OpenAPE

A framework for personalised interaction in smart environments

Lukas Smirek, Patrick Münster, Gottfried Zimmermann
Stuttgart Media University
Stuttgart, Germany
e-mail: {smirek, muenster, gzimmermann}@hdm-stuttgart.de

***Abstract*—In this contribution, we describe our preliminary work on openAPE - the open Accessibility and Personalization Extension framework. The main goal of the framework is to transfer platform independent context information from one device to another and infer personalized settings for user interface and device adaptation according to the user's needs. This shall contribute to improved usability and accessibility in smart environments. Two exemplary use cases are described, to illustrate in which contexts the framework can be used.**

Keywords-openAPE; Adaptive User Interfaces;Smart Environments; Eclipse Smart Home; Create@school

I. Introduction

During the last decades, information and communication technology (ICT) has found brought entrance in our everyday lives. This trend is expected to go on for the next years. Thereby, it is not only the amount of electronic devices and services that will increase but also their interdependencies and network capabilities. Overall, we can find more and more setups of smart environments in our surroundings. The field of Smart Environments and its subdomain of Ambient Assisted Living (AAL) have the high potential to support regular users as well as the elderly and people with disabilities in their everyday lives [1].

It is also expected that the way we interact with smart environments will differ from the way we interact with current ICT in the sense that interaction patterns will change from explicit to implicit and more natural ones [2][3].

Considering the huge amount of interconnected devices and new interaction patterns it must be assured that everyone, independent of age, computer skills, cultural background or disability is still able to profit from these developments. One major obstacle might however be inaccessible user interfaces (UI) [1]. Since almost everyone in the society will be affected from that, the users' requirements for accessible UIs might be very heterogeneous and sometimes even contradictory [4]. Therefore, it will be difficult to follow a one-size-fits-all approach and there will appear the need for personalized UIs in smart environments that take the individual user needs and preferences into account. Due to the huge amount of devices that users will face in future smart environments, it will be exhausting to adapt every device by hand to the user's needs. To transfer settings from one device to another is therefore a major step for enabling wide spread usable and accessible UIs. This is in line with the development goals of openAPE – the open Accessibility Personalization Extension [5].

The paper is structured as follows. In Section 2, we describe an illustrative use case and infer requirements for personalization in smart environment scenarios. The consecutive Section 3 will give a brief overview of related, existing systems. In Section 4, we will explain our approach. In Section 5, we will describe two research projects in which our framework is used.

II. Use case and Requirements for personalisation in smart environments

Let us think about a visually impaired businessperson from Germany. In Germany, he is living in his own apartment equipped with different devices like a smart TV, a lighting system and a smart heating system. He has configured his home in a way, that when he switches to TV mode the TV set is switched on and the light is dimmed down. He has also set a preferred room temperature. He can control the status of his home via his smart phone. The smart phone is configured with large font size and strong contrast.

Now, the businessperson has to travel to China. When he arrives in his hotel room, he immediately notices that the air-conditioning system has cooled the room too much. He approaches the control panel at the wall that is connected to the openAPE infrastructure. The businessperson authenticates himself via a RFID tag and the panel connects to the openAPE infrastructure to look up the users preferred settings. Among other information there is stored that the user has configured its smart phone with a larger font size and stronger contrast and that his preferred language is German. Therefore, the control panel reads aloud a short welcome message with some basic explanations. It also increases the font size and contrast and downloads all text labels for the UI in German language. Furthermore, it proposes the user's preferred room temperature.

Some weeks later, a deaf person stays in the same hotel. For him there are no adjustments made with regard to font size and contrast. However, since the person has problems with written language, for him the welcome message and all help texts are displayed as sign language videos. With regard to this use case, the following requirements for personalization in smart environments can be deduced:

When looking at different systems from the fields of smart homes, AAL, context aware computing and mobile computing, the following development goals can be identified:

- Interoperability: users will have to cope with different back-end technologies that must be integrated (e.g., in Smart Homes the integration of existing and new devices/systems).
- Device and service overarching use cases: in most cases, users do not want to make use of a single device or service only. Instead, several ones should be integrated to help the user perform a task.
- Adaptive UIs: to provide the best user experience, a UI should adjust to the context of use (user preferences, environmental conditions, technical conditions and the current task). It is also important, that in smart environments the possibilities of traditional GUIs are restricted and adaptivity must be considered from a generic interaction perspective <>.
- Adaptivity independent of controller device and place: in smart environments, users will move around and will interact with different devices. Hence, it is important that user preferences required for adaptation can be shared between devices.
- Openness for third party contributions: for the average UI developer it is difficult to develop user interfaces for people with disabilities, hence the system must enable the injection of expert knowledge [4].

III. Related work

The Eclipse Smart Home project [6] deals with interoperability problems and device abstraction in Smart Homes. The existing UIs and rule engine enable device overarching use cases and automatization. However, UI personalization can be achieved only to a very low degree by hand[7]. Other frameworks like AllJoyn [8] or OCF [9] provide abstract descriptions of device functions and states that can be accessed by a UI. The device models could be used to auto-generate UIs. However, currently there seem to be no adaptation engines available. Furthermore, the models mainly contain information about data types that shall be displayed in UI elements. Anyway, Mayer at al. [10] claims that this kind of information enables the generation of very simplistic UIs only. The authors argue that not only data types, but also the semantic of the interaction should be modeled. The authors present their own solution accordingly.

Projects like Supple [11] or MyUI [12] have tried to provide adaptive UIs, but relay on application models that do not abstract from devices and backend technologies and are consequently difficult to use in smart environments. The Universal Remote Console (URC) [13] and its runtime implementation the Universal Remote Hub [14] provide abstract device descriptions and a mechanism for exchanging personalized UIs to one or several devices. Furthermore, the URC framework leverages the concept of a resource server to provide specialized UI resources. This enables third party UI contributions like labels in different languages, icon sets, sign language videos, etc., (e.g., from accessibility experts) [15], even at runtime. However, an adaptation engine is missing.

MyUI provides a mechanism for third party contributions. Nevertheless, in the URC framework, specialized UI content can be provided, while in MyUI only a generic interaction pattern for a certain interaction situation can be contributed, but no content.

The Global Public Inclusive Infrastructure [16] provides a mechanism to transfer platform independent user preferences from one device to another, but lacks an adaptation engine.

IV. openAPE

When developing the openAPE framework [17][5] the focus was to address the following requirements:

- enable a platform independent mechanism to transfer context of use data from one device to another
- provide adaptation and UI settings information independent of place
- Enable third parties to contribute specialized content
- Provide specialized content independent of place.

Considering these developments, it is clear that there are overlapping with some other technologies, mainly GPII and URC. Nevertheless, there are some important differences. Similar to URC, openAPE implements the concept of a resource server. However, OpenAPE ships with a context management infrastructure, something that is missing in URC.

GPII also provides a way to exchange context data (mainly user preferences regarding UI settings). OpenAPE differs in this case in the sense that it is not implemented as a monolithic system like GPII; instead, it is a very lightweight RESTFUL web service. It is also further in its API specification. OpenAPE is the reference implementation of ISO/IEC 24752-8 [18]that has already reached the status of a Committee Draft.

A. *Main components*

The openAPE infrastructure shown in Figure 1 is based on the specifications defined in ISO/IEC 24752-8 [18]. The main services are the following:

- Context services that can be used by any device to upload user preferences/settings, equipment, environment and task contexts (context of use) in order to make them globally available.

- Listing service that can be requested to get recommendations for UI settings and adaptations
- Resource service to make additional UI components available
- A feedback service to rate the proposed solution.

B. *The related workflow is as follows:*

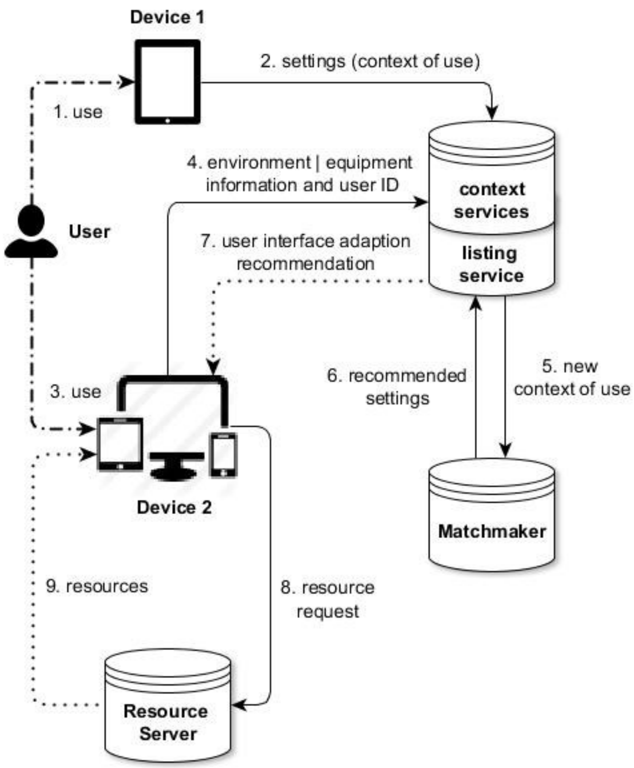

Figure 1: OpenAPE Architecture

1. A user personalizes a device according to his needs (e.g., font size, language etc.).
2. The device creates different context objects that contain the relevant user settings and for which context they were made. These contexts are uploaded to the corresponding context web services.
3. In a next step, the user can approach any other device connected to the openAPE infrastructure and can authenticate himself.
4. After the authentication, this second device uploads the current context conditions (equipment, environment and task context).
5. In a next step, it sends a request message to the listing service in order to obtain information about optimized UI settings and additional UI resources. Thereby it refers to the uploaded context information.
6. The listing service starts a matchmaking mechanism to infer the recommended UI settings and adaptations.
7. The listing service exposes the recommended settings to the client.
8. The client downloads the recommendations and adjusts its UI.
9. If mentioned in the recommendations, the device can download additional UI resources.
10. Optionally, the client gives feedback on the quality of the recommended settings to openAPE.

V. APPLICATIONS

A. *Eclipse Smart Home*

Smart Homes and AAL yield the high potential to enable a longer independent life for the elderly and people with disabilities. However, as mentioned before, such environments must be adjusted to the users in order to let them exploit the full advantages of these technologies. The Eclipse Smart Home (ESH) is an open source framework addressing not only the field of smart homes but also of AAL. As pointed out in [7] personalization features are currently not very far developed yet. Nevertheless, the ESH framework provides enough connecting factors that enable the establishment of personalization features. Therefore, concepts from the URC framework and a connection between ESH and openAPE will be utilized.

Our goal is to develop a module that is deployed inside the ESH runtime and that connects to the openAPE infrastructure. The module can upload different context data such as devices being connected to the ESH server (equipment context) or environmental data (environment context). These data are than used to either download additional UIs or configuration data and automation rules. Downloading automatization rules goes far beyond personalization of UIs. They enable to personalize the behavior of the whole environment.

B. *Creat@School*

Playing games is a popular leisure activity for young people, it makes them focusing onto problem, accept challenges and push them further. Also creating games is a very motivating challenge, but creating a game seems to be a difficult task. Therefore, the Pocket Code app was created. It allows students to program small applications easily directly on their smartphone without the requirement for any additional hardware or learning a programming language. Within the context of the “No One Left Behind” (NOLB) project, mobile game-based learning should be integrated into school curricula.

Create@School is an enhanced version of Pocket Code which integrates the results of the pilot studies with teachers and students. One of these extensions is our integration of personalization features in Create@School to use the openAPE framework and to make the app more useable for various user groups. Software programs, mobile apps and websites have a default UI that tries to cater for many people, but that is often unsuitable for people with special needs. Many such programs are adaptable, but onside observations has shown that most people never adapt the settings of the software they use. This may be because they think the default settings are all there is, because they are afraid of breaking something or because it is too difficult. Another reason is that devices at schools are not used every lesson by the same student. For this reason, the students can choose a predefined profile in Create@School whenever they want. The chosen profile only influences the Create@School app and not the general device settings.

Therefore, we have provided the profile changing option direct within the Create@School settings menu and have made them selectable by the name of mythological characters to make them more distinctive and attractive for the user.

At the moment Create@School has five profiles (includign the standard profile) to personalize the UI. Individual settings are stored on the device and not in the openAPE framework. Therefore, if a pupil uses another device, his settings are not available on this one and everything must be customized manual again.

For this reason, the next step is to develop GPII enabled featured to provide individual profiles which are stored in the cloud and which are available on every device. All these further individual personalization features will be developed in the openAPE project, which will provide GPII enabled services to auto-adapt the Create@School UI to the user's needs and desires.

VI. Conclusion

At this stage, the matchmaking algorithm to infer the settings recommendations is a very simplistic one. In the future, we will work towards a more advanced one. Rule based solutions as well as such that utilize concepts from machine learning can be thought of. Furthermore, we will work on use cases in the field of smart homes and e-learning. Furthermore, the system must be evaluated with regard to different dimensions. First of all, there is the technical dimension. Is the REST infrastructure robust under high load, especially if there are frequent changes of context conditions? Next, it must be evaluated, wheater developers of adaptive applications see a benefit in using a REST API?

Acknowledgment

The research leading to these results has received funding from the European Union Seventh Framework Program (FP7/2007-2011) under grant agreement no. 610510, Prosperity4All (”Ecosystem infrastructure for smart and personalized inclusion and PROSPERITY for ALL stakeholders”) as well as under the EC H2020 Innovation Action No One Left Behind, Grant Agreement No. 645215. This publication reflects only the authors’ views and the European Union is not liable for any use that may be made of the information contained herein.

References

[1] P. L. Emiliani and C. Stephanidis, “Universal access to ambient intelligence environments: opportunities and challenges for people with disabilities,” *IBM Systems Journal*, vol. 44, no. 3, pp. 605–619, 2005.

[2] J. Nehmer, M. Becker, A. Karshmer, and R. Lamm, “Living assistance systems: an ambient intelligence approach,” in *Proceedings of the 28th international conference on Software engineering*, 2006, pp. 43–50.

[3] M.-R. Tazari, “An open distributed framework for adaptive user interaction in ambient intelligence,” in *International Joint Conference on Ambient Intelligence*, 2010, pp. 227–238.

[4] M. Peissner, A. Schuller, D. Ziegler, C. Knecht, and G. Zimmermann, “Requirements for the Successful Market Adoption of Adaptive User Interfaces for Accessibility,” in *Universal Access in Human-Computer Interaction. Design for All and Accessibility Practice*, C. Stephanidis and M. Antona, Eds. Springer International Publishing, 2014, pp. 431–442.

[5] “openAPE,” *Home*, 2017. [Online]. Available: http://gpii.eu. [Accessed: 26-Feb-2017].

[6] “Eclipse SmartHome - A Flexible Framework for the Smart Home,” 2016. [Online]. Available: http://www.eclipse.org/smarthome/. [Accessed: 26-Feb-2017].

[7] L. Smirek, G. Zimmermann, and M. Beigl, “Just a Smart Home or Your Smart Home – A Framework for Personalized User Interfaces Based on Eclipse Smart Home and Universal Remote Console,” *Procedia Computer Science*, vol. 98, pp. 107–116, 2016.

[8] AllSeen Alliance, “AllJoyn Framework,” 2016. [Online]. Available: https://allseenalliance.org/framework. [Accessed: 26-Feb-2017].

[9] Open connectivity foundation (OCF), 2016. [Online]. Available: https://openconnectivity.org/. [Accessed: 26-Feb-2017].

[10] S. Mayer, A. Tschofen, A. K. Dey, and F. Mattern, “User Interfaces for Smart Things,” *Institute for Pervasive Computing, Zurich*, 2014.

[11] K. Gajos and D. S. Weld, “SUPPLE: automatically generating user interfaces,” in *Proceedings of the 9th international conference on Intelligent user interfaces*, 2004, pp. 93–100.

[12] Matthias Peißner, “Entwurfsmusterbasierter Ansatz zur Überwindung von Nutzungsbarrieren,” Fraunhofer IAO, Universität Stuttgart, 2014.

[13] G. Zimmermann *et al.*, “Universal Remote Console Standard - Toward Natural User Interaction in Ambient Intelligence,” in *Extended abstracts of the 2004 conference on Human factors and computing systems - CHI ’04*, New York, New York, USA, 2004, p. 1608.

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

[15] G. Zimmermann and B. Wassermann, “Why We Need a User Interface Resource Server for Intelligent Environments,” in *Workshops Proceedings of the 5th International Conference on Intelligent Environments*, 2009, vol. 4, pp. 209–216.

[16] "GPII," *Global Public Inclusive Infrastructure*, 2016. [Online]. Available: www.gpii.net. [Accessed: 26-Feb-2017].
[17] "GitHub - REMEXLabs/OpenAPE: Settings-based personalization framework," 2016. [Online]. Available: https://github.com/REMEXLabs/OpenAPE. [Accessed: 26-Feb-2017].
[18] ISO/IEC 24752-8, "Information technology — User interfaces — Universal remote console — Part 8: User interface resource framework (Committee Draft)." 2017.

 ISBN: 978-1-61208-589-0