Creating Rule Sets for Smart Environments through Behavior Recording

Alexander Marinc, Tim Dutz, Felix Kamieth, Maxim Djakow, and Pia Weiss

Fraunhofer Institute for Computer Graphics Research IGD, Darmstadt, Germany
{alexander.marinc,tim.dutz,felix.kamieth,maxim.djakow,
pia.weiss}@igd.fraunhofer.de

Abstract. In recent years, there has been a steady rise in the installation of smart environment systems. These systems can consist of a wide range of sensors and actuators and as such can become very complex, which brings the average user to the limits of her technical understanding. Consequently, innovative methods are required to simplify the interaction with such systems. This paper describes an approach for recording events triggered by a user and for linking those to actuator effects. Through this, even those end-users who are inexperienced with modern day technology can create custom rule sets for smart environment systems.

Keywords: Smart Environments, User Interaction, Behavior Recording.

1 Introduction

Over the last few decades, the amount of technical devices that populate our homes and work places has increased significantly. However, as of today, the majority of those devices are not capable of communicating with one another (just think of your TV and your fridge). Researchers envision that in the not-too-far future, all technical devices distributed in our surroundings will work together and form device ensembles with which we will be able to interact in a natural manner.

A multitude of challenges need to be overcome before this vision can become reality and in recent years, many research and development projects have been aimed at creating a software platform to support smart environments (e.g., [2], [3], and [4]). However, many of these research projects focus on solving the problem of interoperability, that is, on developing the means for integrating heterogeneous devices into a single system and thus enabling them to communicate with one another. And while this is obviously a key aspect of making smart environments a reality, we have noticed that oftentimes the question of how the user is supposed to control these complex systems once they have been established falls a bit short.

From the perspective of smart environments, the technical devices that they are based on can be grouped into two categories. While devices from the first category, sensors, are used by the system to perceive the state of its environment, devices from the second category, actuators, are used to influence the environment's state.

N. Streitz and C. Stephanidis (Eds.): DAPI/HCII 2013, LNCS 8028, pp. 138–143, 2013.

Frequently, the differentiation between these two categories is not clear and many devices are both, sensors and actuators (e.g., a smartphone). Usually, smart environment systems will rely on sets of control rules for the purpose of logically linking the sensor input to the actions performed by its actuators [1], meaning that in a specific situation S (as perceived by the system's sensors), a system will instruct its actuators to perform a certain set of actions A. In our work, we are proposing a new approach on how to enable users to define control rules through "demonstrating" the desired behavior to the system. The underlying principle is not entirely new, and our main contribution lies in its adaptation to the area of smart environments.

2 Related Work

Our approach was largely inspired by the Visual Basic recorder that comes with the Microsoft Office software suite. This application allows users to record his or her activities while using one of the products from the Office suite, for example Microsoft Word. Users are capable of saving these behavior patterns to shortcuts for the purpose of evoking them at a later time – a single button click will then start and automatically "replay" the whole process.

Research on the acceptance of such features goes back to Rosson's work in 1984 [7], which evaluated the advances people make when learning to work with a text-editor software. Similar macro-recording features have been used in more recent work where repetitive browsing tasks have been automated using voice-enabled macros [8]. This shows that the use of recordable macros withstood the test of time as an intuitive and user-friendly means of end-user programming.

The use of end-user programming in the context of smart environments has been suggested as a new research perspective in the field in [6]. This suggestion was then built upon in the detailed description of a concept for end-user configuration of smart environments in [1]. This concept takes into account different levels of technical expertise among end-users and presents solutions for the description as well as the recording and definition of system configurations based on multimodal user input.

3 Concept

For this work, we have developed a simple application (dubbed "the macro recorder") that provides a very similar functionality as the Visual Basic recorder described in section 2 for a smart environment system (Figure 1 shows a screenshot of the application's prototypical user interface). In a nutshell, the macro recorder application works like this: If a user wants to add a specific behavior pattern to the system, she will start the application, press the "record" button and then trigger the individual events and effects which are supposed to be part of the desired pattern. Our application, which must have access to the communication channels of the system, will then display all sensor events (e.g., movement registered by a movement sensor) and actuator actions (e.g., the activation of the kitchen lights) in order of their occurrence (more specifically: in the order of their processing by the system). Once all relevant events and actions have been recorded and the user has stopped the recording, a first version of the new control rule is automatically generated.

This preliminary rule requires the appearance of the exact same pattern of sensor events in order to trigger the same series of actuator actions. However, this may not lead to the desired effect, as it may include additional, unlooked for events and actions. As an example, imagine that the user has entered the living room (an event captured by a movement sensor) while the ambient temperature is 75.2 degrees Fahrenheit (an event captured by a temperature sensor). These two events will then comprise the event part of the new rule, and the occurrence of these events will initiate the rule's action part, for example, an activation of the ceiling lights in the living room. Obviously, in most cases one will not to want to be dependent on the ambient temperature from the rule, though, as otherwise the rule will only apply in those rare cases in which the exact same temperature is measured. For this reason, we have also provided the means for editing rules, i.e., for manually removing (and adding) sensor events and actuator actions from (and to) a given rule.

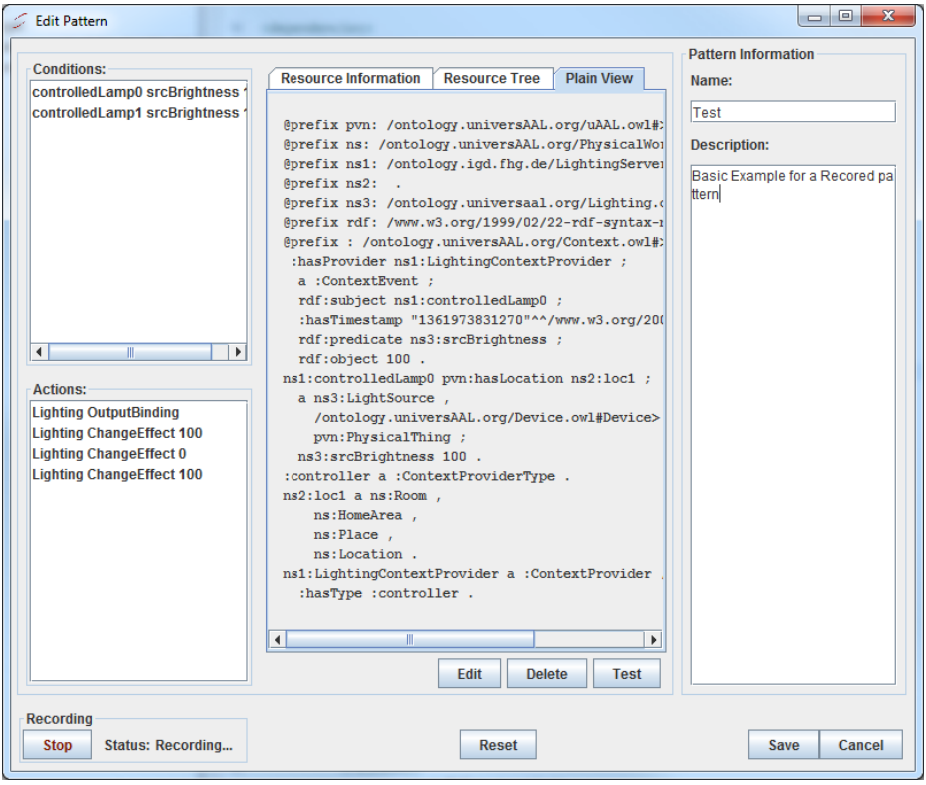

Fig. 1. Macro recorder main screen

Figure 1 shows a screenshot of the macro recorder's main screen. On the left side of the screenshot, one can see two lists, one for the occurred conditions (sensor events) and another one for the occurred actions (actuator effects). The central view shows the information stream as processed by the underlying platform, from which

.the conditions and actions are being extracted. The text boxes on the right enable the user to name and describe the current "pattern" (practically the rule currently created). Once the recording is stopped by the user, the new pattern (rule) is automatically added to the platform's rule database, but flagged as being created by the macro recorder software. This allows an easy identification of the same and its later adaptation. Figure 2 has a screenshot of the frame that shows an overview of all patterns created during this session (on the screenshot, there is only a single pattern called "Test").

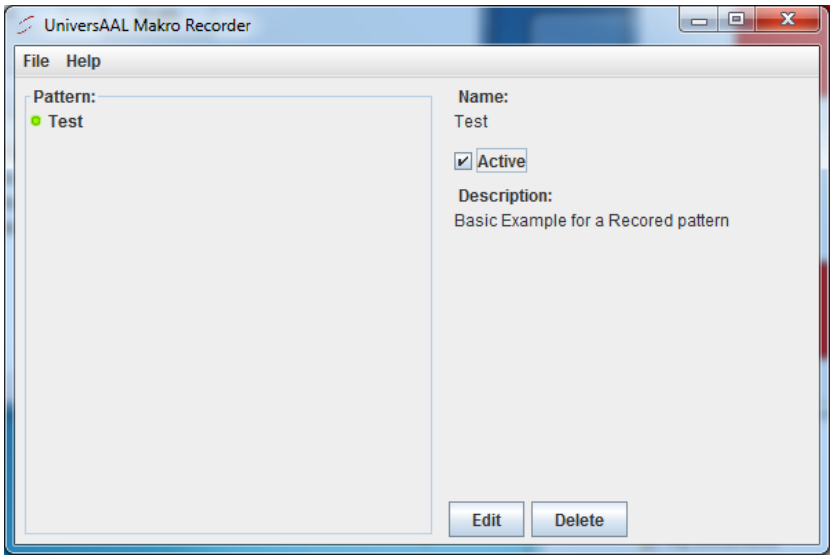

Fig. 2. Pattern overview

By selecting one of the patterns and clicking the "Edit"-button, the users is capable of editing both the "Conditions" (sensor events) and the "Actions" (actuator events) of this specific pattern. This includes the removal of specific entries, and also the possibility of moving certain events and actions upwards and downwards in the lists.

The smart environment system for which our macro recorder was developed is based on the universAAL platform [4]. This software platform allows the connection of heterogeneous devices to one another and uses multiple communication channels for the purpose of transferring information and service calls between them. The two main channels which we have been making use of are the so-called "Context Bus" for sharing contextual information between the components of the system (so-called "context events"), and the "Service Bus", whose job is the delivery of service calls ("service requests") between devices and applications.

Context events sent via the Context Bus of the universAAL platform are modeled based on the RDF/OWL format. According to this standardized format, statements are described as triples of a subject, a predicate and an object to convey information about the environment's current state. As a simple example, consider the statement "Lamp hasBrightness 100". It consists of three parts, the subject "Lamp", the predicate "ha-

sBrightness" and the object value "100". This triple describes the state of a lamp which has a brightness value of 100 percent (indicating that the lamp is "on"). Such a triple is sent through the system via the Context Bus and conveys the information that the respective lamp has been turned on (via a light switch, for example). To allow for later access, all of these context events are also saved in a central database. The reason for this is that system components which enter the system (as it is an open system facilitating dynamic interoperability between system components) can then be informed about the current state of the system by referring to past context events saved in this database. Similarly, service requests are defined using the same message structure (statements of triples based on RDF/OWL).

4 Evaluation

We have assessed the macro recorder software with a group of nine users in our institute's living lab. Three participants were computer scientists involved in the development of the underlying platform (universAAL) and as such, they were well-versed in the mechanisms and concepts that the platform is based on (understanding concepts such as context-events and service-calls). The remaining six evaluation participants had not had experience in using the platform, but provided a solid technical background (students of a lecture on Ambient Intelligence).

The evaluation consisted of a set of three test cases of rules which had to be created by the participants using the macro recorder (for example, a rule that triggered the activation of the stereo device when the movement sensor detected presence in the entrance area of our lab). Evaluation results showed that both groups found benefit in the recording of messages, but the developers still preferred the direct encoding of behavior rules using Java, especially for complex scenarios that involved many context events and service calls. However, they confirmed the hypothesis that they might consider using the recorder for the encoding of simple rules once they would have gotten used to it.

The second group of users, the students, reported positively on the fact that they were able to "program" the system, something which they had considered beyond their capabilities before the test. However, some aspects of the system were considered confusing and irritating by this group, especially the statements in the RDF/OWL format. The users explained that they would have actually preferred a different user interface which hid the exact technical details and rather only shows simple abstractions of the underlying information (such as "A light was activated in the living room").

4.1 Conclusion and Future Work

In this work, we have described our approach for creating rule sets for smart environments by recording the behavior of a user. We were able to successfully implement and test the concept with a group of nine test users, who were able to "program" the system, even if they had no further experience in working with it. However, the

evaluation showed some areas of improvement for the software. While developers of the underlying platform were comfortable with the user interface, the other users found some aspects of it too technical and offered various suggestions for its improvement.

The main point of criticism was the overall complexity and too many references to the technical concepts of the underlying platform. As such, as a future work we intend to create a more advanced version of our macro recorder software which will be capable of hiding many of the technical details. It may even be possible to provide an entirely graphical user interfaces based on graphical representations of the sensors and actuators only, which should make "programming" of the system possible for a much larger group of users.

Furthermore, as the macro recorder is currently only covering implicit interaction events (such as movement and occupation of certain areas), we intend to extend it to also handle explicit user interaction, such as voice and gesture commands. This would allow for the realization of even more complex rule patterns.

Acknowledgements. We would like to thank all evaluation participants from Fraunhofer IGD and the Technische Universitaet Darmstadt. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 247950, project *universAAL*.

References

1. Marinc, A., Stocklöw, C., Braun, A., Limberger, C., Hofmann, C., Kuijper, A.: Interactive personalization of Ambient Assisted Living environments. In: Smith, M.J., Salvendy, G. (eds.) HCII 2011, Part I. LNCS, vol. 6771, pp. 567–576. Springer, Heidelberg (2011)
2. Georgantas, N., Mokhtar, S., Bromberg, Y., Issarny, V., Kalaoja, J., Kantarovitch, J., Gerodolle, A., Mevissen, R.: The Amigo service architecture for the open networked home environment. In: Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture. IEEE Computer Society, Washington DC (2005)
3. Fides-Valero, Á., Freddi, M., Furfari, F., Tazari, M.-R.: The PERSONA framework for supporting context-awareness in open distributed systems. In: Aarts, E., Crowley, J.L., de Ruyter, B., Gerhäuser, H., Pflaum, A., Schmidt, J., Wichert, R. (eds.) AmI 2008. LNCS, vol. 5355, pp. 91–108. Springer, Heidelberg (2008)
4. universAAL - universal open platform and reference specification for Ambient Assisted Living, `http://www.universaal.org` (accessed February 28, 2013)
5. Aarts, E., de Ruyter, B.: New research perspectives on Ambient Intelligence. Journal of Ambient Intelligence and Smart Environments 1 (2009)
6. Rosson, M.: Effects of Experience on Learning, Using, and Evaluating a Text Editor. Human Factors: The Journal of the Human Factors and Ergonomics Society 26 (1984)
7. Borodin, Y.: Automation of repetitive web browsing tasks with voice-enabled macros. In: Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility. ACM, New York (2011)