Enriched Web-services to Facilitate the Creation of User-tailored User Interfaces

Gorka Epelde
Vicomtech-IK4
Donostia-San Sebastián, Spain
gepelde@vicomtech.org

Ane Murua
Vicomtech-IK4
Donostia-San Sebastián, Spain
amurua@vicomtech.org

Eduardo Carrasco
Vicomtech-IK4
Donostia-San Sebastián, Spain
ecarrasco@vicomtech.org

Begoña Pecharroman
Farapi
Donostia-San Sebastián, Spain
bego@farapi.com

Adriana Martínez
APTES
Donostia-San Sebastián, Spain
adriana@tecnologiasocial.org

Borja Gomez
CSS San Ignacio
Donostia-San Sebastián, Spain
bgomez@promaiorem.com

José María Susperregui
Clínica de la Asunción
Tolosa, Spain
informatica@clinicadelaasuncion.com

Agustin Aguirre
Clínica de la Asunción
Tolosa, Spain
aagirre@clinicadelaasuncion.com

ABSTRACT
This paper presents a new approach to augment web-services to facilitate the creation of user-tailored user interfaces. Our proposal is based on the ISO/IEC 24752-6: 2014 "Universal remote console -- Part 6: Web service integration" standard. This standard defines how to embed an abstract user interface layer called "user interface socket" in a web service description to allow the development of pluggable user interfaces for any type of user. Besides, a prototype of the proposed architecture has been built. In this prototype, the URC Target Kit (UTK) has been used as the URC framework implementation, deploying the framework in the same web server as the pre-existing web service. Taking advantage of the resource and grouping sheets technology of the URC framework, personalized user interfaces can be developed. The introduced method has been implemented within a remote health follow-up platform, which gives patients access to their health records and provides them with a set of health monitoring services.

Categories and Subject Descriptors
• Human-centered computing~Web-based interaction
• Human-centered computing~Interactive systems and tools
• Human-centered computing~User interface design
• Human-centered computing~User centered design
• Human-centered computing~Interaction design theory, concepts and paradigms • Applied computing~Health informatics

Keywords
Web services; Universal Remote Console; ISO / IEC 24752 -6; User Interface; e-Health; remote health follow-up; patient empowerment

MIDI '15, June 29-30, 2015, Warsaw, Poland

DOI: http://dx.doi.org/10.1145/2814464.2814473

1. INTRODUCTION
Many authors have underlined that "one-size-fit-all" user interface approach does not meet individual users need and preferences [10, 16]. This leads to a restricted user experience and even to the exclusion of some user collectives. This is especially true when it comes to remote health applications for patients [11]. In remote health applications the provision of user-tailored user interfaces can be key to achieve patients' adherence to the therapy.

Traditionally, the UI personalization has been achieved by developing different applications per each corresponding configuration need. Normally, this approach implies costly developments.

In the web world, solutions based on style sheets (CSS, XSL) have been developed, which allow modifying the presentation of the UI for different users and interaction devices.

The downside of the style-sheets-based web approach is that UI personalization is limited to the style sheet's personalization capabilities, which do not provide mechanisms for neither partial UI resource substitution nor complete UI substitution.

Several projects have developed abstract user interface (AUI) based middleware approaches allowing to exchange advanced UIs [1, 17, 27]. Middleware availability for different platforms, and the need to either run it locally or on a neighbor machine in the local environment reduces this approach's uptake.

In this paper, we first analyze the related work on the UI personalization technologies. Then, we detail our contribution, i.e., an ISO / IEC 24752 – 6 based approach to enrich web-services to facilitate the creation of user-tailored user interfaces. Next, we describe an implementation done for a remote health follow-up platform. Finally we summarize the paper and present our conclusions.

2. RELATED WORK ON UI PERSONALIZATION

2.1 Style Sheet based Web UI Personalization
Traditionally, the user interface personalization has been achieved by developing a different application per each configuration needed. In the web world, style sheets technology based solutions (Cascading Style Sheets CSS [3] or Extensible Stylesheet Language Formatting Objects XSL [9]) have been developed,

which allow modifying the presentation of the user interface for different users and different interaction devices.

The main benefits of the style sheet technologies come from the separation of the structure and the content of the presentation's document. This technology allows more precise control (outside tagging) over the spacing between characters, text alignment, the position of objects in a web page, the audio and voice output, text type characteristics, etc.

The main benefit of the web approach is that stylesheet supporting browser clients are available for almost any interaction device.

The downside of this approach is that user interface personalization is limited to style sheet personalization capabilities, and user interface composition is limited to web technologies requirements. Web technologies do not facilitate for partial user interface resource substitution or complete user interface substitution to meet users' diverse needs and preferences.

2.2 Web Service and Service Oriented Architectures

Web Service and Service Oriented Architectures have enabled service composition and personalization to be consumed by clients in different applications or business processes.

In this approach, services are defined following service interface definition standards such as Web Services Description Language (WSDL) [7] or publicly given web APIs using RESTful [25]. Service interfaces are accessed using standard protocols, such as SOAP [4] or HTTP 1.1 [12].

This approximation allows building multi-device personalized user interfaces, but it lacks any abstract user interface concept. It is rather focused in a machine-to-machine communication, which requires user interface developers to have service implementation's pre-knowledge.

2.3 Middleware Based UI Frameworks

In recent years many different projects have developed middleware approach platforms to support the user interface personalization and the composition of services mixing both local and remote services and devices.

In the Ambient Assisted Living (AAL) application domain, several European research projects have developed platforms with various specific objectives: e.g. OASIS [2], PERSONA [27], or i2home [1].

UniversAAL was a European project funded to integrate the various features developed in the aforementioned projects, providing the research and development community with a unified platform. Their deliverable D2.2C reports on their system architecture and the technology decisions made on each aspect of the middleware platform [28].

Regarding the user interaction management, they considered that only Persona [27], URC / UCH [29] developed within i2home, and the portable UI technology from Open Health tools provided a whole and consistent concept for user interaction in terms of a UI framework.

From those projects, Open Health's portable UI was discarded due to its immatureness. Meanwhile, AALuis [17], a project that was not considered by UniversAAL, has developed an alternative UI Framework based on Concurrent Task Trees [22] and MariaXML [23].

The UCH follows a pluggable user interface approach while the AALuis and UniversAAL follow a transformational approach.

2.4 AUI Technology Based Approaches for Web Services

In parallel to the development of middleware initiatives integrating UI Frameworks, some authors have targeted AUI based approaches applied to the web services for the generation of multi-device user interfaces. They bring the existing service UI mash-up platforms [6, 26] one step further.

This allows to extend the web services with a UI Framework or to move parts of the UI Framework logic to the web services, augmenting them through annotations.

One of initiatives in this line is the work done in the Servface project [24]. They propose to augment the existing service WSDL file by creating a separate task grain level annotation using Concurrent Task Tree [22]. This task level annotation is used to generate and evolve the user interface description to implementation using the MariaXML language.

Similarly, the OpenURC Alliance has proposed and developed an approach to annotate web services, so that the XML files (User Interface Socket and Target Description) required by the URC standard can be inherited from it. This work has been published as the part 6 of the ISO / IEC 24752 Universal Remote Console Framework standard [14].

These two initiatives allow to simplify the deployment of the UI personalization logic compared the middleware based UI framework approaches. This approach can be deployed in the same web server, and does not need to deploy a middleware on the client devices or in its local environment.

3. METHOD

In order to facilitate the development of user tailored UIs, we have selected the URC framework as the core technology to enrich traditional web services. Its pluggable user interface approach is preferred compared to the transformational approach of its main alternatives. Its external resource specification scheme is also a relevant advantage over alternative approaches.

3.1 URC Framework and Web Service Integration

URC technology is an open user interface platform, allowing third parties to create a pluggable user interface and use it with any device or service that exposes its functionality through a socket. The framework includes "resource servers" as global market places for any kind of user interfaces and resources necessary for interacting with appliances and services to be shared among the user community.

The Universal Remote Console (URC) framework [15] was first published in 2008 as a 5-part international standard (ISO/IEC 24752). The URC framework defines a "user interface socket" (or "socket" for short) as the interaction point between a pluggable user interface and a target device or service ("Target"). In the context of the URC, pluggable user interfaces are either generic, i.e., automatically generating a user interface based on the socket description, or specific to a socket, i.e., relying on hard-coded knowledge about the socket.

In 2014 some parts of the URC international standard were updated. A new part (part 6) was also published, focusing specifically on web service integration [14]. This document

defines how the target description [13] (xml file pointing to user interface socket location, and to UI building resource specification files) and the user interface socket (UIS) are embedded to the web services WSDL description.

The URC-HTTP protocol, as defined by [18], facilitates remote access by a controller to the sockets running in a URC compliant target or service implementation. This protocol defines the HTTP-based messaging and functions for a controller accessing the sockets of a target of a service.

The Universal Control Hub (UCH) is a gateway oriented architecture for implementing the Universal Remote Console (URC) framework in the digital home [29]. The UCH follows a middleware based UI framework strategy as introduced on the related work section.

Recently the OpenURC Alliance has released an Approved Technical Report (URC-HTTP Target 1.0) for the profiling of the URC technology in the context of enabling a Target to be URC-HTTP compatible [19]. An open source implementation of the URC-HTTP Target is provided by the OpenURC at [20], named as URC Target Kit for Java (UTK).

3.2 Proposed Approach

Once an existing web service WSDL description is extended, we have identified two approaches to implement the logic to take advantage of the UI plug-and-play features of the URC (access the AUI instantiation, generate UI / substitute specific UI resources / grouping, connect user tailored new concrete UIs):

- Middleware approach in which a user interface server provides access to web services. In this first approach the augmented web service is integrated as a target of the UTK (lightweight middleware being deployed in the same web-server) and provide access to the user interface socket and UI resources through URC-HTTP.
- Web service centric approach in which a Web service exposes a user interface socket via its WSDL-based interface. In this second approach the access to the augmented web service WSDL description would be provided through regular web service access (SOAP), access it through a JavaScript SOAP client [8] and implement the user interface socket instantiation in JavaScript.

The availability of an open source version of the UTK, makes the development efforts for the first aforementioned option to require less time to develop an initial prototype. Therefore the approach to integrate the augmented web service as a target of the UTK and to provide access to the user interface socket and the UI resources through URC-HTTP was selected:

Figure 1 depicts the selected approach to implement the logic to take advantage of the UI plug-and-play features of the URC for the integrated enriched web service:

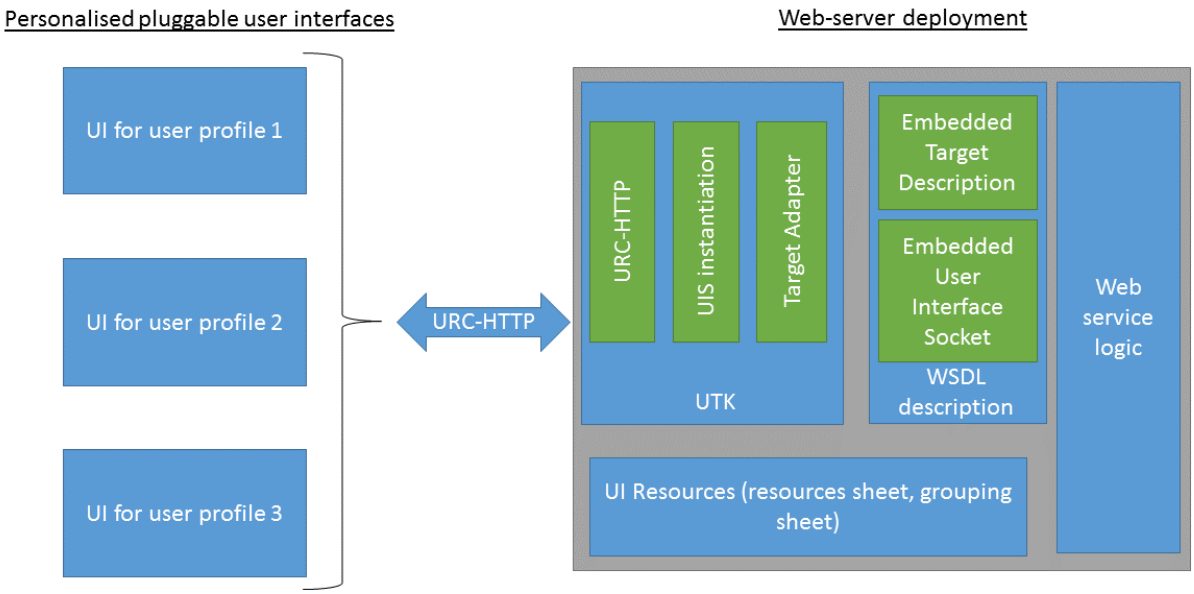


Figure 1. URC-HTTP access for UTK integrated enriched web service.

The first step is to extend the web service WSDL description with an embedded target description and user interface socket following the ISO/IEC 24752-6 standard [14]. The target description provides pointers to a target's user interface socket descriptions, and accompanying resource sheets. It may also contain pointers to external sources of resources, such as resource services. The user interface socket describes the data/interaction model of the target. A user interface socket description is composed of: a description of the functionality of the target as a set of typed variables, commands and notifications (exception-like states generated by the target).

The web service description is extended by providing a mapping description consisting of the following parts: a mapping of a target and its properties to a Web service and its properties, a mapping of each of the target's sockets, its sets and its elements to one of the Web service's partition and its elements. A Web service partition is a functional unit of a Web service. In WSDL1, this is named a port type (element <wsdl:portType>), and in WSDL2 an interface (element <wsdl:interface>).

A UIS variable is mapped using the following Web service operations: a get operation, a set operation, and a get-resources

operation (optional). A UIS command is mapped using the following Web service operations: a command operation, a get-status operation, and a get-resources operation (optional). A UIS notification is mapped to the following Web service operations: a check operation, an extend-timeout operation (optional), an acknowledge operation, and a get-resources operation (optional). Specific detail and concrete syntax for mapping is provided in the ISO / IEC 24752-6 standard [14].

Next, the UI resources that will be used in the concrete pluggable user interfaces need to be defined in the resource sheets and the grouping sheets. Resource Sheets and Grouping Sheets define concrete resources and presentation structures for the UIS that can be used to generate or adapt the user interface to the specific context of use: A Resource Sheet can contain a variety of atomic resources, in different modalities (text, image, video, or any other digital medium that can be stored as a file with a known MIME type). A Grouping Sheet defines the navigation and grouping of elements of the user interface (presentation structure).

Subsequently, the modules corresponding to the UTK need to be developed. First the web service embedded target description needs to be loaded in order to instantiate the user interface socket (whose description was embedded in the web service) and to have access to the defined UI resource files. Next a target adapter (TA) needs to be developed to implement the communication protocols to access the enriched web service. A TA is a module that communicates with a specific Target (WSDL web service in this approach) in its native protocol. TA interacts with the relevant socket instance in the UTK, to update the state of the socket values, or to receive commands from the user controllers. The TA would basically implement the access to the different web service partition using the SOAP control protocol.

Once defined the introduced description files and implemented the required integration logic, the UTK provides access to the URC features through its build-in URC-HTTP protocol module.

At this point personalized pluggable user interfaces (either by substituting UI resources or the presentation structure for identified group profiles, or by exchanging specifically developed complete user interfaces) can be deployed for the enriched web services.

4. IMPLEMENTATION FOR A REMOTE HEALTH FOLLOW-UP PLATFORM

The introduced method is being implemented within a remote health follow-up platform, which gives patients access to their health records and provides them with a set of health monitoring services. Previously, this use case was being addressed with one specific web[1] and one custom Android client app which consumed pre-existing health monitoring web-services [5]. Although these applications scored high in user acceptance (mainly because of their purpose), the patients suggested that a more personalized user experience was convenient.

A team of anthropologist is analyzing the societal and user needs of the targeted application. The aim is to better understand different user types (using Participatory Action Research approach and Ethnographic methods) and design the best experience for the defined profiles.

Following this novel approach, personalized user interfaces are being targeted to both web-browser clients and Android clients. For the development of such clients and accessing the URC-HTTP interface provided by the enriched web-service described before, the Webclient JavaScript Library [21] will be used.

Based on the identified user profiles and the defined designs, UI resources together with navigation and grouping structures will be developed for each user profile.

The presented method will allow to easily test prototype UIs combining different UI resource and grouping structures. Additionally, the adopted method will allow for easily extending the defined UIs for newly identified societal profiles or exchanging the UI resources based on the patient's feedback.

For the testing of the presented approach a proof-of-concept implementation has been done. First, the WSDL 1.1 description of the remote health follow-up has been enriched with an embedded target description and a user interface socket description. For explaining the implementation, only the vital constants submission feature will be presented in the following.

The enriched web service description[2] is composed of: embedded target description, datatype and messages used in socket and target port definitions, WSDL ports definition for the socket and the target operations, and the bindings and services to access such operations through the web service.

Next, UI resources (i.e. labels, images and navigational and grouping structures) have been defined for three differentiated profiles, targeted for: *i)* a default user with technical skills[3][4], *ii)* a simple UI for senior citizens[5][6], and *iii)* a simple UI targeted for young people[7][8].

The goal of the defined resource and grouping sheets is to test the capabilities of the presented method in prototype UIs. Meanwhile the anthropologist team is identifying the required main profiles and is shaping the final UIs and the corresponding resources according to their needs.

[1] Service provided for customers of: http://www.clinicadelaasuncion.com, accessible from: https://hhweb.clinicadelaasuncion.com/

[2] Enriched web service description https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic.wsdl1

[3] Resource sheet for default user: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic.defaultUI.en.rsheet

[4] Grouping sheet for default user: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic_defaultUI.gsheet

[5] Resource sheet for senior citizen: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic.seniorsUI.en.rsheet

[6] Grouping sheet for senior citizen: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic_seniorsUI.gsheet

[7] Resource sheet for young people: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic.youngUI.en.rsheet

[8] Grouping sheet for young people: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic_youngUI.gsheet

Subsequently, the augmented web service was integrated with the UTK by means of replicating and instantiating the web service embedded target description[9] and user interface socket[10] . Additionally, a target adapter was developed to integrate the SOAP communication protocol and to be able to communicate with the web service.

Finally, the proof-of-concept UIs were developed for the web browser, taking advantage of the Webclient JavaScript Library [21] to access the URC-HTTP interface provided by the UTK. A simple JavaScript code was developed to load and apply each UI resources defined for the pre-defined user profiles.

The resulting proof-of-concept UIs for the three profiles defined for testing the concept are presented in Figure 2 for the default user profile, and in Figure 3 (a) for the senior citizens, and (b) for the young people profile.

[9] Target description: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhweb.td

[10] User interface socket: https://github.com/epetxepe/Midi2015-paper17/blob/master/hhwebBasic.uis

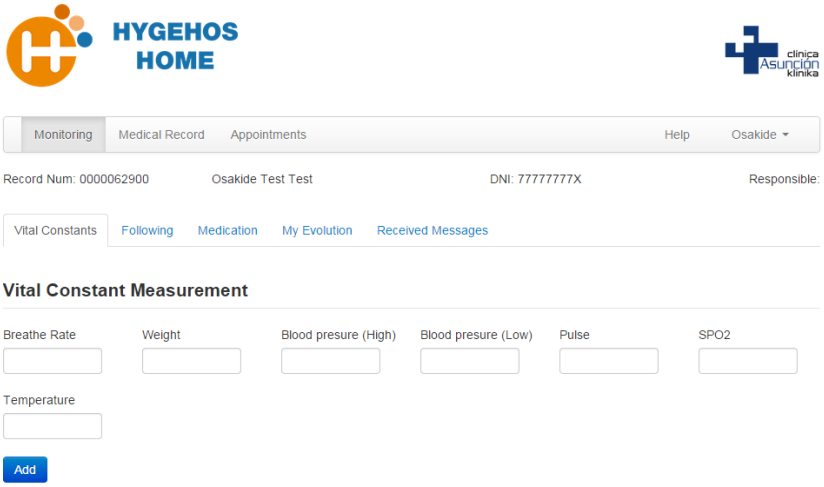
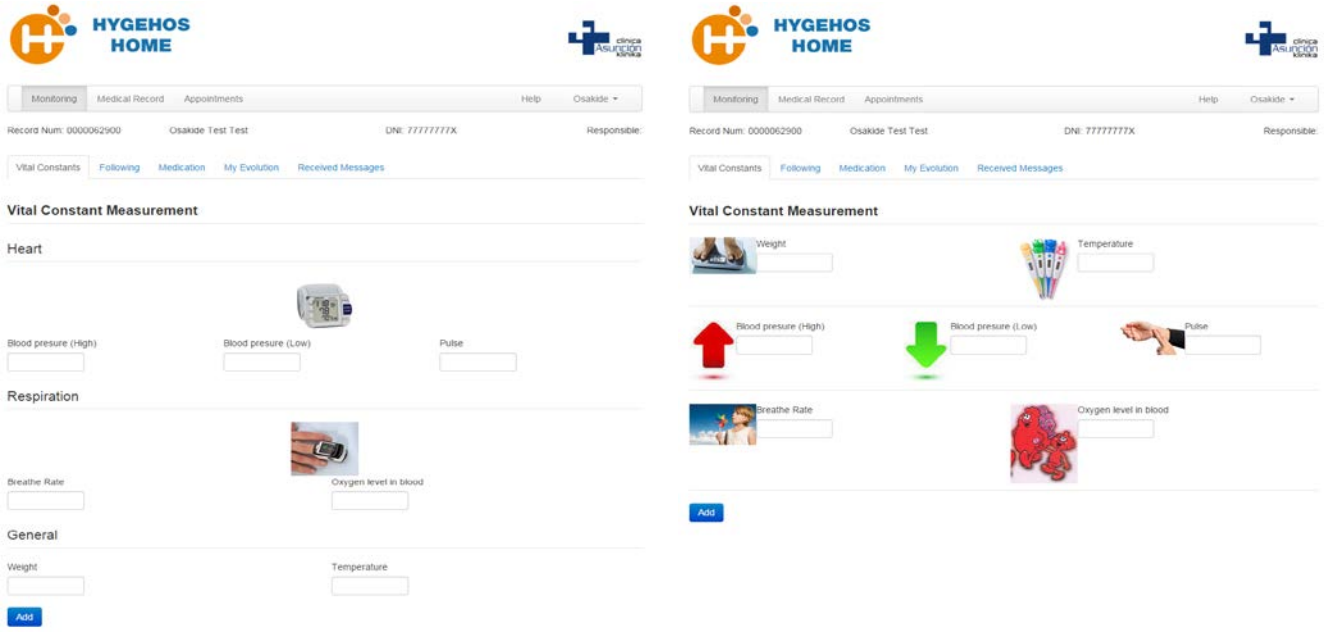


Figure 2. Proof-of-concept UI generated for a default user profile.



(a) **(b)**

Figure 3. Proof-of-concept UIs generated for seniors (a) and for young people (b)

The novel aspects of the presented work are: *i)* its user-tailored UI delivery capabilities, *ii)* the implementation of the abstract user interface logic in the webserver, *iii)* the use of open standards and open source tools, and *iv)* its application on a health care scenario.

5. CONCLUSION AND FUTURE WORK

This paper has presented a novel method to augment web-services in order to facilitate the creation of user-tailored user interfaces. We first have analyzed the existing technologies to provide personalized UIs. Second, we have identified the possible approaches to integrate the URC technology defined by the ISO / IEC 24752-6 into the web services. Next, we have described a novel middleware-based-architecture for deploying the solution on the server where the original web services run. Finally, we have presented a proof-of-concept implementation for a remote health follow-up platform, which provides personalized UI for three pre-defined user profiles.

The presented approach facilitates the provision of personalized user experience and goes a step further from style-sheet-based web approaches allowing to substitute UI resources, navigational and grouping structure, by integrating user interface abstraction technologies. Additionally, since the UI resource definition is externalized, it provides a perfect platform to prototype and update

personalized UIs in a process to refine the usability and the user experience of the end-user.

Future work includes: *i)* the enhancement of the UTK to automatically integrate any augmented web service, *ii)* the alternative of implementing the user interface socket instantiation on the client side taking advantage of the JavaScript SOAP library to access the user interface socket exposed by the Web service via its WSDL-based interface, iii) comparing and evaluating the performance of both approaches, and *iv)* research on adaptive UI techniques, that starting from UIs developed for generalized user profiles, manage to automatically tailor the UI to adapt to user specific needs and preferences.

6. ACKNOWLEDGEMENT

This work has been partially funded by the Basque Government GAITEK 2014 Program (Osakide). This program is supported by the Department of Economic Development and Competitiveness of the Basque Government and the European Regional Development Fund (ERDF).

7. REFERENCES

[1] Alexandersson, J. 2008. i2home–towards a universal home environment for the elderly and disabled. *Künstliche Intelligenz*. 8, 3, 66–68.

[2] Bekiaris, E. and Bonfiglio, S. 2009. The OASIS Concept. *Universal Access in Human-Computer Interaction. Addressing Diversity*. C. Stephanidis, ed. Springer Berlin Heidelberg. 202–209.

[3] Bos, B. et al. 1998. Cascading style sheets, level 2 CSS2 specification. *W3C Recommendation available at http://www.w3.org/TR*.

[4] Box, D. et al. 2000. Simple object access protocol (SOAP) 1.1. *W3C Recommendation available at http://www.w3.org/TR*.

[5] Carrasco, E. et al. 2014. Hygehos Home: an innovative remote follow-up system for chronic patients. *Studies in Health Technology and Informatics*. 207, 261–270.

[6] Casquero, O. et al. 2008. iGoogle and gadgets as a platform for integrating institutional and external services. *Mash-Up Personal Learning Environments. Proc. of 1st Workshop MUPPLE* 37–41.

[7] Christensen, E. et al. 2001. Web services description language (WSDL) 1.1. *W3C Recommendation available at http://www.w3.org/TR*.

[8] CodePlex JavaScript SOAP Client. *http://javascriptsoapclient.codeplex.com/Wikipage?ProjectName=javascriptsoapclient*. Accessed: 2015-02-26.

[9] Deach, S et al. 1999. Extensible stylesheet language (xsl) specification. *W3C Recommendation available at http://www.w3.org/TR*.

[10] Epelde, G. et al. 2013. Universal Remote Console-based next-generation accessible television. *Universal Access in the Information Society*. 12, 1, 73–87.

[11] Epelde, G. et al. 2014. Universal Remote Delivery of Rehabilitation: Validation with seniors' joint rehabilitation therapy. *Cybernetics and Systems*. 45, 2, 109–122.

[12] Fielding, R. et al. 1999. Hypertext transfer protocol–HTTP/1.1. *W3C Recommendation available at http://www.w3.org/TR*.

[13] ISO / IEC 2014. ISO/IEC 24752-4:2014 - Information technology -- User interfaces -- Universal remote console -- Part 4: Target description. *International Organization for Standardization*.

[14] ISO / IEC 2014. ISO/IEC 24752-6:2014 - Information technology -- User interfaces -- Universal remote console -- Part 6: Web service integration. *International Organization for Standardization*.

[15] ISO / IEC 2008. ISO/IEC 24752: Information Technology—User Interfaces—Universal remote console—5 parts. *International Organization for Standardization*.

[16] Kelle, S. et al. 2014. A Showcase for Accessible Online Banking. *Universal Access in Human-Computer Interaction. Design for All and Accessibility Practice*. C. Stephanidis and M. Antona, eds. Springer International Publishing. 37–45.

[17] Mayer, C. et al. 2012. AALuis, a User Interface Layer That Brings Device Independence to Users of AAL Systems. *Computers Helping People with Special Needs*. K. Miesenberger et al., eds. Springer Berlin Heidelberg. 650–657.

[18] OpenURC Alliance 2013. URC-HTTP Protocol 2.0 (ATR). Retrieved March 20, 2015 from *http://www.openurc.org/TR/urc-http-protocol2.0-20131217/*.

[19] OpenURC Alliance 2013. URC-HTTP Target 1.0. Retrieved March 20, 2015 from *http://www.openurc.org/TR/urchttp-target1.0-20150205/*.

[20] OpenURC Alliance 2013. URC Target Kit for Java (UTKj). Retrieved March 20, 2015 from *http://www.openurc.org/tools/urchttp-target-kit-java-3.1/*.

[21] OpenURC Alliance 2013. Webclient JavaScript Library. Retrieved March 22, 2015 from *http://www.openurc.org/tools/webclient-generic-html-3.1/*.

[22] Paternó, F. et al. 1997. ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. *Human-Computer Interaction INTERACT '97*. S. Howard et al., eds. Springer US. 362–369.

[23] Paternó, F. et al. 2009. MARIA: A Universal, Declarative, Multiple Abstraction-level Language for Service-oriented Applications in Ubiquitous Environments. *ACM Trans. Comput.-Hum. Interact.* 16, 4, 19:1–19:30.

[24] Paternò, F. et al. 2009. Model-Based Design of Multi-device Interactive Applications Based on Web Services. *Human-Computer Interaction – INTERACT 2009*. T. Gross et al., eds. Springer Berlin Heidelberg. 892–905.

[25] Richardson, L. and Ruby, S. 2008. *RESTful web services*. O'Reilly Media, Inc.

[26] Song, K. and Lee, K.-H. 2008. Generating multimodal user interfaces for Web services. *Interacting with Computers*. 20, 4–5, 480–490.

[27] Tazari, M.-R. et al. 2010. The PERSONA Service Platform for AAL Spaces. *Handbook of Ambient Intelligence and Smart Environments*. H. Nakashima et al., eds. Springer US. 1171–1199.

[28] UniversAAL Project 2012. D2.2C universAAL execution environment, installation packages, hardware abstraction layer, generic platform services / AAL platform services and

ontology artefacts. Retrieved March 19, 2015 from *http://universaal.org/index.php/en/about/about-deliverables*.

[29] Zimmermann, G. and Vanderheiden, G. 2007. The Universal Control Hub: An Open Platform for Remote User Interfaces in the Digital Home. *Springer LNCS. Human-Computer Interaction. Interaction Platforms and Techniques.* 4551/2007, 1040–1049.