A Multi-Agent Platform for Augmented Reality based Product-Service Systems

(Demonstration)

Nils Masuch, Tobias Küster, Johannes Fähndrich, Marco Lützenberger,
and Sahin Albayrak
DAI-Labor, Technische Universität Berlin
Ernst-Reuter-Platz 7
10587 Berlin, Germany
{firstname.lastname}@dai-labor.de

ABSTRACT

Advances in the IT-sector have shifted the conventional development of products into the direction of integrated product-service systems. Product-service systems promise a long(er) lifetime due to latest versions of (software-based) services that operate on the purchased hardware. Furthermore, they can be extremely adaptable and thus ensure connectivity and compatibility to other existing or future systems. The aim of this demo is twofold. First, the demo shows that a multi-agent approach is perfectly suited to meet the requirements for product-service systems. Second, we present a complete development environment to easily specify and deploy agent-based solutions, especially for UI-oriented processes involving Augmented Reality technology.

1. MOTIVATION

Due to the digitalisation, the integration of service solutions for producers of physical devices (e.g. car manufacturers, domestic appliances, or printing industry) is becoming more and more relevant. This service-orientation leads to so-called *product-service systems*, which are highly complex. The ad-hoc manner, the seamless integration, and the unknown context of the product renders the development of product-service systems an exceptionally complex engineering task. The agent paradigm fits well for product-service systems, as agents have to be autonomous and self-configuring in order to adapt to the ever-changing context.

Two promising technologies for the developers of product service systems are *augmented reality* (*AR*) and *virtual reality* (*VR*). The option to configure, maintain and use a purchased system with a partial software solution increases comfort and usability. To show the complexity and the need for context awareness and autonomy, consider for example the dynamic visualisation of context-specific information and instructions onto the screen of AR-glasses, increasing the efficiency of a task execution. Furthermore, AR-devices allow for a bi-directional communication between user and system.

Although these approaches show a high potential and lead to new business models, reality shows that mainly large companies invest into these technologies [5]. In addition, the application of agent technologies is missing so far, which makes it difficult—especially for small to medium-sized enterprises (SME) that lack the financial resources needed—to develop solutions that are tailored towards the respectively given requirements. Yet, exactly those SMEs demand for such a solution due to their high maintenance costs and expensive machinery, which is being applied in heterogeneous environments.

Based on these observations, several requirements can be deduced that have to be met in order to make augmented reality a key technology for the efficient utilisation, maintenance, and repair of new products:

- An application-domain independent platform for an efficient and reusable development of AR-based software solutions,
- a large variety of AR-components in order to provide all needed functionality for different use cases,
- distribution with regards to the service resourcing,
- a mechanism to search and find services according to functional and QoS aspects, especially w.r.t. business requirements (e.g. pricing, reliability),
- a development environment for the design of complex processes,
- and the designed processes have to be easily deployable and independent of the used AR-device.

Summarizing, a distributed system with autonomous behaviour and automated interpretation of functionalities, combined with a large set of basic AR-features, can be a promising approach to foster the development of AR solutions. In the next chapter we shortly describe our approach before we characterise the demonstration.

2. APPROACH

Based on the requirements, we developed a distributed multi-agent solution combined with a development environment for the definition and modelling of agents and their services [1]. This demonstration shows how such an multi-agent solution can be build, deployed and tested. Within

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

this demo the basic functionalities (such as rendering, object recognition, and authentication) are all embedded into agents. In order to enable autonomous agent behaviour with regards to automated service selection, we enhance the agents' services with a semantic description covering functional as well as Quality-of-Service (QoS) parameters.

The development environment for the specification of complex processes provides a BPMN-based editor combined with mechanisms for searching and integrating existing services. The search is being done by a semantic service matchmaking component [2]. A complex service then is a service composition which is executed through the service providing agents. Finally, the developer can deploy the process dynamically to an interpreter agent, which then provides the new functionality to other agents and therefore also to the AR-devices, through which these processes can interact with the user.

3. DEMONSTRATOR

In the following we shortly describe the concrete use case of the demonstration, whereupon we illustrate how the demonstration will look like in detail.

3.1 Use Case

A manufacturer of autonomous vacuum cleaners intends to provide an AR-based maintenance process where a person not completely familiar with the architecture of the robot is able to maintain and in many cases also to repair it. Therefore, a maintenance process is being assembled by searching for different basic service components such as QR-code reader, authentication, identification, and machine data retrieval. After finalisation, the process can be deployed to the platform and it can be executed via an AR-device (either glasses or tablet).

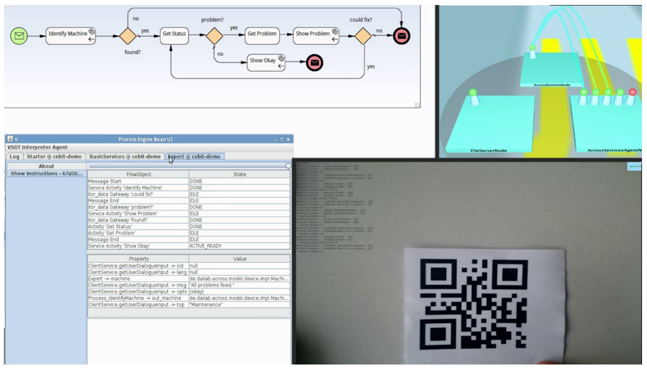
When the process is executed, it identifies the machine the user is looking at, diagnoses any problems, e.g. a camera device not being properly connected, and shows possible solution methods, e.g. where a loose cable has to be inserted. After displaying this information, the person maintaining the robot is able to fix the problems, and the AR-device finally sends an accordant update.

3.2 Sequence of Demonstration

The sequence of the demonstration is mainly divided into two parts: the process configuration and deployment, and the process execution.

In the beginning of the demonstration the audience will be presented a BPMN-based process modelling tool called Visual Service Design Tool (VSDT) [3]. Within this tool two processes are presented: a starter process that enables the user of an AR-device to select a process, and the concrete maintenance process by the robot manufacturer (see Figure 1, top left). The process contains a set of activities, which represent AR-actions provided by agents. The audience will be presented how the actions can be searched and imported. After finalization of both processes, they will be deployed to a so-called interpreter agent, which is responsible for executing the process. The deployment process and the whole agent architecture will be illustrated with the help of ASGARD [6], a visualisation tool for the multi-agent system JIAC V [4] (Figure 1, top right).

After the deployment, the execution phase is being initialised. A generic AR-assistance application is being started on a tablet (or potentially a Microsoft HoloLens). The app



FlowObject	State
Message Start	DONE
Service Activity 'Identify Machine'	DONE
Xor_data Gateway 'could fix?'	IDLE
Message End	IDLE
Xor_data Gateway 'problem?'	DONE
Service Activity 'Show Problem'	IDLE
Xor_data Gateway 'found?'	DONE
Activity 'Get Status'	DONE
Activity 'Get Problem'	IDLE
Message End	IDLE
Service Activity 'Show Okay'	ACTIVE_READY

Property	Value
ClientService.getUserDialogueInput -> cid	null
ClientService.getUserDialogueInput -> lang	null
Expert -> machine	de.dailab.across.model.device.impl.Machi...
ClientService.getUserDialogueInput -> msg	"All problems fixed."
ClientService.getUserDialogueInput -> opts	[okay]
Process_IdentifyMachine -> out_machine	de.dailab.across.model.device.impl.Machi...
ClientService.getUserDialogueInput -> top	"Maintenance"

Figure 1: Collage of the different tools and components shown in the demonstration. From top-left to bottom-right: VSDT editor, ASGARD runtime monitor, process interpreter UI, AR application on tablet.

then triggers the newly deployed starter process. After authentication, the user is asked to select the maintenance process to be executed. The demonstration is performed around an actual autonomous vacuum cleaner (iRobot), being equipped with different peripheral devices. The user gets the request to scan a QR-code (Figure 1, lower right) which is stuck on the robot for the identification of the machine. After that the process requests the current status data of the robot and displays it on the screen of the tablet. The user can then see that the camera device is not connected, and how the problem can be fixed, whereupon the user tries to insert the cable into the robot. Upon doing that the status of this parameter changes to *connected* and the maintenance of the robot is complete. The entire time during the process execution a computer display will show the whole agent communication and how they are interacting to fulfil the processes' goal (Figure 1, lower left).

4. FINAL REMARKS

The demonstration environment consists of three parts relevant for the agent community. First, there is the process development environment, which is able to create agents functionalities and dynamically deploy them on JIAC V agents. Second, there is the possibility to search for agents functionalities based on semantic descriptions. This can also be done at runtime by defining search templates processed at execution time. Third, the demonstration comes with a generic approach how to connect agents functionalities to AR-devices, such as glasses or tablets.

Furthermore the demonstration comes with a realistic product (IRobot), which is connected to the multi-agent system and provides real live data, making the scenario plausible as it shows a complete life cycle from process development to execution which finally leads to the solution of a problem.

A video of the demonstration can be accessed via the URL `https://dainas.dai-labor.de/~masuch@dai/across/`.

Acknowledgements

This work is based on a project funded by the German Federal Ministry of Economic Affairs and Energy under the funding reference number 01MD16016F.

REFERENCES

[1] J. Fähndrich, T. Küster, and N. Masuch. Semantic service management and orchestration for adaptive and evolving processes. *International Journal on Advances in Internet Technology*, 9(3&4):75–88, 2016.

[2] J. Fähndrich, N. Masuch, H. Yildirim, and S. Albayrak. Towards automated service matchmaking and planning for multi-agent systems with OWL-S – approach and challenges. In *Service-Oriented Computing – ICSOC 2013 Workshops*, volume 8377 of *Lecture Notes in Computer Science*, pages 240–247. Springer International Publishing, 2014.

[3] T. Küster, A. Heßler, and S. Albayrak. Process-oriented modelling, creation, and interpretation of multi-agent systems. *Int. J. Agent-Oriented Software Engineering*, 5(2/3):108–133, 2016.

[4] M. Lützenberger, T. Konnerth, and T. Küster. Programming of multiagent applications with JIAC. In P. Leitão and S. Karnouskos, editors, *Industrial Agents – Emerging Applications of Software Agents in Industry*, chapter 21, pages 381–400. Morgan Kaufmann, Boston, 2015.

[5] A. Mehler-Bicher and L. Steiger. *Augmented Reality: Theorie und Praxis.* Walter de Gruyter GmbH & Co KG, 2014.

[6] J. Tonn and S. Kaiser. Asgard – a graphical monitoring tool for distributed agent infrastructures. In Y. Demazeau, F. Dignum, J. M. Corchado, and J. B. Pérez, editors, *Advances in Practical Applications of Agents and Multiagent Systems: 8th Int. Conf. on Practical Applications of Agents and Multiagent Systems (PAAMS 2010)*, pages 163–173. Springer Berlin Heidelberg, 2010.