NOTICE

An Agent-based Augmented Reality Demonstrator in the Domestic Energy Domain

Sebastian Ahrndt, Johannes Fähndrich, Marco Lützenberger, Andreas Rieger, and Sahin Albayrak

Abstract In this work we propose an approach for comfortable and accelerated development of user interfaces for software agents. We apply model-based techniques and emphasise the capability of this technique by describing two user interfaces which are different in nature, but have been developed with the same model. We present the applicability of both user interfaces by means of an agent-based application in the domestic energy domain. As opposed to similar approaches we retain all degrees of freedom for the applied multi-agent framework.

1 Introduction

Due to its innate consideration of distribution, autonomy and interaction, the *Agent Oriented Software Engineering* (AOSE) paradigm counters many challenges in implementing applications for the realm of ubiquitous computing (ubicomp) [6]. Yet, in our opinion, existing approaches neglect interaction between users and the software system (or, when dealing with an agent system, the interaction between users and the multi-agent system, respectively), although such consideration is inevitably required. Commonly, the agent community tries to counter the complexity of user interface (UI) development by web-based solutions [1, 8]. Nevertheless, when it comes to ubiquitous environments there are many requirements which are not easily supported by web-based approaches. Especially distribution, different device types and multi-modality are well known obstacles for web-based approaches. *Model-Based User Interface Development* (MBUID) is considered a remedy, here. The basic idea of MBUID is to formally specify a UI's appearance and behaviour by means of several models from which executable code can be derived. We already presented an approach [3] which applies model-based techniques in order to develop UIs for

Sebastian Ahrndt · Johannes Fähndrich · Marco Lützenberger · Andreas Rieger · Sahin Albayrak
DAI-Labor, Technische Universität Berlin, Ernst-Reuter-Platz 7, 10587 Berlin, Germany
e-mail: sebastian.ahrndt@dai-labor.de (Corresponding author)

software agents. The focus of this work was to retain all degrees of freedom for the applied multi-agent system. In this paper, we utilise our previous work and demonstrate the capability of merging AOSE's distributed view on systems with MBUID's superior usability. We outline an agent-application, which takes one major goal of future ubicomp users into account: A decrease in living expenses.

2 Main Purpose

As mentioned above, the development of UIs is a complex task. In ubicomp environments, this complexity increases even more due to comprehensive requirements. MBUID is considered a remedy here, as most requirements are innately supported. In addition, there are many enhancements of the MBUID technology. Interpreter-based *Model-based User Interfaces* (MBUI), for instance, are able to manipulate their models at runtime, and to dynamically adjust their appearance to the current execution context. It has been argued, that MBUI suits well for ubicomp environments [4]. In a previous work, we presented an approach that bridges the gap between AOSE and MBUID [3]. In this work we argued that the *task-model*, which is available in most MBUID environments [5], can be utilised to apply model-driven techniques for the development of UIs for agents. The task-model formalises the general workflow[1] of the application and distincts between tasks of the user and tasks that belong to the application's logic. Yet, as agents are usually compelled to a superior application goal we had to ensure that the agents were able to comprehend the tasks which have been specified for them. Further, we had to account for the transport of required data from UIs to the multi-agent system. As model-based UIs and multi-agent systems usually apply different technologies and feature different system characteristics (e.g. straight definition vs. degrees of freedom), this task became even more challenging. To solve this problem, we made use of the *Human Agent Interface* [2].

3 Demonstration

Following the spirit of ubicomp, *Ramchurn et al.* [9] presented an agent-based, decentralised demand side management for the future Smart Grid. The authors emphasise that the intelligent, autonomic control of deferrable devices[2] in the domestic energy domain is able to reduce CO_2 emissions by up to 6%. We adopt this idea for our demonstrator, where each device is controlled by an agent. The agents are able to shift the execution time of their devices to (cheaper) time slots and thus authorised to discard a user's original settings. A decision is done by considering

[1] A workflow is considered to be the tasks that can be reached.

[2] Examples of deferrable devices are washing machines, dishwashers, boilers and fridges, to name but a few.

estimated- and current energy prices, as well as the impact of shifting the device's activity on the user's comfort. The main objective of each agent is to maximise the user's comfort and to minimise the device dependent costs. As the impact on the users's comfort depends on his individual preferences and may also differ from device to device, we presume the user to be able to interact with each agent. In fact, this is an enlargement to the static values *Ramchurn et al.* [9] used. To manage the interaction between the user and the agents, we developed two entirely different UIs –an augmented reality one and a web-based one– using the same task-model. Using the interfaces, the user is able to assess the quality of the agents' decision and to override the autonomic control. The agents use the assessment to learn about the user's preferences. This mechanism is realised by reinforcement learning [10]. Figure 1 illustrates the task-model of our application and also shows both UIs.

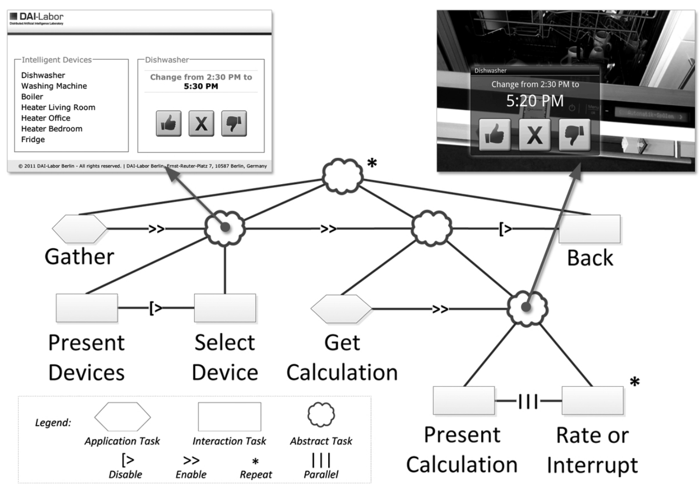

Fig. 1 The task-model of our application in the ConcurTaskTree notation [7] with two screenshots one for each UI. The web-based UI to the left present a list of all available devices and the calculation for the dishwasher. The augmented reality based UI to the right only presents the actual calculation for the dishwasher, as other devices are not in the current focus of the camera.

Once, the task-model is available, the implementation can be started simultaneous at two points: Agent-system and UI(s). The application discovers the available devices and the user is able to select the UI which is to be assessed. This is either done by marker-detection (augmented-reality) or by a list of all devices (web-based). By selecting a device, the user is able to assess the agent's decision or to interrupt the intelligent managing process. Subsequently, the user is able to select another device or to exit the application. Currently, we present this demonstrator in the showroom of our research institute. In order to demonstrate our application outside of our showroom, we use models of a heater and a washing machine, controlled by appropriated plug-pc's, a common notebook, to show the web-based UI and a smartphone for the augmented reality UI.

4 Conclusion

In this work we argue how MBUID and AOSE can be used in order to develop totally different user interface types with the exact same methodology. We clarified our approach by presenting an augmented reality- and a web-based user interface which have been developed in compliance with our approach and which were able to control the exact same agents. We selected agents from the domestic energy domain which are able to control the execution of any device with a deferrable load and demonstrated the capability of our approach to facilitate the development of device- and modality independent UIs for agent applications.
Acknowledgments: This work is partially funded by the German Ministry of Economics and Technology under the funding reference number 16KT0907.

References

1. Agent Oriented Software Pty. Ltd.: JACK Intelligent Agents – WebBot Manual. Agent Oriented Software Pty. Ltd., Victoria, Australia, 5.3 edn. (2009)
2. Ahrndt, S., Lützenberger, M., Heßler, A., Albayrak, S.: HAI – a human agent interface for JIAC. In: F. Klügl, S. Ossowski (eds.) MATES 2011, *Lecture Notes in Computer Science*, vol. 6973, pp. 149–156. Springer (2011)
3. Ahrndt, S., Roscher, D., Lützenberger, M., Rieger, A., Albayrak, S.: Applying model-based techniques to the development of uis for agent systems. In: Trends in Practical Applications of Agents and Multiagent Systems, Advances in Intelligent and Soft Computing. Springer (2012). To appear
4. Braubach, L., Pokahr, A., Moldt, D., Bartelt, A., Lamersdorf, W.: Tool-supported interpreter-based user interface architecture for ubiquitous computing. In: Forbrig, Limbourg, Urban, Vanderdonckt (eds.) Interactive Systems - Design, Specification, and Verification, pp. 89–103. Springer (2002)
5. Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J.: A unifying reference framework for multi-target user interfaces. Interacting with Computers **15**(3), 289–308 (2003)
6. Jennings, N.R., Wooldridge, M.J. (eds.): Agent Technology: Foundations, Applications and Markets, 2 edn. Springer (2010)
7. Paterno, F., Mancini, C., Meniconi, S.: Concurtasktrees: A diagrammatic notation for specifying task models. In: S. Howard, J. Hammond, G. Lindgaard (eds.) Proceedings of Interact'97, HCI Conf. Series. Chapman and Hall (1997)
8. Pokahr, A., Braubach, L.: The webbridge framework for building web-based agent applications. In: M. Dastani, A.E.F. Segrouchni, J. Leite, P.Torroni (eds.) Languages, Methodologies and Development Tools for Multi-Agent Systems, pp. 173–190. Springer (2008)
9. Ramchurn, S.D., Vytelingum, P., Rogers, A., Jennings, N.: Agent-based control for decentralised demand side management in the smart grid. In: Tumer, Yolum, Sonenberg, Stone (eds.) Proc. of the 10th Int. Conf. on Autonomous Agents and MAS, pp. 5–12 (2011)
10. Russel, S., Norvig, P. (eds.): Artificial Intelligence: A Modern Approach, 3 edn. Prentice Hall (2009)