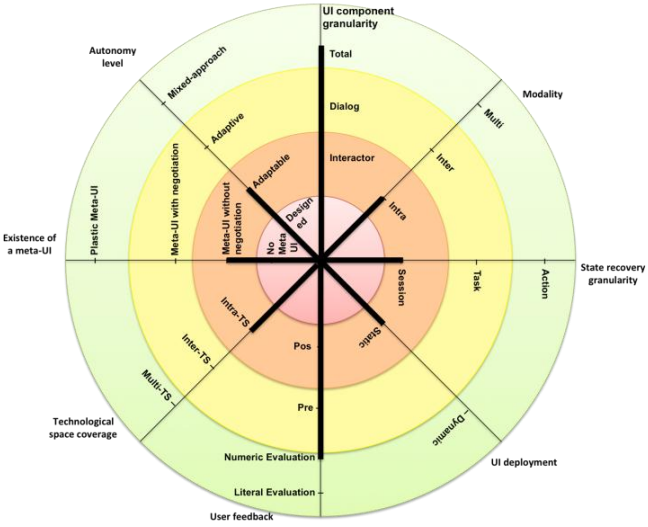

Figure 2 : CADS

Figure 2 illustrates the CADS applied to the analysis of a given application. In clockwise sense, each dimension is defined as follows:

- **UI Component Granularity**: defines the level of granularity that is subject to adaptation, e.g. one edit box, a window, or the complete application;
- **Modality**: refers to the modality types involved in the adaptation process, i.e. *intra* when the same type is considered, *inter* when the type changes, and *multi* when multiple types are available;
- **State Recovery Granularity**: refers to the impact in the user interaction, i.e. if the user needs to start a new *session*, if the user re-starts from the *task* level, or if the user re-starts just her *action*;
- **UI Deployment**: defines whether it is *dynamically* executed, or *statically* executed;
- **User Feedback**: if users can accept or reject the adaptation after (*pos*) or before (*pre*) it is performed, or if the users can evaluate it *numerically* or *literally*;
- **Technological Space Coverage**: if the technologies involved in the adaptation are all of the same (*intra*), switch types (*inter*), or if multiple technologies are involved (*multi*);
- **Existence of a Meta-UI**: the Mata-UI is an abstract model able to govern the adaptation process; it can be absent, or a meta-UI without negotiation (i.e. pre-defined), with negotiation (i.e. able to evolve), or a plastic meta-UI (capable of automatically adapting across contexts)
- **Autonomy Level**: *designed* systems are pre-defined by default and no adaptation is performed, *adaptable* ones allow users to intervene, *adaptive* ones are automatically adapted and *mixed-approach* ones combine user and systems' adaptations.

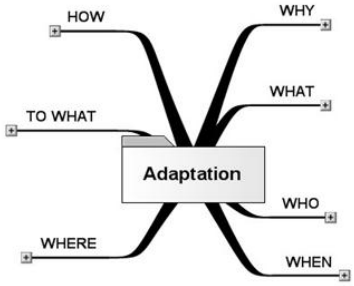

Figure 3: CARF

The CARF, as Figure 3 illustrates, is composed of 7 distinct dimensions that define:

- **What**: the resources subject to adaptation, i.e. the navigational flow, the contents (of any type, like audio, text, or images) or the presentation;
- **Why**: regarding the software qualities, as to improve the performance or the accessibility levels;
- **How**: which are the techniques, methods and strategies applied to adapt (e.g. improve the contrast level, by changing the colors of the background and text, with a smooth transition);
- **To What**: defines the context information that is taken into account to perform adaptation, mainly regarding User, Platform and Environment;
- **Who**: the agents responsible for triggering an adaptation process, such as the end user, a third-party or a developer;

- **When**: if the adaptation occurs at run-time, design-time, compilation time;
- **Where**: adaptation can be performed at the client, server, proxy, or mixed approaches.

The CARF can not only be used before the implementation, to provide stakeholders alternatives for developing an application, but also after it was implemented to analyze additional possibilities that were not considered before.

The CADS and the CARF are complementary approaches that provide a theoretical methodology to support the implementation and analysis of adaptive and adaptable applications.

5. Technology Domain

The SERENOA Framework is based on a modular architecture. In the next subsections, the different modules are presented in detail.

5.1 Agile Methodology for adaptive UIs

As mentioned previously, the agile methodology is set in the context of general software development life-cycles and specific SERENOA languages. This aims towards a methodology which fuses agile methods, e.g. Scrum and product development life-cycles, e.g. User Centered Design (UCD) with the SERENOA adaptation rules. An attempt to fuse agile methods and UCD for the development of model-based UIs has already been reported in (Losada, B. et al., 2011). The key issue in SERENOA is the cyclic collection of adaptation rules from the user by a member of the development team. This team member should hold the role of an end user experience expert and follow mainly an UCD approach. A Scrum setting for the development team will ensure that the collected adaptation rules will be implemented and evaluated within two Sprints (2 – 8 weeks). This ensures an early and continuous feedback on UI adaptation from the user.

5.2 CARFO Ontology

The CARFO is meant to represent the different concepts expressed in the CARF and the CADS.

The CARFO is divided also into several sub-modules. It is extensible so it is open to include in a next future new sub-modules improving the expressivity of the ontology and opening space for the enhancement of existing modules in the SERENOA architecture and the creation of new ones.

One of the most relevant modules in the CARFO is the Context of Use module. It provides the formal definition of concepts handled by the Context Manager module and the relationships among them. This facilitates reasoning on the current context conditions of the users of a SERENOA application. It includes not only dynamic information such as user location (GPS, WiFi, GSM cells), user preferences, user movement (GPS, accelerometer, compass), measurement of environmental noise, measurement of environmental lighting or device status (CPU, memory, storage available), but also static information about devices after populating the ontology from any existing Device Description Repository (DDR). This ontology population of device descriptions will allow providing developers with their own device classifications, including classification criteria not in the device knowledge base itself, but in queries in SPARQL language. In this way, each developer can divide devices into families under their own different criteria –as opposed to the current approach of including static classification of devices annotated in the DDR, thus being shared by all users.

Different other modules express all the concepts in the rest of the CARF and CADS, such as which adaptation techniques and methods per resource type are supported by the SERENOA framework, to what users it is devoted (target audience), target platforms (interaction modalities, operating systems, input devices, output devices, browser, device type, device properties –note that target platforms can be solved by means of a query to the Context of Use module and, potentially, to an underlying DDR technology), environment concepts handled (location, external events, presence and arrangements, time, …), where each type of adaptation takes place (client-side, server-side, intermediate proxy), what resources are subject to adaptation, who triggers the different adaptation processes (developer, end user, system, third-party) and when the distinct adaptation tasks take place (design time, compilation time, runtime).

Different other modules in the SERENOA framework may make reference to the concepts defined in the CARFO. In this sense, it is used as a common vocabulary to express, for instance, the various items appearing in the antecedent of a rule expressed in the AAL-DL.

5.3 Context Manager Module

Our goal is to be able to gather information related to all the possible contextual dimensions such as the user (tasks, preferences, state...), the technology (device, browser, connectivity...), the environment (noise, light...), social relations (privacy, collaboration, …).

The information that can be gathered by the Context Manager module may be classified in two main groups, according to the moment at which such information may be obtained. On the one hand, there is **static** information, which can be known beforehand and does not usually change over time. On the other hand, there is **dynamic** information that, due to its evolving nature, can only be retrieved at application execution time.

The context management support is an interactive deposit, accessible to external modules and devices, able to store information and to provide it upon request. Any kind of information can be modelled so as to be manipulated by the context management: for instance, the fact that a user is

using a particular device to access a specific web page, or that he is crossing a certain district might be stored and updated real time.

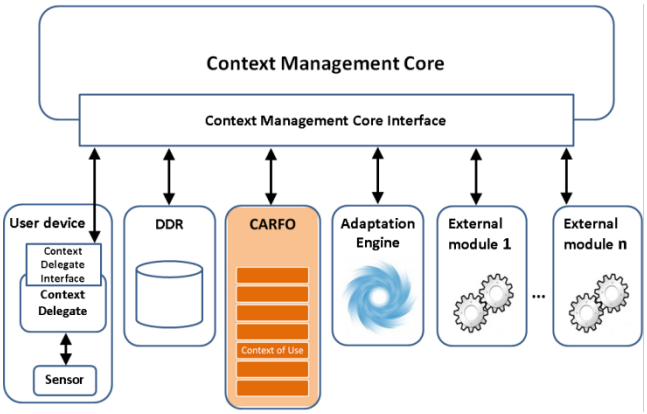

Figure 4: Architecture overview of the Context Manager

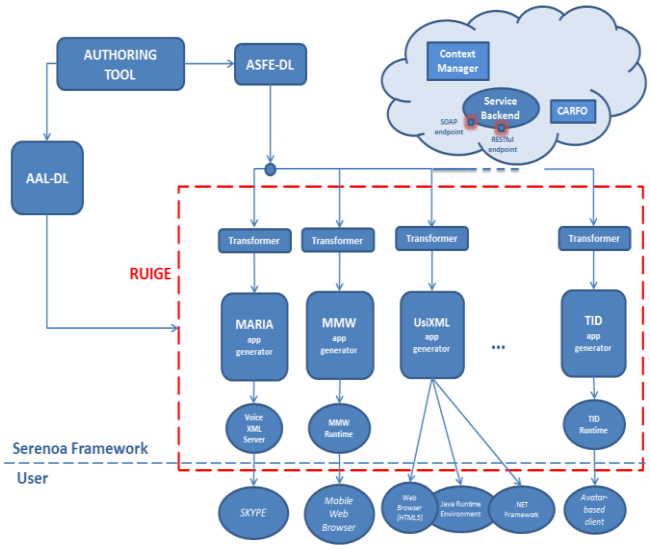

Figure 5: Architecture overview for the Runtime User Interface Generation Engine, including some possible sub-modules

As shown in Figure 4, different modules of the SERENOA framework may provide the Core Interface with dynamic context information (for instance, extracted from device sensors at client side) or static context information (such as a Device Description Repository). The CARFO might also be queried for static or dynamic information (after a population from the same context delegates at client-side or from a DDR), if semantic processing is required.

The Context Management Core can be defined as a database storing all the context information for each user session in one or more SERENOA applications. It may be accessed by applications through as a RESTful endpoint, a SOAP endpoint or a web socket endpoint (lighter, for HTML5 applications).

5.4 Runtime UI generation module

The Runtime User Interface Generation Engine (RUIGE) is in charge of the generation of SERENOA interactive applications from their definition in the ASFE-DL to a final user interface specification according to the rules described through the AAL-DL. Actually, this component consists of a set of RUIGE sub-modules. A RUIGE sub-module is a software component able to execute an application after its definition in ASFE-DL and AAL-DL. In addition, it may use the functionality exposed by the Context Manager, the API to access the CARFO or, of course, the backend services used by the application (which are SFEs). Each RUIGE sub-module is in charge of developing applications for different target platforms, which can differ in terms of interaction modalities, target audiences, etc.

As suggested in Figure 5, different sub-modules are being considered in the SERENOA framework. The MyMobileWeb (2008-2011) sub-module generates mobile web applications (WML, HTML4, XHTML Basic/MP and HTML5), the MARIA sub-module generates vocal interfaces in Voice XML as well desktop and mobile applications in HTML5/4 and JSP when accessing Web services is required, the UsiXML sub-module generates desktop applications (HTML5 and Java Swing), the Avatar submodule generates graphical interaction by means of an avatar and the LEONARDI generator allows the creation of desktop business applications which adapt to changes in the data model at runtime.

An interesting possibility for RUIGE modules is that each RUIGE sub-module publishes a manifest which indicates the target audience for the applications that it can generate, the target platforms, the kind of adaptations carried out, their purpose and on resources, where each type of adaptation takes place, who triggers each adaptation task and when they take place.

In this way, it would be possible (if all the RUIGE sub-modules publish their manifest) to programmatically decide which RUIGE sub-module(s) is/are better for each application in the mind of a developer.

5.5 Adaptation logic module

One component of the adaptation part in the SERENOA architecture is the desktop-to-vocal adapter. This module can be executed dynamically and is able to produce a VoiceXML SFE taking as input its HTML+CSS implementation. The final result adapts the hierarchical information structure of the initial SFE, by including menus that facilitate vocal navigation of the content and remove

elements not relevant for the vocal rendering. Both input and output techniques are also adapted so that they can be supported by the vocal modality. In this process, we exploit the MARIA language, which provides an abstract SFE language (with a vocabulary independent of the interaction modality) and a set of concrete SFE languages in which the abstract vocabulary is refined taking into account the target interaction platform.

The dynamic adaptation process consists of three main steps, implemented by the following modules:

- The **Reverser** parses the HTML and CSS implementation, and creates a MARIA model-based, logical description of the SFE, including contents and semantic information such as groupings, heading, navigation etc. (some are already included in the HTML, others are obtained by applying specific heuristics). This description is tied to a graphical desktop vocabulary.
- The **Graphical-to-Vocal Adapter** transforms the Reverser output into another MARIA model-based, logical description, switching the vocabulary from graphical desktop to vocal. This step consists of three sub-steps. The first performs a content and structure optimization, removing elements that are only used for layout purposes and splitting the contents so that they can be easily browsed vocally. The second step creates the vocal menus that support the interface navigation, finding meaningful labels for the identified content groups. The last one transforms the desktop elements into vocal ones (i.e. a drop-down list to a vocal choice among a predefined set of options, images to alternative text descriptions, submit buttons to confirmation prompts, etc.).
- The Voice XML Generator takes as an input a MARIA SFE description for the vocal platform and translates it into a Voice XML application that can be accessed through vocal browsers.

Since the adaptation is performed on logical SFE descriptions, it is independent on the implementation languages. In addition, the adaptation process can be customized by SFE designers through a tool that allows a set of parameters to be changed (e.g. the maximum number of items in a vocal menu). The menu structure that will be generated can also be previewed and updated when a parameter has been changed.

5.6 Orchestrator module

The orchestrator is a software module that fulfills two purposes:

The first one is to allow developers to enter the requirements of their applications in terms of concepts defined in the CARF and in the CADS and to programmatically compare them to the manifest of each RUIGE sub-module. The result of the comparison is a recommendation of the best RUIGE sub-modules to fulfill the requirements of its application(s).

The second one is to allow users to access an application from a device and decide which RUIGE sub-module must provide them with an SFE application. This includes the possibility for a user to access the application several times from distinct client devices with the orchestrator remembering the state in which the application was when it was used for the last time. In this way, the orchestrator enforces UI migration.

5.7 Authoring tools for adaptive UIs

The authoring tools will be used to support the design of model-based user interfaces which have the capability of connecting to a server-side adaptation engine. Towards this goal, the authoring tools will be able to handle the UI Description Language for SERENOA (ASFE-DL), which describes UIs on an abstract level, as well as the description language for expressing the AAL-DL.

In order to serve the different demands of the market, SERENOA will follow two technological approaches. The first approach will be a plug-in for the eclipse development tool, as this is state-of-the-art in many software companies. The second approach, based on *Quill*, will be an HTML5 browser based application and supports distributed teams of UI designers with near real-time revision control.

Both tools will also be able to create, edit and update adaptation rules expressed in the form of event-condition(s)-action and operate on different layers of the Cameleon reference framework (Calvary, Gaëlle, et al., 2003).

Quill is an HTML5 browser-based authoring tool for distributed editing of model-based user interface designs. It is being designed to work on desktop computers and tablets. Quill organizes work into projects, and holds the project data on a web server. Multiple people can work concurrently on the same project, in an analogous fashion to Google Docs. Each person can only view and edit one layer of the Cameleon reference framework at a time. Quill supports the ability to work top-down or bottom-up. It makes use of a rule engine that runs in the Cloud and responds dynamically to the changes authors make to UI models.

As an example, an author could choose to work at the concrete UI layer level on models for a desktop interface. The rule engine will then automatically update the models at the abstract UI layer, and from there propagate design changes to models at the concrete UI layer for other chosen target platforms such as mobile or TV. Adaptation rules can add tasks to Quill's Design Agenda for the authors to deal with. This is necessary when the adaptation process requires human intervention.

Quill has a modular design that separates out the authoring user interface for each layer, and makes it easy to switch the model visualization used at each layer. Layout

algorithms are used to adapt the visualization to the author's browser window size and personal preferences, and authors working on the same project will see their own visualization.

Quill is work in progress. The current focus is on developing the authoring UI, and exploring the role of animated layout techniques. A graphical visualization has been developed for the abstract UI, and plans are in place for adding an alternative visualization involving nested containers. We plan to track the emerging model specification languages from the W3C Model-Based UI Working Group, which aims to enable interface of UI designs between different authoring tools. Work has just started on the server-side adaptation engine, where we are considering the use of the JESS forward chaining rule engine.

6. Conclusions

In this paper, the SERENOA objectives and framework are presented. Currently, the project has reached the first half of its lifecycle and a demonstrator for month 18 has been produced to show the integration of the different components of the architecture. The demo shows how an application for car rental can be described using the abstract ASFE-DL SERENOA language and then versions adapted for different devices can be derived from it.

The first versions of SERENOA components have been released, including its adaptation engine, context management infrastructure and runtimes. In addition, new useful tools for the definition of adaptation concepts have been delivered, including an adaptation rule language and a first version of the knowledge base built upon the CARFO ontology.

SERENOA enters now in the critical stage for its standardization objectives: a W3C Working Group on Model-Based UIs was established has started collaborating with the consortium.

7. Acknowledgements

This work received funding from the European Commission's Seventh Framework Program under grant agreement number 258030 (FP7-ICT-2009-5).

8. References

Ali, S., Rett, J., Lewandowski, A. (2011). *A SOA based Context-Aware Order Picking System for Warehouses using Laser Range Finder and Wearable Computer*. Lucca, Italy: IEEE WoWMoM, 201, pp. 1 - 8.

Calvary, Gaëlle, et al. (2003). *A Unifying Reference Framework for multi-target user interfaces*. Interacting with Computers, vol. 15, pp. 289-308. Computer-Aided Design of User Interfaces.

Cavazza, Marc, et al. (2010). *How was your day? An Affective Companion ECA Prototype*. Association for Computational Linguistics (ACL), 2010, pp. 277-280.

Limbourg, Q., et al. (2005). USIXML: *A language supporting multi-path development of user interfaces*. Springer, 2005, Engineering Human Computer Interaction and Interactive Systems, pp. 134-135.

Losada, B. et al. (2011). *Agile Development of Interactive Software by means of User Objectives*. Barcelona, Spain: ICSEA 2011 - The Sixth International Conference on Software Engineering Advances, 2011, pp. 539-545.

MyMobileWeb (2008-2011). *"Open source software platform for the development of top-quality mobile web applications and portals, providing an advanced content & application adaptation environment"*. http://mymobileweb.morfeo-project.org, 2008-2011.

Oreizy, Peyman, et al.(1999). *An Architecture-Based Approach to Self-Adaptive Software*. IEEE Educational Activities Department, 1999, IEEE Intelligent Systems, vol. 14, pp. 54-62.

Paternò, F., Santoro, C. and Spano, L.D. (1999). *MARIA: A universal, declarative, multiple abstraction-level language for service-oriented applications in ubiquitous environments.* ACM 2009, ACM Trans. Computer-Human Interaction, vol. 16, pp. 19:1-19:30.

W3C (2005). *Incubator Group on Model-based User Interfaces.* http://www.w3.org/2005/Incubator/model-based-ui

W3C (2011). *Model-Based UI Working Group Charter*, http://www.w3.org/2011/01/mbui-wg-charter