Visualization of an Agent-based Batch Process System

Mathias Rodler, Martin Melik-Merkumians, Wilfried Lepuschitz and Munir Merdan
Automation and Control Institute
Vienna University of Technology
Vienna, Austria
Email: {rodler, melik-merkumians, lepuschitz, merdan
@acin.tuwien.ac.at}

***Abstract*—Current batch process systems have a limited capability concerning agile adaptation in a dynamic environment. Distributed intelligent control systems based on agent technologies are seen as a promising approach to handle the dynamics in large complex systems. However, even though regarded as a promising approach, its wide application in industry is still missing. Lack of trust in the idea of delegating tasks to autonomous agents is recognized as one of the main reasons for this. This paper presents a visualisation tool for an agent-based batch process system able to offer the human user the complete overview of the agents' activities as well as a possibility to supervise their actions through an adequate Human Machine Interface. The visualization is also compatible with the standard ISA-88. The system is currently being tested and evaluated in the Odo Struger Laboratory at the Automation and Control Institute.**

I. Introduction

Batch processes are widely used in industry, primarily in the chemical and pharmaceutical industries. These industries face a continuously changing environment, where sudden component failures can significantly influence the system performance if not solved within a prescribed period. Moreover, they have to meet current market trends such as short production times, lower price and a broad spectrum of product and process variety. Additionally, there are requirements to enhance the reconfigurability, responsiveness and flexibility properties of process automation systems [1]. However, conventional control approaches based on hierarchical architectures are limited in dealing with such emerging requirements due to their inflexible structures and operating rules [2].

The introduction of artificial intelligence techniques is seen as a promising trend in the process industry [3]. Multi-Agent Systems (MASs) have the capability to respond quickly and correctly to change, and differ from the conventional approaches due to their inherent capabilities to adapt to emergence without external intervention [4], [5]. In the past, several agent-based approaches have been successfully applied to handle different issues in the process domain. Hong et al. describe a concept for a MAS designed mainly to solve scheduling problems [6]. A fault-tolerant multi-agent approach has been successfully used by Maturana et al. to control a chilled water system (CWS) thereby reducing the required manning level and improving the readiness and survivability of U.S. Navy shipboard systems [7]. MAS have been applied by Pirttioja et al. for advanced process automation condition monitoring [8] and by Seng and Srinivasan for detecting and diagnosing faults [9].

Although agent-based concepts were confirmed as a promising approach and deployed in a number of different applications throughout the last few years, their widespread adoption by industry is still missing. Lack of awareness about the potentials of agent technology [10] and paradigm misunderstanding [11] due to the lack of real industrial applications, missing trust in the idea of delegating tasks to autonomous agents [12] as well as concerns regarding the stability, scalability and survivability [13] are identified as major reasons for that. Besides, the specific nature of software agents, which are designed to be distributed, autonomous, and deliberative, makes it difficult to apply existing software testing, monitoring, and diagnostics techniques to them. For instance, agents operate asynchronously, in parallel and concurrently, which can lead to non-reproducible effects [14]. It is not ensured that two executions of the systems will result in the same state, even if the same inputs are used. As a consequence, looking for a particular error is difficult, as it is usually impossible to reproduce in a systematic fashion [15]. The introduction of tools, techniques and methodologies, which ensure easier and more abstract ways of agent system development, modification and management, will lead to a higher rate of acceptance as well as understanding of the concept. In this context it is of vital importance to offer the human user the complete overview of the agent's activities as well as a possibility to supervise their actions using an adequate Human Machine Interface (HMI). A user has to be able to follow and understand the current state of the process. This will on one side improve safety and risk grade and on the other side enhance trust in the idea of delegating tasks to autonomous agents. The assurance of security and trust in agents is a significant aspect to be considered in future solutions [10].

Visualization is the representation of abstract facts and information in general. The main task is to increase the cognitive abilities of the mind with external mechanisms. Using visualization, software processes can be made visible, creating cognitive images for the viewer that support the understanding [16] and through this reduce the complexity [17]. The advantage of visualization is that it can enable

Post-print version of the article: Mathias Rodler, Martin Melik-Merkumians, Wilfried Lepuschitz and Munir Merdan, "Visualization of an Agent-based Batch Process System," *2013 IEEE International Conference on Industrial Technology*, 2013. DOI: 10.1109/ICIT.2013.6505880

a fast interpretation of a wide range of information by the human. Visualization systems have to fulfill the following tasks: support for the understanding of large amounts of information; perception of emergent behaviors which are not predictable; fast discovery of errors by means of the visualized data; and shorter time and lower cost solutions to problems through timely and intuitive access to visual data.

Several visualization systems have been designed for agent technology. First to mention is the Manufacturing Agent Simulation Tool (MAST) representing a unified agent-based simulation, control and visualization framework for discrete manufacturing systems, especially dealing with dynamic product routing [18]. The second one, called CWS, represents a combination of simulation and monitoring of the chilled water system of a U.S. Navy ship. In this case the agent-based control system is responsible for the dynamic routing of water to keep various equipment on board cooled down [7]. The third tool, called JavaSniffer [19], has been designed to provide an advanced instrument for displaying the flow of messages among the agents. Additionally, JavaSniffer can bundle sequences of messages related to negotiations following the contract-net protocol and display the workflow to the user. The SOCRADES project, aimed at developing service-oriented infrastructure for distributed smart embedded devices, incorporates various technologies applied for visualization. In this project the visualization is provided by the commercial 2D/3D solution Delmia, which enables experimenting with the virtual production cell in a 3D environment [11]. Finally, the Actor-based Assembly Systems Architecture (ABAS), developed by Lastra et al., features a 3D view of the physical system controlled by intelligent physical agents [20], [21].

However, different from discrete manufacturing processes which rely on a number of visualization tools, an appropriate agent-based visualization for the batch process domain, beside the CWS that covers mainly routing and monitoring, is still missing. Therefore, an agent-based batch process tool, able to cover the entire agent-based batch process system, including its creation, function, monitoring, routing, and messaging, is required. Such a visualization should show the planned agent activities and the results of the execution. The important issue is to represent the dynamic nature of software agents and MASs, as well as to visualize their behaviors and actions.

This paper presents a visualization technique that enables better supervision and understanding of an agent-based batch process system. It enables humans to closely follow the agents' activities and states. Additionally, features for simpler creation and management of MAS are incorporated in the approach.

This paper is structured as follows. Section II introduces the agent control architecture. In the third section the visualization of the agents' world model is presented in detail. Section IV describes the implementation and the used tools and finally the paper is concluded in the fifth section.

II. Architecture of the Multi-Agent System

The following sections present the architecture of the MAS, whose activities need to be visualized.

A. Agent Architecture

In previous work, a MAS was developed, which is capable of handling the activities of a batch process system [22]. The MAS was also successfully used for controlling physical components that integrate on the fly reconfiguration abilities of the hardware-near control layer [23], for monitoring and diagnostic tasks [24], as well as for enhancing the reconfigurability, robustness and fault tolerance of a transportation system [25]. For the batch process domain, the following different types of agents and their general activities are introduced (see Fig. 1).

The Order Agent is used by a human operator or an external system to generate a product order. The related recipe is then selected and a corresponding job created including relevant data such as the product type, the quantity as well as the job ID.

Afterwards, a Task Agent is informed to start the execution of a new batch based on the recipe and the incorporated operations. The execution of a job to produce a batch is governed by the Task Agent, which determines the equipment for performing the operations of the recipe by searching for Automation Agents controlling the physical equipment with suitable services in the list of the Directory Facilitator Agent. As soon as appropriate Automation Agents are found, the Contract Net Protocol [26] is used to select the best suitable one and the tasks for the first operation of the recipe are created including transport tasks. These tasks are then forwarded to Work Agents for their accomplishment. When all tasks of one operation are finished, tasks for the next operation are created. Upon completion of all operations, the Task Agent informs the Order Agent about the finished production of the batch.

Work Agents are responsible for managing transport and production tasks, which incorporates calculating a path through the system in the case of a transport task. For the execution of such tasks, commands are sent to the corresponding Automation Agents that supervise the related equipment to e.g. start/stop or induce work state changes such as the activation or deactivation of a mixer or heater. The Work Agents also send reservation commands to valves and pumps for reserving a path for the transport of fluid or to tanks for reserving them for a particular operation. After completion of a task the related Task Agent is informed, which then will issue the next task.

Automation Agents are those agents that incorporate some kind of physical representation [27]. In the applied approach, the architecture of automation agents distinguishes low level control (LLC) and high level control (HLC) within each agent. The LLC is responsible for the direct control of the hardware and has to carry out some of its tasks under real-time constraints (e.g. closed-loop controllers, such as a PI controller for the flow control or a two-position controller for the temperature control). The HLC layer is in charge of higher-level diagnostics, of the coordination with other agents and of adaptations based on the representation of the world. It incorporates an ontology-based world model that provides an explicit representation of the agent's immediate environment and supports reasoning about its state, diagnostic activities

Post-print version of the article: Mathias Rodler, Martin Melik-Merkumians, Wilfried Lepuschitz and Munir Merdan, "Visualization of an Agent-based Batch Process System," *2013 IEEE International Conference on Industrial Technology*, 2013. DOI: 10.1109/ICIT.2013.6505880

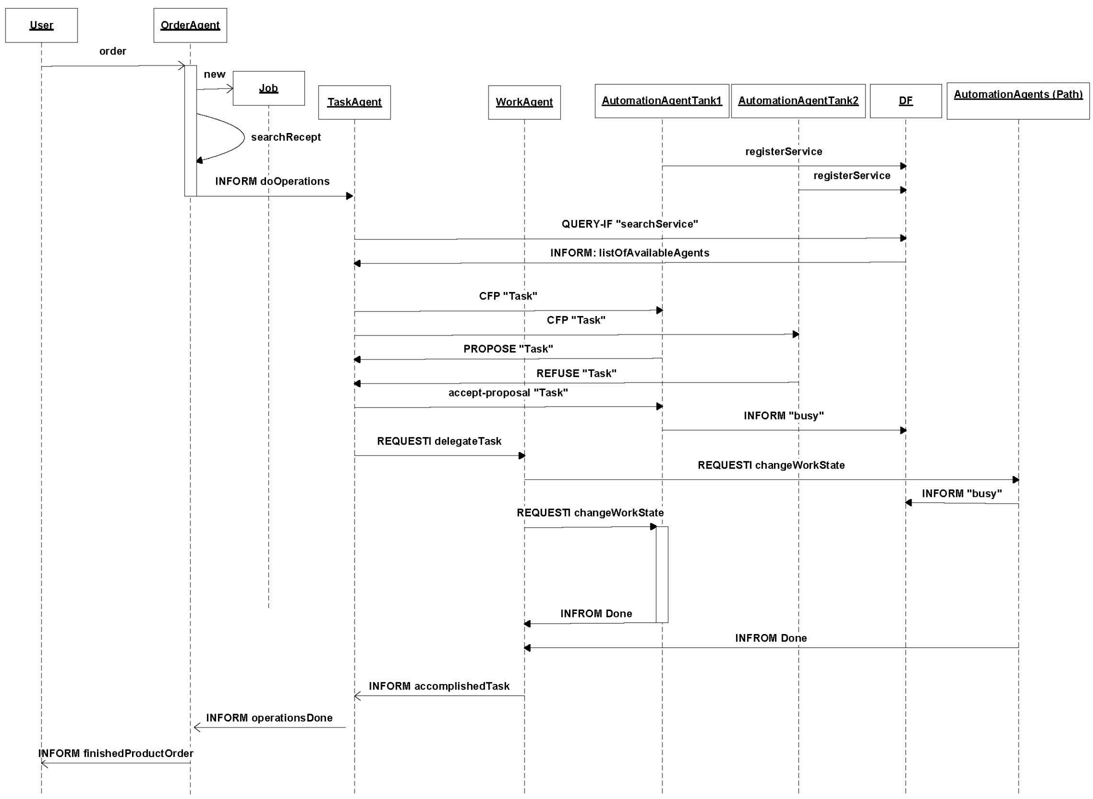

Fig. 1. The sequence of activities for producing a batch.

and the coordination with other agents [24]. We distinguish two types of automation agents: agents that are controlling one entity (e.g. a valve) and agents that are controlling a group of entities such as the Temperature Control Agent (TCA) controlling a heater as well as a temperature sensor. The services offered by the Automation Agents are listed in a register managed by the Directory Facilitator Agent (DF Agent) to be found by Task Agents for the execution of tasks. A busy flag is set for each Automation Agent when it is reserved for carrying out a distinct task to show that this particular Automation Agent is temporarily not available for other tasks.

B. States of the Agents

In order to keep the developed system close to current used software systems applied in the batch process domain, we used the states and commands as presented in the widely applied industrial standard ISA-88 [28] as a basis for defining the agents' states in the developed MAS. Hence, the following states are implemented: idle, running, complete, paused, holding (reserved), held, and aborted (see Fig. 2). If a procedure is in the idle state, i.e., it is available, it may be started, and the state will change to running. From the running state the procedure may be aborted (in the case of a failure), held, and paused, or the procedure will reach its end and return to the idle state passing the complete state. The state of a component, e.g., a valve can be: open, closed, reserved, and error. The error state would be reached if the valve fails to react on an open or close command from its controlling agent. It is important to notice that particular agents, such as order agent, work agent and task agent, do not have a physical representation. Consequently, in this approach they do not possess the states reserved, held or paused.

C. Communication within the Automation Agents

OPC Unified Architecture (OPC UA) is, as its predecessors from the OLE for Process Control (OPC) family of specifications (e.g., OPC Data Access (OPC DA), OPC Historical Data Access (OPC HDA), etc.), a mechanism for data access on field level devices. The main reason for the development of OPC UA was to combine all the different OPC interfaces into one unifying and powerful interface [29]. The newer OPC UA interface is an evolutionary development of the OPC family interfaces and it is based on the OPC specification and interfaces to enable a smooth migration path for existing plants. Additionally OPC UA introduces an advanced data model, capable of representing up to full-meshed data structures, whereas the OPC data model (a.k.a. name space) is limited to hierarchical

Post-print version of the article: Mathias Rodler, Martin Melik-Merkumians, Wilfried Lepuschitz and Munir Merdan, "Visualization of an Agent-based Batch Process System," *2013 IEEE International Conference on Industrial Technology*, 2013. DOI: 10.1109/ICIT.2013.6505880

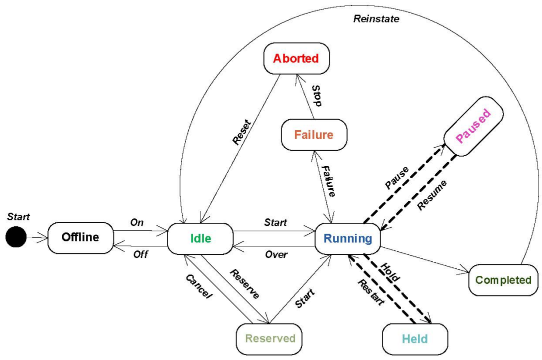

Fig. 2. Agent state diagram.

tree structures. Due to these Object-Oriented (OO)-like data structures, OPC UA is well suited for the interaction with MAS, as complex relationships between agents and data can be directly modeled in the OPC UA address space. This flexible mechanism enabled [30] to integrate the ISA-95 [31] model directly in the OPC UA address space.

We have chosen OPC UA as our main transport layer for communicating with the LLC of the automation agents, for all the reasons above, but the most important fact is that OPC and OPC UA are de facto standards in the industrial automation domain, implemented in almost every Programmable Logic Controller (PLC) or Industrial PC (IPC). Therefore, OPC UA (and of course OPC) can be used as universal data access interface on almost every automation component, and the MAS can be connected to every industrial automation system with only minor modifications.

III. Implementation

In order to to increase the acceptance in the industry as well as for easier handling, development and debugging purposes, it is necessary to offer a graphical representation of the agents and their interaction with each other. In Section I several visualization tools were mentioned but they are designed to represent individual graphical interfaces for special applications which make them not suitable for general purposes.

To be able to understand the behavior of autonomous agents, it is important to have an overview of their activities and states. This includes the LLC data that is provided by the OPC UA interface and the HLC messages which are handled by the Java Agent DEvelopment Framework (JADE) [32]. In general, the representation of information can either be displayed in a textual form or in a graphical manner. A higher information density can be reached easier by a textual representation of information, but to be able to follow the interactions of the agents' behaviors it is necessary to filter the information and represent the necessary fragments through a graphical illustration. This enables the user the ability to recognize changes within the MAS in a better comprehensible way. Nevertheless, the information in both textual and graphical representation should still be available in detail and provided to a user. To achieve a wide area of application, the visualization of the agents must be highly customizable.

The Eclipse Rich Client Platform (RCP) [33] framework has been chosen as basis and the graphical interface itself has been developed as a RCP Feature, which contains several plugins, instead of a standalone application. This advantage can be used to integrate the application easily in other RCP applications. For tracking the messages between the agents, the JADE sniffer is employed. Fig. 3 illustrates the information flow beginning from the MAS (each Agent) until the data is visualized by the user interface.

Two possibilities are implemented to offer the whole amount of data from the MAS as an overview to the user. An agent registers itself at the JADE container as sniffer for all messages that will be sent by the defined agents in the visualization. The direction of a message is considered as well as the content, sender and the receivers. Furthermore the sniffer listens to all necessary agent state changes (dead, born) as defined by JADE. The collected information is passed through to the Data Exchange Layer, which transforms and adds the information to the domain model. On the other hand, the data of the LLC is gathered through the developed *Pronto.Comm* interface, which is a client for reading OPC UA variables. The connection strings for each variable can be defined for each property of an agent in the offline state of the visualization.

The Eclipse Modeling Framework (EMF) [34] based domain model (see Fig. 4) is the main point for storing and exchanging data. It is a flexible and fast possibility to add new agent types without any coding as the creation of the needed source code is provided by the EMF. The Graphiti framework [35] based diagram editor is responsible to offer the ability to create a diagram of the agents' relationships in an offline state and to illustrate the operations of the MAS in the online state. It listens to changes in the model and visualizes the information in the diagram to achieve the

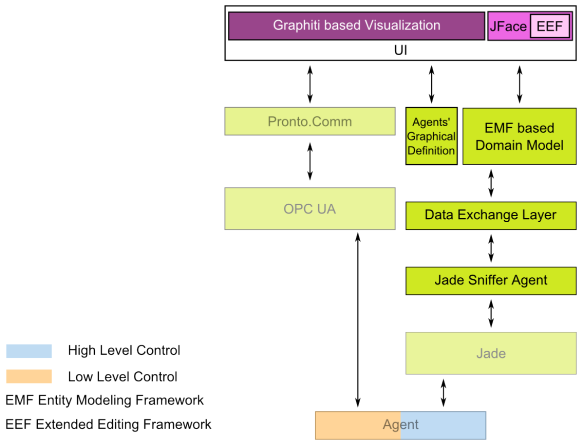

Fig. 3. Flow of information

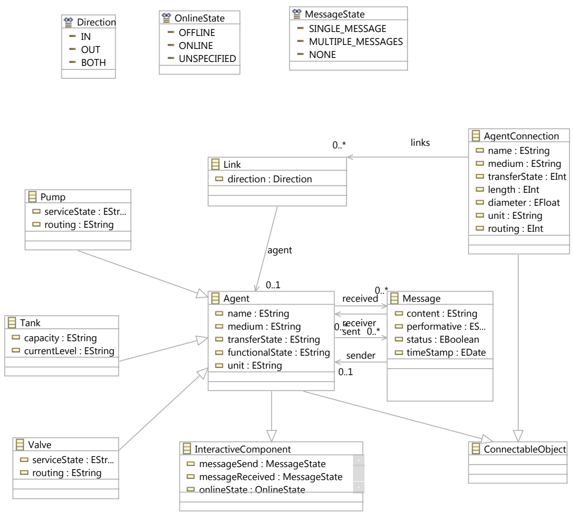

Fig. 4. Domain Model

graphical representation of the online state. The following possibilities are used to notify the user about changes:

- Decorators
- Tooltips
- Colors of agents

The Extended Editing Framework (EEF) [36] implements the views to modify the attributes of each component of the real system. The graphical definition is given by the shape configuration within an XML file which offers the information to draw a specific agent within a diagram. The name of an entry has to match the name of a derived class of agent and the currently possible figures are polygon, ellipse and poly line. The coordinates are defined in percentage relative to the size of the whole drawing of an agent, which enables the application to automatically rotate and resize any drawn agent. The objects can also be nested within each other, which allows a wide range of options to draw particular figures for each agent (as shown in Fig. 5).

This kind of visualization is able to present dynamic routing to a user showing shortest, broken, reserved or dirty paths after being informed by the according Work Agents. New components can easily be added to the system or the existing ones can be removed while the rest of the system still continues in its operation. Newly added agents, for instance representing new pipes or valves, are integrated online and can be used to transport fluid while removed agents just inform the community that they no longer exist and thus cannot be used.

IV. Conclusion

In this paper we presented a visualization tool that enables a user to follow and understand the current state of an agent-based batch process system. The system integrates various types of agents in a consistent configuration able to manage an entire batch process system and the agents cooperate and coordinate their activities in order to accomplish particular orders. The visualization is able to cover the entire management by displaying the overall behavior of the controlled system plus providing agent-specific information, like the state of the interaction with other agents or with their low level control. Thereby, our approach helps to understand the system behavior and to detect the patterns of emergent/aggregate behavior.

Currently we directly link this visualization tool with a laboratory process plant in the Odo Struger Laboratory at the Automation and Control Institute, Vienna University of Technology, in order to test its behavior with real-life batch processes. In further research we plan to test this visualization tool on more complex batch process plants and other multi-agent architectures as well as to migrate the approach to other industrial domains.

Acknowledgment

The authors acknowledge the financial support from the BRIDGE program by the Austrian Research Promotion Agency under contract FFG 829576 as well as from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement No. 284573.

References

[1] I. Seilonen, T. Pirttioja, and K. Koskinen, "Extending process automation systems with multi-agent techniques," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 7, pp. 1056–1067, Oct. 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197608001681

[2] N. N. Chokshi and D. C. McFarlane, *A Distributed Coordination Approach to Reconfigurable Process Control*, ser. Springer Series in Advanced Manufacturing, 2008, DOI: 10.1007/978-1-84800-060-5.

[3] W. Wang, O. Llanes-Santiago, and S. Jamsa-Jounela, "Future trends in process automation," O. Llanes, Ed., Feb. 2007, pp. 1–10. [Online]. Available: http://www.ifac-papersonline.net/Detailed/30275.html

[4] S. Bussmann, N. R. Jennings, and M. Wooldridge, *Multiagent systems for manufacturing control: a design methodology*. Springer Berlin-Heidelberg., 2004. [Online]. Available: http://eprints.ecs.soton.ac.uk/9548/

[5] R. Srinivasan, "Artificial intelligence methodologies for agile refining: an overview," *Knowledge and Information Systems*, vol. 12, no. 2, pp. 129–145, 2007. [Online]. Available: http://www.springerlink.com/content/h6ug7gh3217xt52n/abstract/

[6] X. Hong and S. Jiancheng, "Multi-agent based scheduling for batch process," in *Electronic Measurement and Instruments, 2007. ICEMI '07. 8th International Conference on*, 16 2007-july 18 2007, pp. 2–464 –2–467.

[7] F. Maturana, P. Tich, P. Łlechta, R. Discenzo, F. Staron, and K. Hall, "Distributed multi-agent architecture for automation systems," *Expert Systems with Applications*, vol. 26, no. 1, pp. 49–56, 2004.

[8] T. Pirttioja, A. Pakonen, I. Seilonen, A. Halme, and K. Koskinen, "Multi-agent based information access services for condition monitoring in process automation," in *Industrial Informatics, 2005. INDIN '05. 2005 3rd IEEE International Conference on*, Aug. 2005, pp. 240 – 245.

[9] Y. Seng Ng and R. Srinivasan, "Multi-agent based collaborative fault detection and identification in chemical processes," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 6, pp. 934–949, Sep. 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095219761000059X

[10] M. Pěchouček and V. Mařík, "Industrial deployment of multi-agent technologies: review and selected case studies," *Autonomous Agents and Multi-Agent Systems*, vol. 17, no. 3, pp. 397–431, Dec. 2008. [Online]. Available: http://dx.doi.org/10.1007/s10458-008-9050-0

Post-print version of the article: Mathias Rodler, Martin Melik-Merkumians, Wilfried Lepuschitz and Munir Merdan, "Visualization of an Agent-based Batch Process System," *2013 IEEE International Conference on Industrial Technology*, 2013. DOI: 10.1109/ICIT.2013.6505880

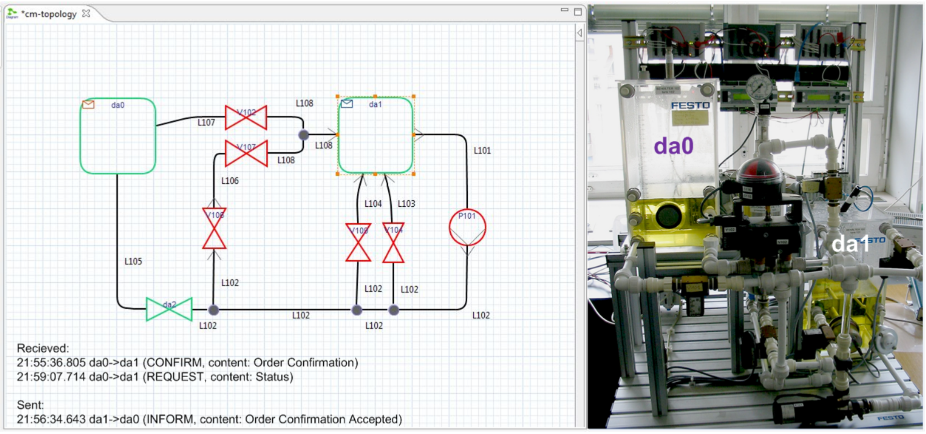

Fig. 5. Developed graphical interface (on the left) and laboratory process plant (on the right)

[11] G. Candido, J. Barata, F. Jammes, and A. Colombo, "Applications of dynamic deployment of services in industrial automation," in *Emerging Trends in Technological Innovation*, ser. IFIP Advances in Information and Communication Technology, L. Camarinha-Matos, P. Pereira, and L. Ribeiro, Eds. Springer Boston, 2010, vol. 314, pp. 151–158. [Online]. Available: http://www.springerlink.com/content/f860222w2q513q83/abstract/

[12] K. P. Sycara, "Multiagent systems," *AI MAGAZINE*, vol. 19, no. 2, pp. 79–92, 1998.

[13] A. Helsinger, M. Thome, and T. Wright, "Cougaar: a scalable, distributed multi-agent architecture," in *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, vol. 2, Oct. 2004, pp. 1910 – 1917 vol.2.

[14] C. D. Nguyen, A. Perini, C. Bernon, J. Pavón, and J. Thangarajah, "Testing in multi-agent systems," in *Proceedings of the 10th international conference on Agent-oriented software engineering*, ser. AOSE'10. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 180–190. [Online]. Available: http://dl.acm.org/citation.cfm?id=1965954.1965971

[15] M. Huget and Y. Demazeau, "Evaluating multiagent systems: a record/replay approach," in *Intelligent Agent Technology, 2004. (IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on*, Sep. 2004, pp. 536 – 539.

[16] S. Diehl, "Softwarevisualisierung," *Informatik Spektrum*, pp. 257–260, 2003.

[17] T. Ball and S. G. Eick, "Software visualization in the large," *Computer*, vol. 29, no. 4, pp. 33–43, Apr. 1996. [Online]. Available: http://dx.doi.org/10.1109/2.488299

[18] P. Vrba and V. Marik, "Capabilities of dynamic reconfiguration of Multiagent-Based industrial control systems," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 40, no. 2, pp. 213 –223, Mar. 2010.

[19] "Javasniffer," http://jade.tilab.com/, accessed: June 14, 2012.

[20] J. L. Martinez Lastra, C. Insaurralde, and A. Colombo, *Collaborative Design and Planning for Digital Manufacturing*, ch. Agent-based control for desktop assembly factories, pp. 265–291.

[21] P. Vrba, P. Tichy, V. Marik, K. Hall, R. Staron, F. Maturana, and P. Kadera, "Rockwell automation's holonic and multiagent control systems compendium," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 41, no. 1, pp. 14–30, 2011.

[22] M. Merdan, W. Lepuschitz, and E. Axinia, "Advanced process automation using automation agents," in *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on*, dec. 2011, pp. 34 –39.

[23] W. Lepuschitz, A. Zoitl, M. Vallée, and M. Merdan, "Towards self-reconfiguration of manufacturing systems using automation agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 41, no. 1, pp. 52–69, 2011.

[24] M. Merdan, M. Vallée, W. Lepuschitz, and A. Zoitl, "Monitoring and diagnostics of industrial systems using automation agents," *International Journal of Production Research*, vol. 49, no. 5, pp. 1497–1509, 2011.

[25] M. Vallée, M. Merdan, W. Lepuschitz, and G. Koppensteiner, "Decentralized Reconfiguration of a Flexible Transportation System," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 3, pp. 505–516, 2011.

[26] R. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *Computers, IEEE Transactions on*, vol. C-29, no. 12, pp. 1104 –1113, dec. 1980.

[27] M. Vallee, H. Kaindl, M. Merdan, W. Lepuschitz, E. Arnautovic, and P. Vrba, "An automation agent architecture with a reflective world model in manufacturing systems," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, oct. 2009, pp. 305 –310.

[28] Int. Society of Automation, *ISA-88 – Batch Control*, 1995.

[29] J. Lange, F. Iwanitz, and T. J. Burke, *OPC - From Data Access to Unified Architecture*. VDE Verlag, 2010.

[30] J. Virta, I. Seilonen, A. Tuomi, and K. Koskinen, "SOA-Based integration for batch process management with OPC UA and ISA-88/95," in *Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on*, sept. 2010, pp. 1 –8.

[31] *ANSI/ISA-95.00.01-2000, Enterprise-Control System Integration Part 1: Models and Terminology*, American National Standards Institute ANSI.

[32] "Java agent development framework," http://jade.tilab.com/, accessed: June 09, 2012.

[33] "Rich client platform," www:eclipse:org/rcp/, The Eclipse Foundation, accessed: June 09, 2012.

[34] "Eclipse modeling framework project (emf)," http://www.eclipse.org/modeling/emf/, accessed: June 09, 2012.

[35] "Graphiti framework," http://www.eclipse.org/graphiti/, The Eclipse Foundation, accessed: June 03,2012.

[36] "Eclipse modeling framework technology (emft)," http://www.eclipse.org/modeling/emft/, accessed: June 09, 2012.

Post-print version of the article: Mathias Rodler, Martin Melik-Merkumians, Wilfried Lepuschitz and Munir Merdan, "Visualization of an Agent-based Batch Process System," *2013 IEEE International Conference on Industrial Technology*, 2013. DOI: 10.1109/ICIT.2013.6505880