Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 49-12 (2016) 1526–1531

Agent-based communication to map and exchange shop floor data between MES and material flow simulation based on the open standard CMSD

Dr.-Ing. Frank, Timo*. Romero-López, Mónica*.
Block, Christian. Morlock, Friedrich**. Kuhlenkötter, Bernd****
Dr. Burges, Urlich*. Steinmetz, Waldemar ***.**

*GEFASOFT AG, Dessauerstraße 15, 80992 München Germany
(Tel: 0049-89-125565-170, -173; e-mail: {timo.frank, monica.romero.lopez}@gefasoft.de).
**Chair of Production Systems, Ruhr-Universität Bochum, Universitätsstraße 150, 44801 Bochum Germany
(Tel: 0049-234-32-26298, -29890, -26310; e-mail: {block, morlock, kuhlenkoetter}@lps.rub.de)
*** SimPlan AG, Edmund-Seng-Str 3-5, 63477 Maintal Germany
(Tel: 0049-6181-40296-16, 21; e-mail: {ulrich.burges, waldemar.steinmetz}@simplan.de).

Abstract: Simulation of manufacturing systems is state of the art in many companies. The simulations are used to optimize processes, to plan alternative solutions before a shop floor modification is realized or to test PLC programs. In all these cases, the simulation program has to represent the real shop floor processes and the corresponding machinery as well as possible. In this paper, we present a method to parameterize a simulation model with real-time shop floor data collected by an MES (manufacturing execution system). The approach uses autonomous agents based on JADE (JAVA Agent Development Framework) as system connectors. The agent communication is based on the open SISO (Simulation Interoperability Standards Organization) standard CMSD (Core Manufacturing Simulation Data) as exchange format which can be represented in XML.

Keywords: agents, JADE, manufacturing systems, simulation, shop-floor oriented systems, MES.

1. INTRODUCTION

Within the last years, the use of simulations became state of the art in the automotive industry. Among the simulation of the later products, the simulation of the manufacturing system gains increasing importance. What we consider as manufacturing system in this context consists of all shop floor components below the manufacturing execution system (MES) (VDI 2007).

In contrast to most product simulations using e.g. FEM (Finite Element Method), the material flow simulation is also useful during system operation. Many changes, such as the exchange of a machine, occur during operation. For an employee, it is often difficult to assess the effects of these changes in production directly (Wang, Chang, Xiao & Wang 2011). The employee has to answer several questions. Would a change really increase production volume? Does a change unexpectedly affect other parts of the production line? Since the systems are usually very complex, the answers to these questions are not easy to determine. This is where simulation is used as a decision support tool, for example in order to:

- calculate the theoretical output – the capacity – of a production line,
- identify a bottleneck within a production line.

As in all simulations, it is important that the simulation parameters correspond to the parameters of the real-world object. A special characteristic of simulations in the context of production is that the real production process is changing very quickly. In addition, many aspects directly influence the results of the simulation. For this reason, the system boundary must include many system elements. This leads to high costs to maintain the simulation and increases the likelihood of errors. If the results are not reliable, however, the system's acceptance decreases dramatically.

To address these challenges, an approach is presented, which enables an automatic parameterization of a simulation model to offer the possibility of online simulation on the way to a digital factory. The approach is based on agents, which collect the needed data from a manufacturing execution system in real-time.

This paper is structured as follows: in the next chapter, the state of the art is presented. In chapter 3, the core concept of the data exchange between MES and simulation will described. This section includes the basic agent-based architecture, the mapping concept, and the communication procedure. In the last section – chapter 4 – the implementation of our concept based on the two software products Legato (MES) and Plant Simulation (simulation) is shown. The paper concludes with a summary and outlook.

2. BASICS AND PROBLEM DEFINITION

"Digital factory is the generic term for a comprehensive network of digital models, methods and tools – including simulation and 3D visualisation – integrated by a continuous data management system. Its aim is the holistic planning, evaluation and ongoing improvement of all main structures, processes and resources of the real factory in connection with the product." (VDI 2010) In the context of this work, the two

2405-8963
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2016.07.796

aspects simulation models for planning and input data are considered.

2.1 Simulation

As mentioned, the field of simulation is widely spread. In every phase of the product lifecycle different simulations can be done (VDI 2010). Therefore, at first the aim of simulation can be separated. In production industry, the product-related simulation with regard to stresses and tensions can be differentiated from the simulation of fabrication. In the production process, there can be done ergonomic or processes, kinematic, robotic simulations or even material flow simulations. Furthermore, simulation can be divided into different forms of modelling types or internal calculation mechanisms. In the focused simulation of material flows, the discrete-event-based simulation is used for the production processes (VDI 2014).

These simulations can be used in three phases: 1) the planning phase with a long-term planning of the production structure, 2) in the implementation phase to validate and optimize the planned structure or 3) in the operation phase for production planning and control. The last phase has the aim to calculate a new order schedule, change the product routing or optimize the order sequence based on product quantities or resource capacities (VDI 2014). In contrast to the defined objective, the current focus of the digital factory is the planning and design before the start of production. The simulation in operation is currently not in use. (VDI 2008) To implement simulation in this phase, an online simulation is necessary. Figure 1 shows the components and the dataflow for such an online simulation. Simulation results improve with real input data which describe the current state. Therefore, for planning purposes the actual situation has to be committed in the simulation model before every simulation run. To solve this issue, it is to clarify where the data comes from, how it is transferred into the model and how the data is structured (Bengtsson, Shao & Johanson 2009).

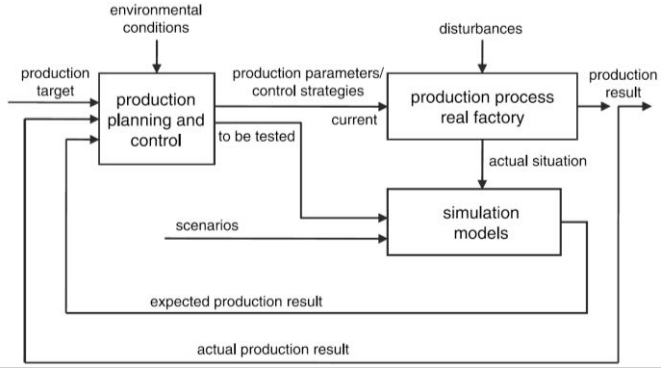

Fig. 1: Components of online simulation (VDI 2008)

2.2 MES

The first step to address this issue is to define a system which represents the instantaneous state of the real factory. MES as the production management level is the medium level between the enterprise management and the shop floor. Manufacturing execution systems are modular systems representing the interface between the long time planning ERP (enterprise resource planning) systems on the top level and the executing shop floor level. MES receives production orders from the ERP and through the module machine data collection, an MES owns a link to the real resources. Due to this feature, an MES is the ideal system as data source for simulation input parameters (VDI 2007).

2.3 Data connection

The second step to address the issue of an online simulation is to exchange information. To synchronize the simulation models with the real world the systems have to be connected. In order to implement an online simulation this connection should be able to communicate in real time. However, the connection of different systems is a complex challenge. Every system has specific proprietary exchange formats which are rather complicated to handle, a fact that has been a general problem in the last decades.

With regard to simulation systems, there are several approaches to link the digital world with the real factory. In this context, the simulation system PlantSimulation offers various interface options from a specific spreadsheet interface for Microsoft Excel to more general interfaces like ODBC, DDE, XML or TCP/IP socket connection. All of them have different disadvantages. Either the interface is very static or they need a direct connection to a specific database like the MES-database. The MES side is even more individual because a lot of different ME systems are on the market and they are mostly customized. Therefore, an individual customer solution with direct interfaces is needed and has to be programmed. Currently, the data transfer is only implemented in the direction of the simulation. As a result of security risks through direct interfaces, data exchange in both directions is not in use today.

The challenge for research and industry is to link production software systems intelligently and thus to enable a bidirectional data exchange without redundancies. In recent years, there has been a trend to multi-agent-systems in computer science (Geng, Chen, Liu, Zhang 2005). There is no uniform definition for an agent in literature. However, the association of German engineers VDI/VDI founded a professional committee for agent systems and developed a definition which is used here. (VDI 2011) defines an agent as follows:

“An agent is an encapsulated (hardware/software) entity with specified objectives. An agent endeavours to reach these objectives through its autonomous behaviour, in interacting with its environment and with other agents.”

Other authors expand this understanding by the agent characteristics of being autonomous, reactive, proactive, mobile, communicative and social (Wooldridge 2009; Bellifemine, Caire & Greenwood, 2007). Especially the last two aspects are important in this work and can only be used through multiple agents. Therefore, a multi-agent-system is necessary, which is defined by (VDI 2011) as follows:

“A multi-agent system consists of a set of agents interacting to fulfil one or more tasks.”

Through the acting of autonomous agents, a multi-agent-system may be used to support communication in production systems. Due to the communication aspect, agents enable an interchanging of data and information. The data is processed by a system-related agent and not by an individual direct interface. Thereby, agents form an indirect interface. This reduces security risks and enables bidirectional data exchange between two or more systems simultaneously.

According to (VDI 2011), special frameworks and runtime environments can be a fundamental basis for implementing agent systems. In the presented approach the Framework JADE (JAVA Agent DEvelopment Framework) is used (Bellifemine, Caire & Greenwood 2007).

2.4 Data structure

In addition to the technical data exchange, the data structure is crucial. A data structure is necessary to form the data to information the agents can handle. In the context of the digital factory, many different datasets and exchange formats are available (e. g. STEP for product description) (VDI 2010). However, data structures for material flow simulations are rare. (Bergmann 2013) analyzes different types in the context of automatic model generation and chooses CMSD (Core Manufacturing Simulation Data) as the best data structure to integrate data relevant for material flow. CMSD is an open SISO (Simulation Interoperability Standards Organization) standardized format (SISO 2010) and has a XML representation (SISO 2012). Based on the XML representation and many possibilities for the agent system, the CMSD format has been chosen as data structure for input parameters. Furthermore, it is to define which data has to be exchanged. In the first step, only those indicators describing machines as input parameters have been chosen.

3. CONCEPT

As shown in the previous chapter, several basic concepts exist. In this chapter, we present our core concept to combine them in order to exchange shop floor data in real-time. As detailed above, the main goal is to connect an MES with a simulation. Figure 2 shows a system overview. The MES collects the shop floor data and stores it within a database. This data is a digital representation of the production plant. The simulation model is also structured in such a way that it represents the real production as realistically as possible. Of course, both digital representations can never perfectly replicate reality and will therefore contain a certain amount of representation errors. The extent of representation errors of the MES is nearly constant over time as the MES updates the digital representation cyclically. In contrast to the MES, the representation error of the simulation normally increases over time as the simulation is only modelled once.

To minimize the representation errors of the simulation model, it is necessary to update the simulation regularly. In our concept, this is done by an agent-based connector, which is described in detail below.

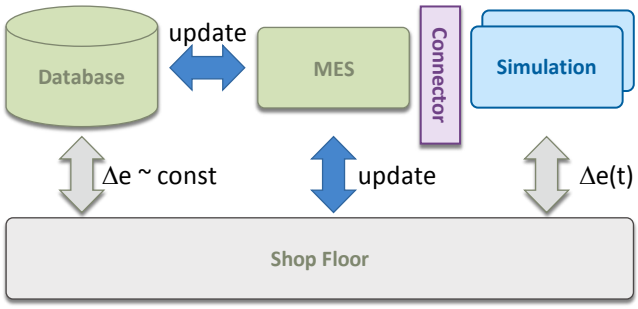

Fig. 2: Error in MES vs. error in simulation models

3.1 Multi-Agent-System as system connector

The main task of the connector is to parameterize the simulation model, to start/stop the simulation and to read the simulation results. Both the MES and the simulation have their own system state, cycle time, and their own data model. For this reason, it is useful to keep the two systems separate. Furthermore, this approach decentralizes the planning system. Figure 3 shows the architecture of the connector. To disconnect the systems, autonomous agents are used. One agent is responsible for each system. This agent knows the data model of the system, knows the systems behavior and knows how to react to system events.

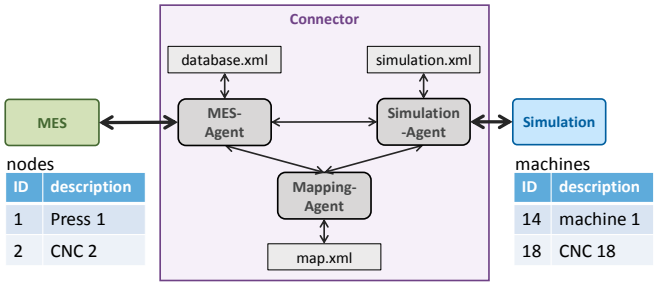

Fig. 3: Architecture of the agent-based system connector

Apart from the challenge that all systems have a different data model, all elements of the respective data model are identified differently. For example, a machine is called *"machine 1"* with the identifier *"ID 14"* in the simulation program, whereas the same machine is a node named *"press 1"* and identified by *"ID 1"* in the MES data model. The concept provides a mapping agent to resolve these differences and will described in chapter 3.2 in detail.

An additional challenge is that the systems have different cycle times. One important point is that the data consistence is influenced by this cycle time. For example, there could be a loss of individual work pieces if the capturing of work piece positions is not done consistently by the MES agent. For this reason, the data is not exchanged attribute by attribute, but as a whole simultaneously.

3.2 Mapping-Agent

In order to synchronize the real and the digital factory, the IT-systems have to talk the same language. For this purpose, it is necessary that the focused objects have the same names

or identifiers. This requirement is not given in current data sets. As introduced in the chapter above, the resource identifier between the real world (MES) and the digital factory (simulation) are not consistent. For instance the machine names in the MES can be counted (e.g. 123) in contrast to names of characters (e.g. drilling) in the simulation area. Therefore, a translation service is needed.

This translator is implemented in a separated agent which is called Mapping-Agent. The data mapping is a core feature of the presented solution. It is used to define the relations between the systems. The mapping workflow is separated in the following three steps.

1. At first, all agents of the connected systems send a list of their managed resources. In the presented approach, the considered systems are the MES, which represents the real factory, and the simulation as the digital factory. The system agents push all relevant IDs via a message to the Mapping-Agent. The MES-Agents sends all internal known IDs, whereas the simulation-Agent only sends the IDs of the current model. This step is repeated frequently.
2. Secondly, the Mapping-Agent processes the two ID lists. In the initialization phase, a human worker has to map the factory and simulation IDs in a graphical user interface. The chosen relations are saved in a XML based file. In case of new unmapped IDs, the worker gets a message to map the IDs. An example of a mapping user interface is shown in figure 4.
3. After the start-up phase and in case that no ID of the current simulation model is unmapped, the multi agent system can communicate autonomous. The exchanged CMSD based messages are translated by the Mapping-Agent.

The detailed mapping communication can be seen in Fig. 6

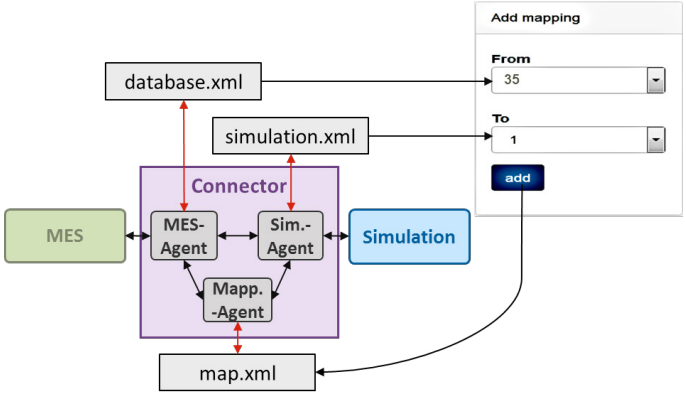

Fig. 4: Prototype of the mapping user interface

3.3 MES-Agent

The MES-Agent has to connect the MES with the agent system. The main task of the agent is to select the right data from the database. This data has to correspond to the simulation goal, the simulation time, and to the machines which are simulated. Therefore, data of four different categories have to be selected:

- machine status – machine key performance indicators (KPIs), machine state
- process image – work piece positions, process values
- shift information – shift times, planned breaks, times for Total Productive Maintenance (TPM)
- order information – current orders, planned orders

For example, the simulation runtime specifies which KPIs have to be selected. If the quantity of the next night shift is simulated, the agent has to parameterize the simulation using the average night shift KPIs from the last days. In contrast, a simulation of the current shift would need current KPIs of the machinery. A similar problem arises with the machine state. The current state is only needed if the simulation starts at the current time. A start time which is in the future needs a default machinery setup.

The MES agent knowledge base includes a representation of the MES database as a class diagram. This class diagram includes all relevant information and enables the agent to select all requested information. One example is the link between machines and KPIs. One class *NodeList* describes a list of existing machines. This class links to the *node* class. The *node* class includes a detailed description of one machine and links to the class *KPI* that describes all machine KPIs in detail. This class includes time information like shift times. Based on these classes it is possible to perform an XML (un)marshalling with e.g. JAXB to provide possibility of exchanging data with other agents.

3.4 Simulation model library and simulation agent

Based on the simulation tool Plant Simulation, a special library with a standardized communication interface to the simulation agent was developed. Main elements of the library are:

- administration and statistic elements
- material flow elements, such as source, buffer and a flexible process element (variant-depending process time, setup, tool change, periodic stop and assembly function)
- communication interface to the simulation agent

The focus of the material flow elements is on the mechanical processing and assembly of manual or partly automated linked machining systems.

The main tasks of the simulation agent are:

- the administration of requests,
- simultaneous remote control of simulation models
- intermediation between simulation model and the SOPHIE components (e.g. MES).

For each request a separate simulation model is started, so that the simulation agent is able to handle multiple requests at the same time. The simulation agent is able to start and remote-control models of different simulation tools (e.g.

Plant Simulation or AnyLogic). It is necessary to ensure that the agent may communicate securely with each model. For this, the first contact made with the model via a default configuration. After successfully making first contact, the simulation agent transmits a unique configuration for a subsequently secure communication.

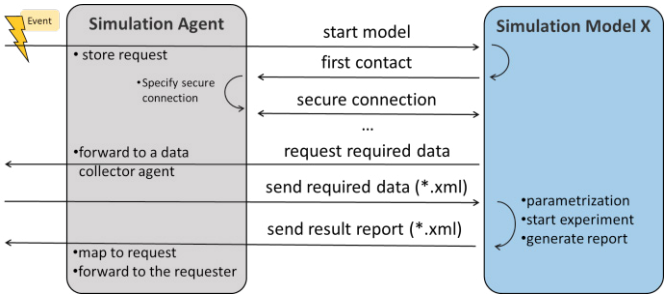

Fig. 5: Communication between simulation and simulation agent

After a secure connection has been ensured, the model gets the required parameter data (xml file) from the simulation agent (see Fig. 5). The subsequently generated results are sent to the agent and then forwarded to the requester.

3.5 Agent communication

Before the communication process flow that involves MES, simulation and mapping agents can be described, it is necessary to define some important elements for a correct data exchange. Therefore, we developed the agent communication based on the guide provided by JADE (Bellifemine 2010). Among others, this manual provides introductions to parameterize an agent communication language, to establish an ontology for the communication, and to adopt a communication protocol.

As a content language, JADE provides text-based FIPA-SL or XML codecs. Furthermore there is a byte formatted LEAP codec (Bellifemine 2010). All options are domain-independent. In this case the XML codec was chosen to avoid coding problems.

In order to identify the different elements of the communication, an ontology definition is needed. For the definition of the ontology, JADE provides three elements: *Concept* (represents an object), *Predicate* (represents information or a state of an object), and *AgentAction* (represents an order or required action). (Bellifemine, Caire & Greenwood 2007) Figure 6 shows the ontology approach.

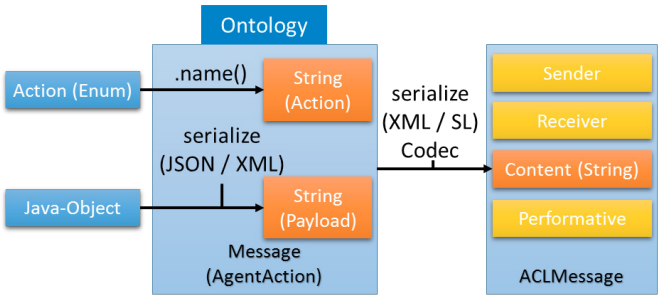

Fig. 6: Ontology based message structure for the agent communication

We adopted the FIPA-Request-Protocol (Foundation for Intelligent Physical Agents 2002) to ensure that the agents can communicate by the same protocol and to reduce the communication complexity.

Once all elements are defined, the process flow can be designed (compare figure 7, where each communication is based on FIPA-Request-Protocol). Before a simulation can be requested, an event from the MES or by an employee (manual) is needed (Step 1). After that, the simulation will require data from the database in order to simulate with shop floor data. The simulation agent creates a special data collector agent and gets a message with all relevant resources to collect data for (Step 2). The collection process starts with a translation of the simulation's ID list. The Result is a List of sources and the origin identifiers. In this case the source is always the MES and gets a request to deliver all necessary data for the identifiers. The MES responses with a CMSD based message, which has to be translated in the simulation context again. The simulation will receive an XML document, in which all relevant information is stored. Afterwards, the simulation program will kill the data collector agent, simulate the model and send the results back to the MES agent. The MES agent can save these results in the database.

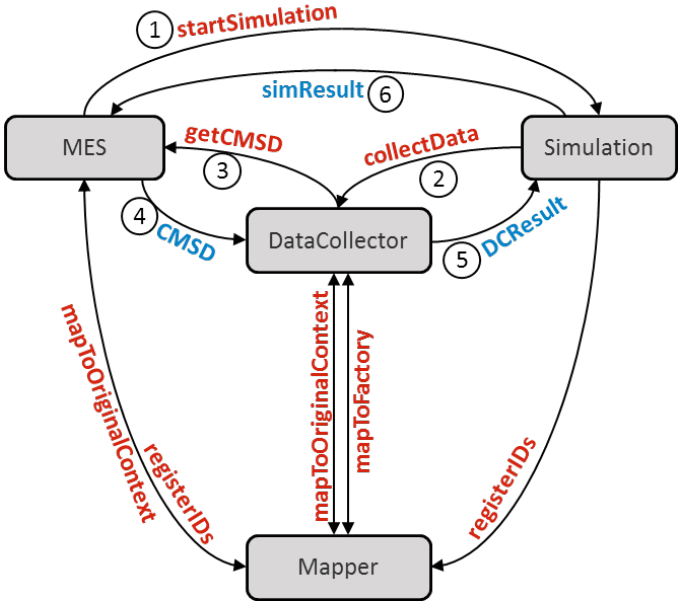

Fig. 7: Communication between agents

4. EVALUATION

To evaluate the concept described in this paper the MES Legato and PlantSimulation have been connected via the agent network. A detailed extract of the communication is described in the following section. Both, the simulation and the MES run on different devices.

After the initialization the MES agent gets the *getCMSD* (Step 3) request from the DataCollector agent and provides a CMSD. This CMSD is shown in figure 8. Every node has an identifier and a list of properties. In this use-case these properties are selected KPIs of the nodes identified by the name.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CMSDDocument>
    <HeaderSection/>
    <DataSection>
        <Resource>
            <Identifier>165</Identifier>
            <Property>
                <Name>Gesamt-Stördauer (FTTOT)</Name>
                <Value>26991</Value>
            </Property>
            <Property>
                <Name>Stördauer technisch (FTTEC)</Name>
                <Value>26980</Value>
            </Property>
            <Property>
```

Fig. 8: MES CMSD output

The KPI values are read by the simulation agent and set as parameters within the simulation. In the prototype, the mapping of the KPIs is done by unique names. The mapping of machines and nodes is realized by the mapping agent. After the experiment, a report is generated and shown within PlantSimulation.

5. SUMMARY AND OUTLOOK

The paper presents an approach to connect the real factory represented by a MES with the digital factory here as a material flow simulation to solve various planning issues. The developed solution is agent-based. One of the next research and development steps is to enumerate further input data (e. g. shift information etc.). Additionally, a data structure for simulation results has to be defined to save and communicate the output data, which cannot be handled in the CMSD structure.

Furthermore, the current state of research and development of the presented agent-based system connector is similar to an “Enterprise Service Bus”. However, through the agent-based architecture the connection is more flexible and open for extensions. For instance it is possible to integrate additional software systems or especially humans as Worker-Agents in the communication process. A first approach was developed. The Worker-Agents can communicate with other agents. To create an interface for the real worker a webserver is integrated in the JADE agent framework. The web interface makes it possible to interact with the simulation from every device, which has a web browser.

REFERENCES

Bellifemine, F. L., Caire, G., Greenwood, D. (2007). Developing multi-agent systems with JADE. Chichester, England.

Bellifemine, F. L., Caire, G, Trucco, T, Rimassa, G. (2010). JADE Programmer's Guide.

Bengtsson, N., Shao, G., Johanson, B. 2009. Input data management methodology for discrete event simulation.

Bergmann, S. (2013). Automatische Generierung adaptiver Modelle zur Simulation von Produktionssystemen. Technische Universität Ilmenau, Ilmenau.

Caire, G., Cabanillas, D. (2010). Application-Defined Content Languages and Ontologies.

Foundation for Intelligent Physical Agents (2002). FIPA Request Interaction Protocol Specification.

Geng, Chen, Liu, Zhang (2005). Consensus of a heterogeneous multi-agent system with input saturation. *Neurocomputing* 166, 382-388.

SISO (2010). Core Manufacturing Simulation Data - UML Model, SISO-STD-008-2010.

SISO (2012). Core Manufacturing Simulation Data - XML Representation, SISO-STD-010-2012-DRAFT.

VDI (2007). Manufacturing Execution Systems (MES), VDI 5600-1, 2007.

VDI (2010). Multi-agent systems in industrial automation – Fundamentals, *Beuth Verlag*, VDI 4499-2.

VDI/VDE (2008). Digital factory – Fundamentals, *Beuth Verlag*, VDI 4499-2.

VDI/VDE (2011). Digital Factory – Digital Factory Operations, *Beuth Verlag*, VDI 2653-1.

VDI/VDE (2014). Simulation of systems in materials handling, logistics and production – Fundamentals, *Beuth Verlag*, VDI 3633.

Wang, J., Chang, Q., Xiao, G., Wang, N. (2011). Data driven production modeling and simulation of complex automobile general assembly plant. Computers in Industry.

Wooldridge, M. J. (2009). An introduction to multiagent systems. *2nd ed. Chichester*, U.K.: John Wiley & Sons.