Interactive Personalization of Ambient Assisted Living Environments

Alexander Marinc, Carsten Stocklöw, Anreas Braun, Carsten Limberger, Cristian Hofmann and Arjan Kuijper

Fraunhofer Institute for Computer Graphics Research,
Fraunhoferstr. 5, 64283 Darmstadt, Germany
{alexander.marinc,carsten.stockloew,andreas.braun,carsten.limberger,
cristian.hofmann,arjan.kuijper}@igd.fraunhofer.de

Abstract. Ambient Assisted Living (AAL) comprises methods, systems, and services applied to improve the quality of daily life for humans, especially elderly people. Recent research emphasizes the implementation of comprehensive AAL platforms which control all technological components included in the entire environment such as one's apartment. The behavior of the system is often determined by a specific set of rules. Thus, personalization according to the person's needs and preferences includes a configuration of the given rule system. Assuming that configuration is not only conducted by technical staff but also by the person him or herself, this process can be regarded as complex, requiring technical knowledge. In this work, we present an interactive and architectural approach to support at the personalization of an AAL system by different types of users.

Keywords: Ambient Assisted Living, End User Configuration, Personalization

1 Introduction

Ambient Assisted Living (AAL) comprises methods, technological systems, products, and attendances applied to improve the quality of daily life for humans in different periods of life. Considering predictions of the demographic changes in society, AAL particularly focuses on elderly people. Especially, the integration of technology is user-centered, that is, it is adapted according to the specific needs of assisted persons and unobtrusively embedded into the environment. Recent research directions emphasize the implementation of comprehensive AAL platforms which manage and control all technological components included in an entire environment such as ones apartment. Among these technological components are i/o-devices like interactive TV, RFID tags or capacitive sensors for localization and activity recognition, or actuators such as robotics.

Characteristical AAL platforms bear specific challenges referring to their personalization, regarded as the configuration and modification of the rules which determine the behavior of the environment according the specific needs and preferences of the assisted person. First of all, such challenges arise from the diversity of heterogeneous components and information the platform needs to handle, e.g., computing unit(s), devices, software components, services, or different participants. Second, such environments are usually driven by a specific set of rules which finally form the application tier. Here, rule-based applications define the overall behavior of the system and determine the state of different elements potentially resulting from specific predefined events. As a result, personalization processes require technical knowledge and skills such as specific notations or programming languages, which are normally not among the standard competencies of regular users, and even less of assisted persons. In this context, we must consider various types of users which differ according to their technical knowledge, skills and expertise on the one hand, and their physical capabilities or disabilities [7]: *expert*, *normal*, and *impaired* user. While the first type can particularly be associated with technical staff, normal and impaired users refer to the assisted persons. Here, we regard impaired users as persons which have physical and mental disabilities.

In this paper, we present a new approach for the interactive configuration of comprehensive Ambient Assisted Living Environments on the level of authoring tools, focusing on the application of AAL at home. First, essential requirements are discussed which refer to the given AAL-system and the different types of users. In Section 3, configuration parameters are elucidated, explaining what needs to be configured regarding objects of the AAL space as well as the given system of rules. As central contribution of this work, in Section 4 we present an interactive concept to support different types of users during the personalization process. Here, we specify different methods for multimodal interaction and abstraction of information and access to the system. In Section 5, we introduce architectural components which realize the established concepts, and integrate them into an existing AAL platform developed in the scope of the *UniversAAL*[1] project. Concluding, we summarize the presented results and describe our future work.

2 Requirements

In this section, first we give an overview of the general requirements which have to be fulfilled by a platform to allow an efficient personalization of a system. Then we present requirements that are of interest for the different user-types such as defined in Section 1.

2.1 General considerations

To allow an intuitive and easy interaction, a technical system needs to fulfill a minimal set of requirements regarding the hard- and software structure. According to our research focus, these requirements generally depend on a) the number of users that are using the system and b) the users' knowledge and skills.

Handling multi-user-systems is a part of current research. The limitations in this case are not primarily given by the software but by sensor systems which track and distinguish between different persons within the same living area. Recent approaches include the equipment of every user with small active sensors or passive tags that can be detected by ultrasound or radio signals, or vision based methods with time-of-flight or video cameras. However, these methods require the user to either wear sensors/tags or suffer from user acceptance in case of cameras. Due to these limitations and assuming that a majority of the elderly people is living alone at home [3] we will handle in this work basically one-user-systems.

For a user interface, considering the user's knowledge and skills is an important aspect. The requirements regarding the categories of users are given in Section 2.3

2.2 Basic requirements

From a high-level point of view, there is a number of requirements that a platform needs to fulfill to enable a system to be configurable by a user. For a detailed description, the reader is referred to [8].

R1 - Hardware abstraction layer: A platform for personalization needs to abstract from the hardware in a way a user can query all available devices, their functionality and benefit in the overall system, as well as their logical connection.

R2 - Interaction framework: The platform needs to support a framework for managing the interaction between human and machine. This includes the processing of multimodal in- and outputs that enables a user to interact in a way that is most natural to him/her and fits his/her skills and physical needs.

R3 - Rule based system: To realize this, we use a rule-based system to describe behavior in a way that it can be understood and saved by the system, and edited and parameterized by the user to fit it to his needs.

R4 - Service based infrastructure: Services are an adequate choice to create an open system. This allows an understandable description, abstraction, and connection of different distributed functionalities.

R5 - Context reasoning: The intended behavior strongly depend on the current state of the system and the context of the user (e.g. the user is now in the kitchen). Therefore a system is needed that allows to save, read and change the current context as well as making it transparent to the user.

R6 - Semantic descriptions: Semantic descriptions enables discovering and processing of services, and providing contextual data in a way that end users can work with it.

2.3 Requirements for different kind of users

In the following, the requirements for the different user types are defined. These are requirements for the tools that realize personalization on top of the requirements described in the previous section. Some requirements are relevant for all kinds of users, but we assign requirement to users where it is most appropriate.

Requirements for expert users: An expert user has profound knowledge about the system and its configuration. Thus, the rules and parameters could even be shown directly, only small abstraction can be considered helpful. However, the following additional requirements should be fulfilled:

R7 - Direct interfaces: Since an expert user is able to program even complex rules by himself he needs to have direct access to all relevant components.

R8 - System details: An expert user will perform personalization as a daily routine and has detailed knowledge about the technical background. Thus, all details must be available. Rules and parameters can be edited "by hand".

R9 - Direct Feedback: All feedback from the system has to be shown directly to the user without modification or delay.

Requirements for regular users: A regular user is typically not interested in all details about the used technologies of the platform, but he is well trained in using common interaction devices. This kind of user is hard to deal with, because he knows what is possible, but not how to realize it. A personalization tool for a regular user therefore needs to hide technical aspects, but in best case offer the same functionality as for the expert user. The following additional requirements are defined:

R10 - Common user interfaces: For regular users tools for personalization must make use of different interaction techniques with the system. Since they are not interested in details, other modalities like speech are very relevant here to gain a bigger benefit in usage.

R11 - Help-files: A regular user is able to adapt new concepts based on common knowledge. Technical details should be hidden, but hints about them have to be available in tutorials and help-files.

R12 - Up-to-date look and feel: For the acceptance of a system it is particularly for the regular users important to have a modern look and feel of the visual components.

Requirements for impaired users: Impaired users need to be assisted much more by the tools than other users. They are typically not able to use the full functionality given by the system and accordingly they will be offered only basic elements of configuration. An interesting aspect is the automatic learning of the user's behavior to try to estimate his/her current needs. This way, it will be possible to guide him/her to use the system more use- and trustfully. For the impaired user we define the additional requirements in the following:

R13 - Limited access to details: Complex and vital details of the system have to be hidden. This way, the user does not get overwhelmed by the user interface or even trapped in nested dialogs. Additionally, some parameters like emergency processing can not be made inoperative by the user.

R14 - Simulation: Offer the possibility to simulate recently created rules. This means, testing without changing anything in the real system. This can take the fear to "play"with the system.

R15 - Self learning: In particular for the group of impaired users it can be helpful if not only the user has the active role. If the system detects common patterns in the behavior of the user (e.g. switch on TV every morning at 8am), it should suggest to the user the permanent acquisition of automatically created rules.

R16 - System interruption: Impaired users have in most cases no idea how the system is working, therefore it is important that every behavior introduced by the system can be interrupted by the user as easy as possible. This way, it is ensured that a user does not feel patronized by the system.

3 Configuration Parameters

In this section, we provide a more detailed look on the technical elements and properties which are to be personalized. As already stated in Section 2, we assume to have a given Service-Oriented Architecture (Requirement 4). The very basic elements are Service-Descriptions, Service-Calls, and appropriate Service-Responses. In addition to this, we need to have a sort of Reasoning Managing System which is able to create, distribute and save changes in the context of the user (context-events). Finally, the platform must support input and output channels which allow a user to interact with the system (see also Section 5).

3.1 Configuration of single elements

One essential object is an editable user-profile which contains general information such as the person's name or physical limitations. The other basic element in the system are the Service-Descriptions that have to be given in same description language. A common way is to use the XML based language WSDL [1], but also semantic expressions can be used, e.g., provided by OWL [2]. In order to make all services available to the user, we have to ensure that enough meta-information is included in order to distinguish between different user types. Additionally, this information must be editable in order to adapt the descriptions according to the individual user's needs (e.g. for a description of a TV "X8300"vs. "TV Living Room"). The needed inputs for Service-Requests can be retrieved by simple questionnaires and the presentation of the Service-Responses strongly depends on the user-profile (e.g. using loudspeakers if the user is blind). The current context of the user (except of his profile) does not have to be configured by the person himself. We must assume that the platform provides an appropriate overview about the available context-events. Furthermore, a prerequisite is that programmers defines suitable names. This gives a user the freedom to include event-reasoning in his personalized rules. Output devices can be included as services and Input-Events are represented as events that provide input for Service-Calls.

[1] Web Services Description Language: XML-based language for the Web Services
[2] Web Ontology Language: RDF/XML-based language for the Semantic Web

3.2 Configuration of the dependencies between elements

Besides the configuration of single elements, the main part of personalization is to take control over the system by creating rules that are usable by the platform, combining given single elements. Combining means in the simplest case just to group services or events. Multiple services are called at the same time and will be executed after all required inputs of all services in the group are given by the user. Examples for sophisticated methods such as entire workflows between services. This offers the possibility to define restrictions in the process (e.g., take service X if Y failed and Z otherwise). Since many interactions are complex and multimodal it most be also possible to combine events (e.g. combine "switch lamp on"and "user is in living room"to create the event "switch lamp in living room on") for futher use. The most important aspect is the combination of services and events. This is what offers the possibility to directly control the behavior of the system. If a (maybe combined) event like an explicit command is triggered, an associated service (this again can consist of many services) should be triggered.

3.3 Representation of rules

All rules and combined services have to be represented in an appropriate way in the system. To realize this, different forms of representation have been researched in the past, e.g. for the definition of workflows the description languages BPEL[3] or XPDL[4] can be used. The platform itself is not bound to a specific representation; there can be multiple components responsible for storing, executing and processing the various types of description languages. In this case, it must be possible to access and configure the rules contained in these components to provide the possibility to combine them into higher level rules, and to provide appropriate interfaces to present them to the user.

4 Interaction

In this section, we present an interaction concept which supports the process of customizing the system of rules of an AAL-system. The main idea can be exemplified with the simple scenario "evening activity". *After dinner, the assisted person usually reads a book in the living room. For this purpose, when he or she leaves the dining room the lights are automatically switched off. In the living room, the ceiling lightings are dimmed and a reading lamp positioned next to the couch is switched on. In the course of time, the person's ability to see has decreased. As a consequence, instead of reading, the person is going to listen to the local radio program. Hence, the reading lamp remains off, and the hifi-system is turned on with the preset radio channel.* Here, the underlying rules system needs to be modified. As already mentioned, these changes can be conducted

[3] Business Process Execution Language

[4] XML Process Definition Language

by different kinds of users with different knowledge and skills. In order to take this aspect into consideration, we enable a multimodal communication with the system on the one hand, and define varying abstraction levels on the other hand.

4.1 Multimodal interaction

Going back to the "evening activity" scenario: assuming that the person's ability to see has decreased, this not only results in a change of the activity, but has also an impact on the way a respective modification is conducted. In this example, a "traditional" interaction by means of a *Graphical* User Interface (GUI) would certainly be the wrong choice. A better solution would be to allow a more intuitive and accustomed interaction, for instance, by pointing at the reading lamp and expressing the command "switch off" in a verbal way. If the person repeats this on several evenings, the system would change the related rule after a confirmation request. Thus, we introduce three different modalities which can be combined in the scope of one single interaction, *GUI-based*, *speech-based*, and *gesture-based* (see Fig. 1).

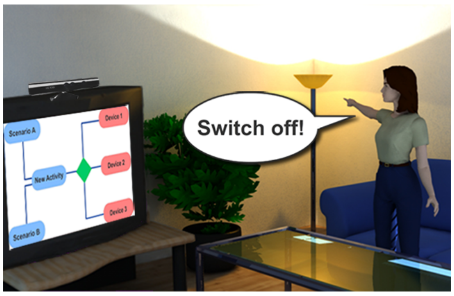

Fig. 1. Multimodal personalization by means of a GUI, speech, and pointing.

GUI-based. Interaction based on a GUI requires an appropriate visualization of the given rules system. Thus, all building blocks of the system have to be graphically presented to the user, supporting the creation of a correct mental representation [5]. Examples for common standard visualization models are activity graphs, workflow graphs, or semantic visualizations [4, 5]. Here, configuration is usually performed by Direct Object Manipulation [6].

Speech-based. The person can explicitly communicate with the system in a habitual verbal manner based on an *automatic speech recognition* approach (ASR) [2]. Here, spoken words are automatically converted to text and compared to a repository of predefined commands.

Gesture-based. Also regarded as habitual, gesture-based interaction is based on automatic gesture recognition. In our work, a particular focus lies on the recognition of pointing activities. These can be combined with a spoken command, but also with a graphical representation, e.g., by highlighting the graphical representative of the referenced object.

4.2 Levels of abstraction

In addition to the options to interact with the system by means of different modalities, we introduce different levels of abstraction in order to meet the needs of persons with differing technical and physical prerequisites. Abstraction implies the presentation of information as well as the access to this data in order to perform modification on the set of rules. Thus, abstraction relates to the visualized information with respect to the amount and/or the level-of-detail. Furthermore, it determines which and how many interactions are allowed out of a set of existing options. In the case of GUI-based personalization, different entities can be presented (see Section 3). For persons with high technical knowledge, the presentation can correspond to the code level, e.g., a workflow definition or an RDF[5] file. In the next level, graphical representatives can be incorporated, for example as part of an activity or workflow graph. Here, a further abstraction can be defined in order to show only basic building blocks of the rules system with a set of few basic graphical elements. With regard to verbal commands based on speech recognition, abstraction does not refer to the presented information, but to the set of commands which can be expressed by the user. That is, different vocabularies or parts of a common vocabulary are assigned to a specific user account. These vocabularies range from the explicit editing of code-level constructs such as RDF tags or SQL queries to simple commands as described in the example scenario. Since this work addresses only pointing gestures, no abstraction levels are defined in the context of gesture-based personalization.

5 Architecture

To support the concepts derived in the previous sections, a generic architecture is presented. This architecture is based on the EU project UniversAAL, as it is supposed to become a standardized general-purpose platform for AAL-spaces. UniversAAL is a consolidated combination of prior work, not following a completely new approach but rather integrating approved concepts from a variety of projects in this area. UniversAAL already has some of the components necessary to realize interactive personalization. The important parts together with some enhancements and extensions are illustrated in Fig. 2.

The middleware is the only component necessary on ever node in the network. It hides distribution and heterogeneity, and facilitates communication by providing a bus-based system with four buses. The input and output bus handle explicit interaction with the end user while context and service bus realize push and pull mechanisms, respectively, for interoperability of various components at a semantic level using ontologies and technologies like RDF and OWL[6].

The remaining components are of particular interest and are described in the following subsections.

[5] Resource Description Framework
[6] Web Ontology Language

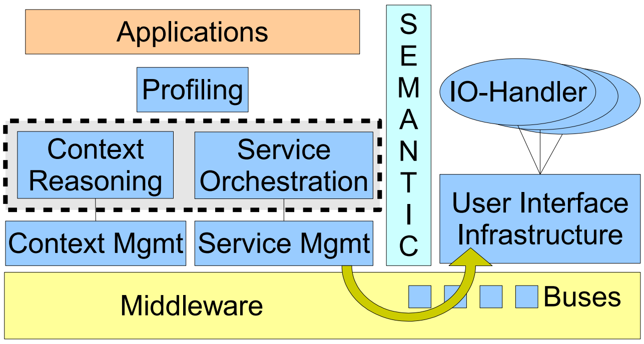

Fig. 2. Architecture of the system, orchestration and reasoning (highlighted by a dotted box) are the parts to be personalized.

5.1 Multimodal interaction

The interaction with the user is probably the most important aspect of a platform to be fulfilled for user acceptance. According to the huge variety of preferences, skills, and impairments, the platform should support multimodal interaction and integrate different modalities. The architecture supports this by integration of various so-called IO-handlers[7] to support different in- and output devices and combinations thereof for interaction with the end users. One IO-handler is responsible for a defined set of modalities and deals with modality fusion and fission. Thus, a different IO-handler can be used for the respective end user. Which handler is used, can be determined by the profiling component, that provides a user model for handling user identity, capabilities, constraints and preferences. Here, a distinction between different user types and their preferred interaction method can be realized.

5.2 Context reasoning and service orchestration

Complex services may not be resolved directly, but by intelligent strategy planning and composition of simpler services. The composed service is then available and registered at the service management component just like regular services.

The context management is responsible for all data and events that can be shared within the system based on a set of shared models and guarantees a certain level of persistence to reflect the current state and allows for querying past events when needed. This data is then used for a rule-based reasoning to derive higher-level situational data from low-level sensor events.

Service orchestration with parameterization and workflow execution as well as the rules for context reasoning are in this work the parts that can be personalized.

[7] Detailed information about the general UI Framework at a technical level can be found in [7].

6 Summary

In this paper, we present a new approach for the interactive personalization of comprehensive Ambient Assisted Living Environments. This approach includes the specification of essential system- and user-centered requirements, as well as configuration parameters which are to be modified. Moreover, we present an interactive concept to support users during the personalization process. Here, we establish methods for multimodal interaction and abstraction of information and access to the system in order to consider different types of users. Furthermore, we define architectural components which realize different aspects of the presented concepts. As a result, we expect a significant improvement regarding the understanding and the execution of personalization processes. As part of our future work, this improvement will be investigated in the scope of user studies based on a respective prototype implementation. Furthermore, additional interaction modalities will be integrated.

Acknowledgements

This work is partially financed by the European Commission under the FP7 IST Project *UniversAAL*, under grand agreement FP7-247950.

References

1. UNIVERsal open platform and reference Specification for Ambient Assisted Living. `http://www.universaal.org/`, [Online - accessed February 2011]
2. Baker, J., Deng, L., Glass, J., Khudanpur, S., hui Lee, C., Morgan, N., O'Shaughnessy, D.: Developments and directions in speech recognition and understanding, part 1 [dsp education]. Signal Processing Magazine, IEEE 26(3), 75 –80 (May 2009)
3. Meyer, S., Mollenkopf, H.: AAL in der alternden Gesellschaft - Anforderungen, Akzeptanz und Perspektiven. VDE Verlag Gmbh (2010)
4. Nazemi, K., Ullmann, T.D., Hornung, C.: Engineering user centered interaction systems for semantic visualizations. In: Proceedings of the 5th International Conference on Universal Access in Human-Computer Interaction. Addressing Diversity. Part I: Held as Part of HCI International 2009. pp. 126–134. UAHCI '09, Springer-Verlag, Berlin, Heidelberg (2009)
5. Rinderle, S., Bobrik, R., Reichert, M., Bauer, T.: Business process visualization - use cases, challenges, solutions. In: ICEIS 2006 - Proceedings of the Eighth International Conference on Enterprise Information Systems: Databases and Information Systems Integration, Paphos, Cyprus, May 23-27, 2006. pp. 204–211 (2006)
6. Shneiderman, B.: Direct manipulation for comprehensible, predictable and controllable user interfaces. In: Proceedings of the 2nd international conference on Intelligent user interfaces. pp. 33–39. IUI '97, ACM, New York, NY, USA (1997)
7. Tazari, M.R.: An open distributed framework for adaptive user interaction in ambient intelligence. In: Proceedings of the First international joint conference on Ambient intelligence. pp. 227–238. AmI'10, Springer-Verlag, Berlin, Heidelberg (2010)
8. Tazari (Ed.), M.R.: AAL Reference Architecture requirements. UniversAAL Deliverable D1.2 (2010)