Giving Elderly Access to Smart Environments

Providing Adaptive User Interfaces Based on Web Components, the Universal Remote Console and the Global Public Inclusive Infrastructure

Lukas Smirek, Alexander Henka(✉), and Gottfried Zimmermann

Stuttgart Media University, Nobelstraße 10, 70569 Stuttgart, Germany
{smirek,henka}@hdm-stuttgart.de, gzimmermann@acm.org

Abstract. An increasing number of devices and applications from the Smart Home and Ambient Assisted Living domain are leaving the experimental state and are reaching commercial viability. These developments come with great opportunities, but also with challenges for elderly and disabled people. In this paper, we propose a holistic approach, using concepts of the Global Public Infrastructure (GPII), the Universal Remote Console (URC) and the upcoming technology of Web Components, to build personalized and adaptive user interfaces for people with special needs. The goal is to provide for everyone the interface fitting best his or her needs. In this paper we present the preliminary result of our approach and discuss its impact on the design of adaptive user interfaces.

Keywords: Human computer interaction · Accessibility · Elderly users · GPII · URC · AAL · Smart home · Web components · Adaptive user interfaces

1 Introduction

We face a world with an increasing presence of information and communication technologies (ICT) in everyday life situations. ICT products are available in public spaces, at work and in our homes. Also the technical interconnections and interdependencies of ICT products are continuously increasing. Consequently, we are approaching a world of so-called smart environments. This trend yields a high potential to support us in our daily life and to make it more comfortable.

This is true for "regular" users but also for people with special needs like people with disabilities or elderly people. Several studies [1–3] have illustrated the potential of providing smart environments for elderly people and that elderly are willing to use them as a support for retaining an independent life. What also needs to be considered in this scenario, is that ICT products are not only getting richer in functionality, but also in complexity [4] involved with new challenges for the design of user interfaces [5]. Different users have different needs, preferences and requirements, regarding the usage of an ICT product. This is of special importance when considering elderly people, where accessibility issues can change over time and the skill level of interacting with ICT products varies broadly [6]. One approach of providing user interfaces to a

J. Zhou and G. Salvendy (Eds.): ITAP 2015, Part II, LNCS 9194, pp. 465–475, 2015.
DOI: 10.1007/978-3-319-20913-5_43

heterogeneous user group, with changing requirements, are so called *adaptive user interfaces* [7], which accommodate the needs and preferences of a distinct user.

However, it must be taken into account that the smart home market is still an emerging and consequently a very volatile market with a lot of different technologies. As pointed out in [8], it is very important that the choice of a certain adaptation engine does not restrict the users' opportunities to connect to multiple smart home technologies rather than to a single one. Instead, an appropriate adaptive user interface platform for smart homes and Ambient Assisted Living (AAL) applications must be able to provide adaptive user interfaces upfront so that a heterogeneous user group can access the system, while at the same time it must be able to integrate different smart home technologies in the backend so that users can benefit from all available technologies.

In this paper, we describe an approach using concepts of the *Global Public Infrastructure* (GPII) [9], the *Universal Remote Console* (URC) [10, 11] and the upcoming technology of *Web Components* [12], to build personalized and adaptive user interfaces for people with special needs. We propose an approach to build adaptive user interfaces, using self-adapting user interface widgets on the basis of Web Components. Beside the adaptation logic, the user interface widgets will comprise mechanisms to connect to GPII and to the URC framework. The connection to GPII enables the component to get access to the users personal preference set [13] with which the component adapts its appearance to the user's preferences and personal needs. We use URC for the adaptation at design time and for connecting to different devices and services in a smart-home environment (e.g., a television, HVAC device). URC introduces an abstract user interface layer for any device or service, called a *user interface socket*. Concrete user and device specific interfaces can be build upon this socket layer taking the user's needs and contextual parameters, such as the device being used, into account.

The remainder of this paper is structured as follows. Section 2 provides an overview of related work. Section 3 describes the adaptation mechanisms of GPII and URC. Section 4 illustrates the concept of our approach. Section 5 provides a discussion on our work, displays envisioned benefits, and offers an outlook on the current status and next steps.

2 Related Work

Today, it is widely agreed that smart homes and the concept of AAL can bring great benefits for a wide spectrum of users. Mavrommati and Darzentas present an overview on HCI issues related to Ambient Intelligence [14]. Nevertheless, Saizmar and Kim claim that HCI research in smart homes is limited and biased to specific situations [15]. Abascal et al. criticize that, although many scenarios have been described in the field of ambient intelligence, the interface between the user and the system still remains unclear [16]. In order to improve acceptance of Ambient Intelligence and to make it capable to provide better life quality in a non-obtrusive way, Casas et al. point out the necessity to combine ongoing Ambient Intelligence technological developments with user-centered design techniques [17].

In the same vein, Mavrommati and Darzentas point to the necessity of focusing on a more user centered HCI perspective [14]. Studies have shown that elderly people are willing to use Smart Home technologies for the purpose of a longer independent life [1, 18]. It is acknowledged that Ambient Assisted Living technologies have the potential of providing safe environments for elderly people [2].

Nevertheless, at the moment, many technologies do not yet meet the needs of elderly people and current solutions overemphasize the importance of smart devices while either neglecting or lacking real implementations on the side of human interaction and human power [17]. Therefore, several authors have argued for a more user-centered view in the Ambient Assisted Living domain [17, 19, 20].

Kleinberger et al. [21] and Abascal et al. [17] are concerned with the design of appropriate user interfaces in the field of Ambient Assisted Living. Their conclusion is that natural and adaptive interfaces can bring great benefits to this field.

The PIAPNE Environment [16] is an adaptive Ambient Assisted Living system for elderly people based on three models: A user model (capabilities, permissions), a task model (user activity) and a context (environment) model. The system consists of multiple layers, including a middleware layer to bridge different network technologies and an intelligent service layer to which intelligent applications (interfaces) can be connected.

The MyUI project [22] provides a framework for self-adaptive user interfaces. The project follows the approach that user interface developers create an abstract application interaction model that is rendered at runtime according to a user profile and environmental conditions. It uses an interactive TV set as a communication point. MyUI allows only controlling one device at the time, with a dedicated user interface. Responding to multiple devices in a single user interface in order to execute scenarios in a smart home environment is not within the project's scope. A scenario that falls under this condition would be dimming the lights, switching on TV and blue-ray player with one command; hence, enabling “cinema mode”.

3 Providing a Suitable User Interface for Everyone

In 2010, Sloan et al. illustrated in their work the potential and benefits of adaptive web user interfaces for the elderly [24]. However, they pointed out the necessity for frameworks and environmental settings to accommodate this adaptation process and that an appropriate system was still missing at that time. By now, The GPII and the URC framework provide appropriate solutions to close this gap of missing foundation technologies.

3.1 The Global Public Inclusive Infrastructure

The GPII serves as a foundation to support adaptation of user interfaces across devices by transferring settings from one device or service to another. The vision of the GPII is to provide personalized, self-adaptive user interfaces to all people, including those facing accessibility barriers when using the Internet or other electronic services.

Independent of age, disability or literacy, people shall be enabled to use the full advantage of the Internet and with that having the chance to access typical features and applications of our modern world.

In order to benefit from this infrastructure users have to customize their private devices like PCs, smart phones, etc., according to their own needs. The GPII uses the settings from the user's personal device and stores them as a so-called *personal preference set*. The user's preferences contain information like the need for increased font size, a scanning keyboard, or volume settings. The preference set is stored in the cloud. From now on the user can use every device connected to the GPII. Typical examples are ticket machines, computers in public libraries or applications running on any platform. The adaptation process is conducted by transferring the user's needs and preferences from one system or application to another. The adaptation process comprises preparation at runtime as well as design time. If a user expresses the need for a larger font size, a specific color theme or magnification, this can easily be done at runtime.

However, more complex adaptations like content adaptations (e.g. the provision of sign-language videos to describe content for deaf people) need to be prepared when designing the application; since sign language videos cannot be automatically generated for the corresponding content at runtime [30].

In several studies it was illustrated that the elderly can benefit from assistive technologies and the conformance of user interfaces to accessibility guidelines [25, 26]. However, elderly users face also issues due to different perception models or strategies in the meaning making process [27–29].

As it was stated in [23], the elderly rely on familiar interaction patterns and tend to use them more frequently rather then to search for alternative interaction flows to accomplish their goals. When confronted with semantic barriers where they cannot apply their familiar interaction concepts they rather blame themselves than the application's design. Therefore, the elderly have preferences and needs in which an adaptation must be conducted on the content's semantic level [23] and as described, semantic content adaptation is hard to undertake without design time preparations. Zimmermann et al. [31] illustrated that user interfaces can be modeled as layered systems, consisting of three layers: presentation and input events, structure and grammar, and content and semantics; hence, in an attempt to provide a strongly user-centered adaption one has to consider runtime and design-time adaption [30].

3.2 The Universal Remote Console

The Universal Remote Console (URC) focuses mainly on electronic devices that can be found in smart home environments and in the AAL domain. URC provides pluggable, portable and personalized user interfaces; hence, people can control any target device or service with a controller-device and with a user interface, which best fits their needs.

In order to enable pluggable user interfaces, every target has to provide an abstract description of its user interface functionality - the *user interface socket description*, or just "*socket description*". A socket description is basically an API description of a devices' operating interface and contains information about properties that can be accessed by a user, in the form of variables (e.g., the temperature of a thermostat), commands that can be sent to the device (e.g., changing the channel on a TV) and notifications that are dispatched by the target (e.g., reminder function of a calendar). Based on these socket descriptions, one can either develop personalized user interfaces for different user groups or additional resources for existing user interfaces.

User interface resources are associated with dedicated socket elements and can be any user interface component, e.g. supplemental labels for multi-language support, additional help texts or instructions to sign language videos. User interfaces and user interface resources are stored on a resource server and are downloaded on demand at runtime. In a usage scenario, a user connects their controller device, e.g., a smartphone running a URC client, via the URC system to a target. Based on their specific controller device, a list of appropriate user interfaces is then presented for the user to choose from. The chosen user interface is automatically downloaded from the resource server and virtually plugged into the socket exposed by the target.

Targets of the same class (like TVs) could all expose a basic common socket, containing common functionality. A person can therefore exchange their device while retaining its familiar user interface. This usage of personalized user interfaces can also be seen as an asset for the elderly, since they can continue to use their well-known user interfaces, without being afraid of using new technologies; therefore, pluggable user interfaces can accommodate different perception models or meaning making strategies.

4 Concept of the Envisioned Approach

We propose an approach to provide self-adapting user interface components, so-called widgets. Each widget is intended to execute a certain task, like one for the log-in to a device, or another one representing a simple switch. All widgets together form a set of building blocks that can be used to create more complex user interfaces. We decided to use the upcoming technology of *Web Components* [32] for implementing the different building blocks, in order to be as platform independent as possible and to be able to define independent components with clear interfaces.

Web Components provide the potential to define and use and arbitrary HTML elements that extend the element space of HTML. For example, one can define a “login element” consisting of two form elements for user name and password, and a button that performs the login action. Grouping these elements by using a nested element structure, a web author can better characterize the semantics of this component, using only one sophisticated HTML element. This new element internally consists of the same three elements (two form elements and a button), but those are hidden inside the widget. Web Components are an umbrella term for three concepts*: Custom Elements* [34], *Shadow DOM* [33] and *HTML Import* [35].

Custom Elements is an API that allows defining and registering arbitrary HTML elements in the web browser. Hereby, web authors can define their own library of HTML elements.

Shadow DOM characterizes the internal DOM tree of HTML elements. Complex elements make use of internal elements, e.g., to form the control components like a “play”, “fast-forward” or “mute” button in the HTML5 video element. So far, only the web browser was permitted to manipulate the internal structure of HTML elements. By the Shadow DOM specification a developer gains access to this internal DOM tree. The API of the Shadow DOM can be used to hide and manipulate the implementation details of arbitrary HTML elements, i.e., the Shadow DOM represents the structure and appearance of a Custom Element and is exposed to a user.

HTML Imports is a mechanism that allows importing HTML documents or individual elements from different documents during runtime. HTML Imports provide the foundation to build libraries of arbitrary HTML elements, which can hereby be reused in multiple HTML documents. The import of the HTML elements can also be carried out from remote sources. We can utilize this to store HTML templates adjusted for dedicated user groups and devices on a server like the URC's resource server and download the appropriate templates during runtime.

One of the Web Component's benefits is their web-based foundation; thus, their platform independence and runtime adaptability. Assistive technologies such as screen readers interact with web pages as the browser renders them; therefore, Web Components and elements in the Shadow DOM are accessed just as any other element and their accessibility depends equally on accessible design and its conformance to accessibility guidelines.

As stated in our previous work, URC and the GPII can be an asset for providing adaptable user interfaces in smart home environments [30, 36]. Web Components can be a connector that accommodates the need for runtime adaptation, platform independence and the connection to URC sockets via common web communication methods. Also, they allow for reloading of resources to conduct deeper adaptations if the options for simple adaptations (e.g., increasing font-size) are exhausted. Figure 1 illustrates the interplay of the involved technologies.

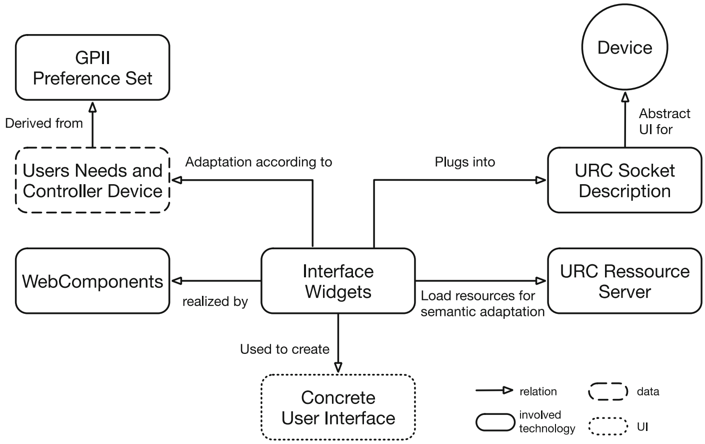

Fig. 1. Overview of the concept and the interplay of the involved technologies

The widgets should be defined using Custom Elements and structured to map the URC socket elements. Therefore, widgets should not contain any specific description of their appearances on a user interface in their names. Instead they should be described by functionality. So, instead of having a "list" element, we would have a "select-one-of-many" element. The concrete appearance depends on the used controller device and the specific user needs and preferences derived from the GPII. In order to accommodate these requirements, each widget must provide the following requirements:

- Some internal adaptation logic to shape its representation. The principle is to adapt the widgets by the means of the user's personal preferences. Simpler adjustments, like changing font-size or color theme, can be directly conducted by the widgets by using common web techniques such as JavaScript and CSS. Also, some more complex adaptations like substituting list-menus for radio buttons can be done by the widget itself, using JavaScript.
- Some appropriate logic to connect to one or several socket elements, so that a target can be controlled. To connect a URC socket to a specific widget, we can set an attribute on the widget that point to the specific element in the socket.
- Procedures enabling the connection to the GPII infrastructure to get access to a user's preference set to perform required adaptations, or – if the widget cannot perform the required adjustments – to download a new and suitable widget appearance from the URC resource server, based on the users preferences and needs, as proposed in [30].

To compensate for insufficient content adaptations, as described in Sect. 3, we propose to use the URC resource server to store alternative widget appearances. If the accommodation of the user needs requires semantic adaptions, e.g., in case of simple interfaces or the usage of sign-language videos, alternative widget appearance versions can be downloaded during runtime and substituted in the widget's Shadow DOM. A user now sees the exchanged and therefore adapted version of the widget. In order to function as a valid substitution, the alternative widget appearances have to satisfy the same interaction paradigm as the original one. If a certain socket element is augmented with a widget using checkboxes to express the functionality of switching on and off a light, an alternative widget appearance has to augment the same functionality.

The example in Listing 1 illustrates our proposed widgets, using an example of augmenting a socket element for light control. The identifier in the attribute "socket-element" specifies the augmented socket element. Switching the light on or off is a boolean operation, since the light can either be on or off. Therefore, the socket can only be augmented by user interface components and interaction pattern that express this behavior (e.g., checkboxes, radio buttons and rocker-switch shaped pictograms that are showing the state of the light in a graphical way, as shown in Fig. 2).

Listing 1: Illustration of the widget usage in an application.

```
<html>
   <head>[...]</head>
   <body>
     <h1>Light Control</h1>
     <boolean-switch socket-element=http://lightController/OnOFF/>
   </body>
</html>
```

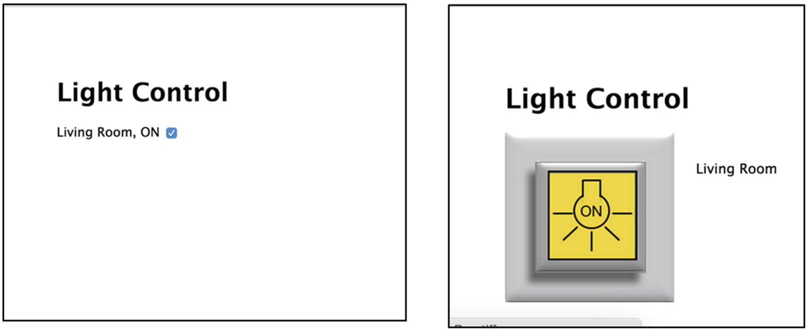

Fig. 2. Examples for the appearance of a widget, which is augmenting the on/off functionality of a socket to switch lights.

The left interface in Fig. 2 shows a checkbox as a very basic user interface. The interface on the right illustrates a more descriptive user interface by using a graphic imitation of a rocker-switch. However, they both rely on the very same widget (cf. Listing 1) with only the appearance in the Shadow DOM exchanged.

5 Discussion and Conclusion

The integration of URC and GPII in predefined widgets provides several benefits for users and developers. Users can profit from user interfaces that adapt to their individual needs and also include semantic adaptions by substituting the appearance of a widget to provide the individually preferred interaction pattern; furthermore, the concept of sockets makes it possible to exchange targets while still providing a familiar user interface.

Due to the set of predefined widgets developers have a base for developing appropriate interfaces for different user groups. Because of the widgets' internal adaptation mechanisms, they do not need to address a different user interface for every user group.

The combination of URC and GPII results in a stronger adaptation mechanisms. URC provides only the possibility to build personalized user interfaces or exchange some parts of it, but does not provide any adaptation mechanisms or user preference sets, like GPII does.

On the other hand, GPII can benefit from the inclusion of the URC to allow changes at runtime. So far, in the GPII, all required sources for the adaptation must be available before runtime, which leads to a rather closed system. Here, the URC resource server can bring additional value to the system by making additional resources available at runtime. While many resources can be prepared at design time, there are some cases in which the need for additional resources occurs at runtime.

A further advantage of this system is the provision of additional resources by third parties, e.g., assistive technology experts can provide sign-language videos for specific user groups and sockets. When uploaded to a resource server, they become available for user interfaces. But third-party contributions bears also security risks such as injecting malware into the system. In order to cope with security problems, one could think about a review process for resources, like in an app store for mobile applications. So far, such techniques are not yet available and also GPII security framework is under development.

Another issue is the acceptance of adaptive user interfaces by developers. By using self-adaptive user interfaces that adapts on the clients' side, the influence on the final appearance of the user interface is shifted from the developer to the renderer; which can break the design and function of user interfaces.

By using widgets, developers still have at least some freedom of choice how certain elements are rendered, located and behave on the final user interface and with that the acceptance for our approach increases.

The main tasks to be accomplished in the future are to provide an appropriate security framework and to widen the set of available widgets; furthermore one could think about a totally automated user interface generation process by parsing a URC socket description and, based on the result, choosing appropriate widgets.

Acknowledgments. 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"). 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. Demiris, G., et al.: Older adults' attitudes towards and perceptions of smart home technologies: a pilot study. Informatics for Health and Social Care **29**, 87–94 (2004). [Hrsg.] Informa UK Ltd. UK
2. Sun, H., et al.: Promises and challenges of ambient assisted living systems. In: Sixth International Conference on Information Technology: New Generations, 2009, ITNG 2009, pp. 1201–1207. IEEE (2009)
3. Den Ouden, E.: Development of a design analysis model for consumer complaints: Revealing a new class of quality failures. Eindhoven University of Technology, Eindhoven (2006)
4. Richter, K., Hellenschmidt, M.: Interacting with the ambience: Multimodal interaction and ambient intelligence. Interaction **19**, 20 (2004)
5. Rich, C.: Building Task-Based User Interfaces With ANSI/CEA-2018. Computer **42**(8), 20–27 (2009)
6. Abou-Zahra, S., Brewer, J., Arch, A.: Towards bridging the accessibility needs of people with disabilities and the ageing. In: Proceedings of the 2008 International Cross-disciplinary Conference on Web Accessibility (W4A), pp. 83–86 (2008)
7. Peissner, M., Schuller, A., Ziegler, D., Knecht, C., Zimmermann, G.: Requirements for the successful market adoption of adaptive user interfaces for accessibility. In: Stephanidis, C., Antona, M. (eds.) UAHCI 2014, Part IV. LNCS, vol. 8516, pp. 431–442. Springer, Heidelberg (2014)
8. Maestre, J.M., Camacho, E.F.: Smart home interoperability: the DomoEsi project approach. Int. J. Smart Home **3**(3), 31–44 (2009)

9. Global Public Inclusive Infrastructure, Jan 2015. http://gpii.net/
10. OpenURC Aliance. OpenURC, Feb 2015. http://www.openurc.org
11. ISO/IEC 24752 Information Technology user interfaces Universal Remote Console 5 parts (2008)
12. W3C, Introduction to Web Components, 07 Nov 2014. http://www.w3.org/TR/2013/WD-components-intro-20130606/
13. Iglesias-Pérez, A., Loitsch, C., Kaklanis, N., Votis, K., Stiegler, A., Kalogirou, K., Serra-Autonell, G., Tzovaras, D., Weber, G.: Accessibility through preferences: context-aware recommender of settings. In: Stephanidis, C., Antona, M. (eds.) UAHCI 2014, Part I. LNCS, vol. 8513, pp. 224–235. Springer, Heidelberg (2014)
14. Mavrommati, I., Darzentas, J.: An overview of AmI from a User Centered Design perspective. In: 2nd IET International Conference on IET, vol. 2, pp. 81–88 (2006)
15. Saizmaa, T., Kim, H.-C.: A holistic understanding of HCI perspectives on smart home. In: Networked Computing and Advanced Information Management, vol. 2, pp. 59–65 (2008)
16. Abascal, J., Fernández de Castro, I., Lafuente, A.L., Cia, J.M.: Adaptive interfaces for supportive ambient intelligence environments. In: Miesenberger, K., Klaus, J., Zagler, W.L., Karshmer, A.I. (eds.) ICCHP 2008. LNCS, vol. 5105, pp. 30–37. Springer, Heidelberg (2008)
17. Casas, R., Blasco Marín, R., Robinet, A., Delgado, A.R., Yarza, A.R., McGinn, J., Picking, R., Grout, V.: User modelling in ambient intelligence for elderly and disabled people. In: Miesenberger, K., Klaus, J., Zagler, W.L., Karshmer, A.I. (eds.) ICCHP 2008. LNCS, vol. 5105, pp. 114–122. Springer, Heidelberg (2008)
18. Demiris, G., Oliver, D., Dickey, G., Skubic, M., Rantz, M.: Findings from a participatory evaluation of a smart home application for older adults. Technol. Health Care **8**(2), 111–118 (2008)
19. Henkemans, O.B., Caine, K., Rogers, W., Fist, A.: Monitoring for independent living: user-centered design of smart home technologies for older adults. In: Proceedings of the Med-e-Tel Conference for eHealth, Telemedicine and Health Information and Communication Technologies, pp. 368–373 (2007)
20. Röcker, C.: User-centered design of intelligent environments: requirements for designing successful ambient assisted living systems. In: Proceedings of the Central European Conference of Information and Intelligent Systems (CECIIS 2013), pp. 4–11 (2013)
21. Kleinberger, T., Becker, M., Ras, E., Holzinger, A., Müller, P.: Ambient intelligence in assisted living: enable elderly people to handle future interfaces. In: Stephanidis, C. (ed.) UAHCI 2007 (Part II). LNCS, vol. 4555, pp. 103–112. Springer, Heidelberg (2007)
22. MyUI Consortium. MyUI project Official Web Page, Jan 2015. http://www.myui.eu/
23. Pernice, K., Estes, J., Nielsen, J.: Senior Citizens on the Web, 2nd Edition | Research Report by Nielsen Norman Group. Nielsen Norman Group (2013)
24. Sloan D., Atkinson, M.T., Machin, C., Li, Y.: The potential of adaptive interfaces as an accessibility aid for older web users. In: Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A), New York, NY, USA, pp. 35:1–35:10 (2010)
25. Affonso de Lara, S.M., Watanabe, W.M., dos Santos, E.P.B., Fortes, R.P.M.: Impro-ving WCAG for elderly web accessibility. In: Proceedings of the 28th ACM International Conference on Design of Communication, New York, NY, USA, pp. 175–182 (2010)
26. Dickinson, A., Arnott, J., Prior, S.: Methods for human – computer interaction research with older people. Behav. Inf. Technol. **26**(4), 343–352 (2007)

27. Abou-Zahra, S., Brewer, J., Arch, A.: Towards bridging the accessibility needs of people with disabilities and the ageing community. In: Proceedings of the 2008 International Cross-Disciplinary Conference on Web Accessibility (W4A), New York, NY, USA, pp. 83–86 (2008)
28. Arch, A.: Web accessibility for older users: successes and opportunities (Keynote). In: Proceedings of the 2009 International Cross-Disciplinary Conference on Web Accessibililty (W4A), New York, NY, USA, pp. 1–6 (2009)
29. Leitner, M., Subasi, Ö., Höller, N., Geven, A., Tscheligi, M.: User requirement analysis for a railway ticketing portal with Emphasis on Semantic Accessibility for older users. In: Proceedings of the 2009 International Cross-Disciplinary Conference on Web Accessibililty (W4A), New York, NY, USA, pp. 114–122 (2009)
30. Zimmermann, G., Henka, A., Strobbe, C., Mack, S., Landmesser, A.: Towards context-driven user interfaces in smart homes - the Cloud4all project's smart house demo. Presented at the CENTRIC 2013, Venice, Italy, pp. 98–103 (2013)
31. Zimmermann, G., Vanderheiden, G.C., Strobbe, C.: Towards deep adaptivity – a framework for the development of fully context-sensitive user interfaces. In: Stephanidis, C., Antona, M. (eds.) UAHCI 2014, Part I. LNCS, vol. 8513, pp. 299–310. Springer, Heidelberg (2014)
32. World Wide Web Consortium (W3C). WebComponents, Jan 2015. http://www.w3.org/wiki/WebComponents/
33. World Wide Web Consortium (W3C, 2015a). Shadow DOM, W3C Editor's Draft, 23 January 2015. http://w3c.github.io/webcomponents/spec/shadow/
34. World Wide Web Consortium (W3C, 2015b). Custom Elements, W3C Editor's Draft, 25 January 2015. http://w3c.github.io/webcomponents/spec/custom/
35. World Wide Web Consortium (W3C, 2015c). HTML Imports, W3C Editor's Draft, 23 January 2015. http://w3c.github.io/webcomponents/spec/imports/
36. Smirek, L., Zimmermann, G., Ziegler, D.: Towards universally usable smart homes - how can MyUI, URC and openHAB contribute to an adaptive user interface platform? Presented at the IARIA Conference, Nice, France, pp. 29–38 (2014)