Semantic Extension of Agent-Based Control: The Packing Cell Case Study

Pavel Vrba[1], Miloslav Radakovič[1,2], Marek Obitko[1,2], and Vladimír Mařík[1,2]

[1] Rockwell Automation Research Center, Pekařská 695/10a, 155 00 Prague 5, Czech Republic
{pvrba,mradakovic,mobitko,vmarik}@ra.rockwell.com
[2] Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Technická 2, 166 27 Prague 6, Czech Republic

Abstract. The paper reports on the latest R&D activities in the field of agent-based manufacturing control systems. It is documented that this area becomes strongly influenced by the advancements of semantic technologies like the Web Ontology Language. The application of ontologies provides the agents with much more effective means for handling, exchanging and reasoning about the knowledge. The ontology dedicated for semantic description of orders, production processes and material handling tasks in discrete manufacturing domain has been developed. In addition, the framework for integration of this ontology in distributed, agent-based control solutions is given. The Manufacturing Agent Simulation Tool (MAST) is used as a base for pilot implementation of the ontology-powered multiagent control system; the packing cell environment is selected as a case study.

Keywords: manufacturing, distributed control, multi-agent systems, semantics, ontologies, semantic web.

1 Introduction

The theory of multi-agent systems (MAS) has been recognized as a promising paradigm for implementing the next generation of distributed, robust and dynamically reconfigurable automation control systems. The current trend of deployment of agents is obvious at all levels of the manufacturing business. At the lowest, real-time control level, so called holons or holonic agents are deployed. They are usually tightly coupled with the hardware as they directly interact with the real time control programs, implemented in standards such as IEC 61131-3 (e.g., ladder diagrams) or IEC 61499 (function blocks) [4]. Intelligent agents are also used for production planning and scheduling tasks both on the workshop and factory levels [18]. The application of MAS approaches is also relevant for implementing the vision of virtual enterprises, where various enterprises form temporary alliances for sharing their resources and competencies in order to increase their competitive advantage [5].

A multi-agent system is a federation of software agents interacting in a shared environment in order to cooperate and coordinate their actions given their own goals and plans [16]. The agent is designed to pursue its local objectives without direct external intervention, perceive and respond to changes in its environment and communicate with other agents in order to share information and cooperate on common goals. The strength of agent solutions is in the ability to exhibit complex behavior with the notions of global intelligence and self-organization even though the behaviors of individual agent instances could be quite simple.

In the past, the research in MAS field aimed at developing standards for designing, implementing and maintaining agent systems, with special attention paid to the inter-agent communication. The FIPA organization provided a comprehensive bundle of standards for multi-agent systems, including popular Agent Communication Language (FIPA-ACL) [10] and the Directory Facilitator (DF) for registering and searching of agent services. The ACL language defines an explicit structure of message including different slots like sender name, receiver name, language, content, etc. Compliance with the FIPA-ACL ensures syntactic interoperability of heterogeneous agent systems. The receiving agent is able to parse the slots of the message and even understand their meaning, i.e. who is the sender, what communication protocol this message follows, etc. However, such semantics is given just for the message level itself. When immersing down into the message content, the receiving agent might not be able to understand if it does not posses the same semantical description and interpretation of concepts in the message as the sending agent does.

With the evolution of semantic technologies in the past years the focus of researchers in the MAS area shifted naturally towards semantic interoperability and advanced methods for expressing and handling knowledge. Sharing information, goals and plans in the agent's social context as well as internal functionality of an agent in terms of maintaining knowledge and reasoning about it are more and more perceived as knowledge-intensive tasks.

The aim of this paper is to briefly review the state-of-the-art of applications of semantic techniques in MAS research related to manufacturing control systems. The proposal of generic manufacturing ontology and guidelines for its use in designing the agent-based control systems is discussed. The semantic extension of agent solution for packing line control is presented as a case study.

2 Ontologies for Distributed Manufacturing Control Applications

To facilitate knowledge sharing for the operation of the whole system and achieving common goals, agents must be able to understand information exchanged in messages. In order to have an extensible system, this understanding should be guaranteed even when the kind of communicated information is extended or changed, without any needs in reprogramming efforts. A possible way for ensuring the common understanding is the application of ontologies. Ontologies provide semantic description of a particular domain (such as manufacturing domain), i.e. they specify the meaning of entities and their relationships in the domain and define the parameters, constraints and possible consequences. Ontology is used by agents to represent the data and information about the environment as well as to share this data with other agents by means of messaging. In the context of Agent Communication Language (like mentioned FIPA ACL), the ontology is used for expressing the content of the message.

2.1 Ontologies in Manufacturing Systems

The ontologies for manufacturing systems have been discussed in detail for example in [17]. There are general purpose manufacturing ontologies such as MASON focusing on manufacturing operations [12], NIST's description of shop data model [15], definition of so called Automation Objects merging mechatronic and control models [13] or OOONEIDA focusing on the infrastructure of automation components [24]. The existing norms like ANSI/ISA-95 or ANSI/ISA-88 [3] should be considered as solid basis for creation of manufacturing ontologies. The ISA-95 „Enterprise-Control System Integration“ standard defines hierarchical model of production organization and event flow and provides basic concepts for the integration of control system with business systems of the enterprise. The ISA-88 "Batch Control" describes in more detail the batch process production environment. It defines hierarchical model of production system from the enterprise, through areas and units down to control modules.

There are also ontologies created specifically for agent-based manufacturing systems – examples are ontologies for shop floor assembly [6], for reconfiguration purposes [1] or logistic planning [20]. These ontologies typically describe the layout of manufacturing equipment together with the description of what operations this equipment provides. Sometimes attention is paid also to low level control connection. The production is then described using sequence of operations, which may in addition specify tools needed to perform operations. Sometimes this sequence is described directly in ontology, not in the knowledge base. In general, the idea is to describe production equipment that is able to perform particular operations and use this description to find appropriate machines to perform production sequence.

We have studied these proposals but found that none of these ontologies provide a suitable semantic model that would fulfill requirements for generality, extensibility and deployability in agent-based discrete manufacturing control. The ANSI/ISA 88 is exclusively designed for batch processing industry what influences the way how physical model as well as process model is defined. The process model for instance defines process stages containing process operations that are furthermore composed of process actions. This seems to be too limiting in case that more than three levels are required in the hierarchy. In the proposed ontology presented in next section the production plan is composed of steps, where each step has either a basic operation or another complex plan associated. The same limitations and also some inconsistencies can be found also in the MASON ontology. Nevertheless, these ontologies provided inspiration for our ontology intended for flexible multi-agent manufacturing system.

2.2 Ontology for Flexible Manufacturing System

We have developed a set of ontologies in the Web Ontology Language (OWL) [7] (using Protégé editor) for description of customer orders, production plans, and the manufacturing system itself including transportation aspects. The goal of these ontologies is to serve as a base for further extensions for particular applications. For example, any new product type can be introduced with new types of parameters and new kinds of operations. The agents in the manufacturing system utilizing a shared ontology structure (see Sect. 3) do not have to be stopped and reprogrammed to absorb this

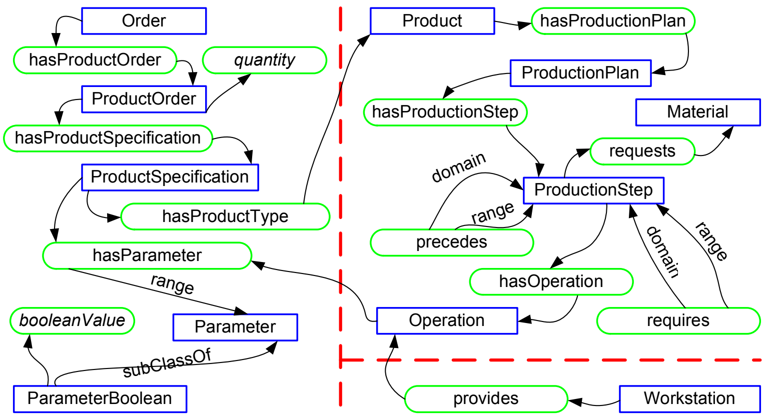

Fig. 1. Parts of ontologies for product orders, for production plans, and for production equipment; only selected OWL classes and relations are shown

change – it is only needed to extend ontologies (i.e., create subclasses of existing classes) to cover new products, new product parameters, new operations if needed, and to provide product plan describing the process of making the new product.

A segment of the proposed ontologies is shown in Fig. 1. The `Order` may consist of several `ProductOrders` that are specified by `ProductSpecification`. The specification of product parameters is expressed using subclasses of `Parameter` class, such as `ParameterBoolean` to specify Boolean values. These subclasses are usually product specific expressing for instance color, shape, material used, etc. Each `Product` has its own generic `ProductionPlan` expressed in a knowledge base, which is based on the extended production plan ontology.

The `ProductionPlan` consists of `ProductionSteps`. The sequence of `ProductionSteps` can be partially ordered by `precedes` relation. There is also possibility to specify required steps via `requires` relation, which is important for customization of the plan for a particular ordered product. Each `ProductionStep` has an `Operation` to be performed in this step together with possible `Parameters` that specify details of the `Operation`. The `Operation` is provided by `Workstation` that is a logical group of equipment (see next Sect.).

3 Integration of Manufacturing Ontology in Distributed Control System

This section provides general overview of the deployment of the proposed manufacturing ontology in distributed, agent-based manufacturing control applications. Each agent uses a specific piece of the proposed ontology for representing and processing the information related to the status and control of the manufacturing process. As well, the ontology is used for expressing the semantics of knowledge and data exchanged between the agents.

One of the basic elements of proposed framework is the concept of a *workstation*, considered as a logical composition of physical manufacturing devices and equipments, like machines, operators, buffers, etc. The workstation is represented by an autonomous control component – Workstation Agent (WA) – that can negotiate about the allocation of its advertised resources with the agents that control the execution of the production process. The single operation provided by a workstation is usually internally decomposed into the execution of several suboperations carried out by the particular equipment. Such an execution is supervised and controlled by the workstation agent by negotiations with the subordinate equipment agents. There are entry and exit points of the workstation where the transportation system is linked to and through which material, semi-products and products can be delivered into and out of the workstation.

Another key element in the proposed framework is the Product Agent (PA). In contrary to the usual comprehension of a product being a passive entity that never communicates or decides, the Product Agent implements the concept of an "active" or "intelligent" product. An active product is an intelligent entity associated with the physical product, which is able to perceive and act on the real state of the system [21]. It manages the information about itself and makes decisions relevant to its own life – it actively participates in negotiations with other agents about resource allocation and self-routing, it reports about its status and problems to the owner, etc. [16]. This definition, where one instance of PA corresponds to one instance of physical product contrasts to the definition of a *product holon* from the well known PROSA architecture. In PROSA the product holon contains the product model of the product type, not dealing with the state of a particular physical product instance being produced [22].

The other agents of the proposed architecture and their communication links are depicted in Fig. 2. The following sequence of steps takes place from receiving the order up to its fulfillment (numbering corresponds to labels of arrows in Fig. 2):

1. An order comes to the control system via a special Order system component, which in fact can also be an agent. It is supposed that the Order system has capabilities of receiving orders in different formats from different entities like human users through HMI, or from computer applications at the customer site sent over a computer network. The Order system converts the order parameters into an RDF format [14] using the OWL order ontology shown in Fig. 1.

2. The Order system creates a new instance of the Order Agent (OA) and passes it the RDF order specification. In this example, the order is composed of two product orders (see the association in Fig. 2), each for a particular type of product.

3. OA creates a new instance of PA for each product order. The PA is given only that part of the RDF order specification that corresponds to the product order. In this example, ProductAgent1 will take care of the fulfillment of ProductOrder1 and ProductAgent2 will process the ProductOrder2, respectively.

4. In order to obtain the instructions of how to make the product, the PA contacts the Production Plan Agent (PPA). This agent is supposed to manage and provide the instances of production plans for different product types implemented on the basis of general production plan ontology. Thus, this agent should not be confused with the PPA agent of the ExplanTech architecture [18] that schedules the production and thus corresponds rather to our PA agent. To select the proper plan, the PPA follows the `hasProductType` relation in the order specification and subsequently selects the plan associated with the `Product` by the `hasProductionPlan` relation (see Fig. 1).

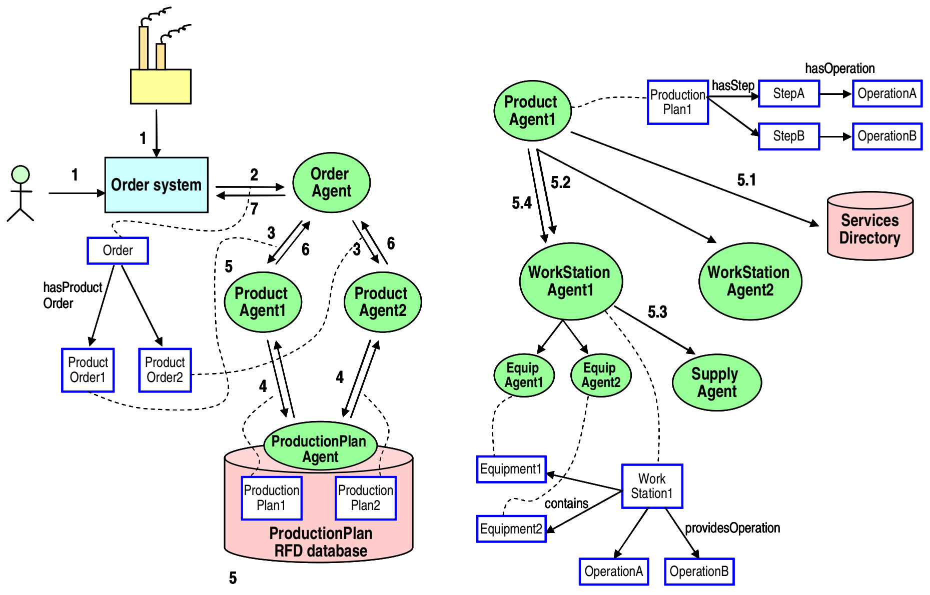

Fig. 2. Architecture for ontology-based multi-agent production control system

It is supposed that such a plan instance describes in general all the steps and operations needed to make the final product. However, such a plan has to be modified according to the order specification. This is basically done by PPA by assigning the parameter values from the order to parameters of operations in the plan or by deleting some production steps if for instance particular product feature is not required. Such a modified plan tailored to the specific product is returned back to the PA who can start to process it.

5. The PA processes the production plan ontology instance in terms of planning the execution of particular production steps (in the example depicted on Fig. 2 there are two steps: `stepA` and `stepB`). Planning of a single step consists of three main phases.

Planning phase. 5.1 The PA agent queries the Service Directory (usually the DF agent) to get the list of workstations that provide operation associated with the step. The association of the operation to step is through the `hasOperation` relation (see both Fig. 1 and 2). In this example, there is `operationA` associated with `stepA` and `operationB` associated with `stepB`, respectively.

5.2 The PA asks Workstation Agents to bid on the nearest available time slot when the operation could be done. The WA has to take into account the time needed to transport all required materials and semi-products from their current location to the workstation (see next step) as well as the time required for the operation itself.

5.3 The WA agent asks its Supply Agent (there is one instance of SA for each workstation) to plan the retrieval and transport of all material(s) and semi-product(s) required as an input to the production step. Some of the material can be already available in the workstation; some of them need to be transported from other work stations. The SA thus contacts all workstations providing materials asking for bidding on the time of delivery of required pieces of material.

Commit phase. 5.4 The PA sends request to such WA that offers the best bid (i.e., nearest time slot) for allocating previously offered resources. The WA subsequently requests SA to commit the retrieval and transport of materials.

Execution phase. In the last phase the PA asks the WA to start execution of the step. The execution phase for this step can be started right after all preceding steps are finished (Plan and Commit phases could be done in advance). The WA requests SA to execute the transport of all materials and semi-product(s). When all transport is done, the WA starts execution of the operation itself, by coordinating its subordinate agents. When the operation has finished (or even sooner), the PA starts planning the next step defined in the production plan. For this three-phase negotiation we have developed a Commit-Plan-Execute protocol, which is in detail described in [11].

6. When all the operations listed in the plan are done the PA informs the OA about finishing the product.

7. After the notification from all PAs the OA informs the Order system about fulfillment of the order.

The functionality inside the workstation can be advantageously described using the same concepts of the proposed ontology. Each of the workstation's externally advertised operation can have its own production plan instance associated. The workstation agent can obtain such plans from the PPA by asking it for all plans where the operations match those provided by the workstation's equipment.

4 DIAL Packing Cell Case Study

In this section we report on the latest developments related to the integration of ontologies in multi-agent manufacturing control applications. The platform that is currently being semantically extended is the Manufacturing Agent Simulation Tool (MAST). This agent-based automation control solution developed by Rockwell Automation provides means for design, simulation and real-world validation of a distributed control system based on intelligent agents [23]. Although named "intelligent", the agents contained in the MAST system can be viewed, in the semantic context of this paper, being rather very "simple". It is mainly because of the fact that semantic technologies were not as enhanced in a year 2000 when the development of MAST began as they are today. There were FIPA standards, newly emergent agent platforms like JADE and FIPA-OS and the XML as the latest technological advancements available those days. So, the communication of agents is based on XML messages, which semantic interpretation is not described explicitly, but is tightly embedded in the agent program code.

Recently, we have reimplemeted the MAST agents in such a way that the communication, representation of knowledge and reasoning is based on ontologies. Additionally, we have provided a flexible framework for explicitly defined behavioral modules [19], which can be dynamically modified in the agent (agent can receive a new behavior module in binary form via message) as the ontology is being extended or updated (see more in Sect. 5). The main objective for this new, semantic-based realization is the creation of a semantical description of the production process using the manufacturing ontologies described in Sect. 2., and integration of the support for ontologies in agents taking part in production planning and control.

4.1 The DIAL Packing Environment and Non-semantic Agent Control Solution

The packing and assembly environment installed in DIAL (Distributed Information and Automation Laboratory), formerly known as CDAC, at Cambridge University's Institute for Manufacturing [8] has been previously used as a real-world platform for verification of the MAST agent-based control principles [9]. The prototype production includes packing of gift boxes that contain arbitrary combination of three cosmetic items, which the user specifies in his/her order by selecting from four different types (gel, razor, deodorant and foam). These items are stored in a vertical, four-slot storage unit, from where they are picked up by a Fanuc M6i robot. The line contains two of such independent production stations (marked as Workstation1 and Workstation2 in Fig. 3) and one Automated Storage and Retrieval System (ASRS) composed of a portal crane and three rack storages (marked as Workstation3 in Fig. 3). The transportation between the workstations is provided by a monorail conveying system with shuttles carrying boxes. As can be seen in Fig. 3 there is one main horizontal transportation loop servicing ASRS and two vertical loops for reaching the production stations.

For the agent-based simulation of the DIAL testbed the MAST system has been extended with a set of new agents controlling the behavior of specific equipment. The storage unit agent maintains the list of available raw items, the Fanuc robot agent controls pick-and-place operations of the robot, the rack storage agent provides empty boxes and the gantry robot agent controls the movement of boxes by the portal crane. The task of the order agent, automatically created with each new order, is to find a suitable product agent that will actually control the production process. For simplicity of implementation, the already available shuttle agent controlling the movement of a shuttle has been extended by the product agent-like behavior. This agent negotiates about elementary operations with individual resource agents. First, it negotiates with rack storage agents to obtain an empty box and subsequently with the gantry robot agent to pick the box out of rack and place it on the shuttle. Then, for each item to be packed it negotiates with the storage units to find the one that can supply the required item and subsequently, after shuttle with box reaches the production station, asks robot to pack the item to the given slot in box. When all slots are filled the finished product returns to ASRS what involves negotiation with rack storage agents again to provide an empty place for final storage. The negotiations are based on the contract-net protocol in which the list of participants requested to bid on specific operation is obtained from the Directory Facilitator that manages the registration of available agent services.

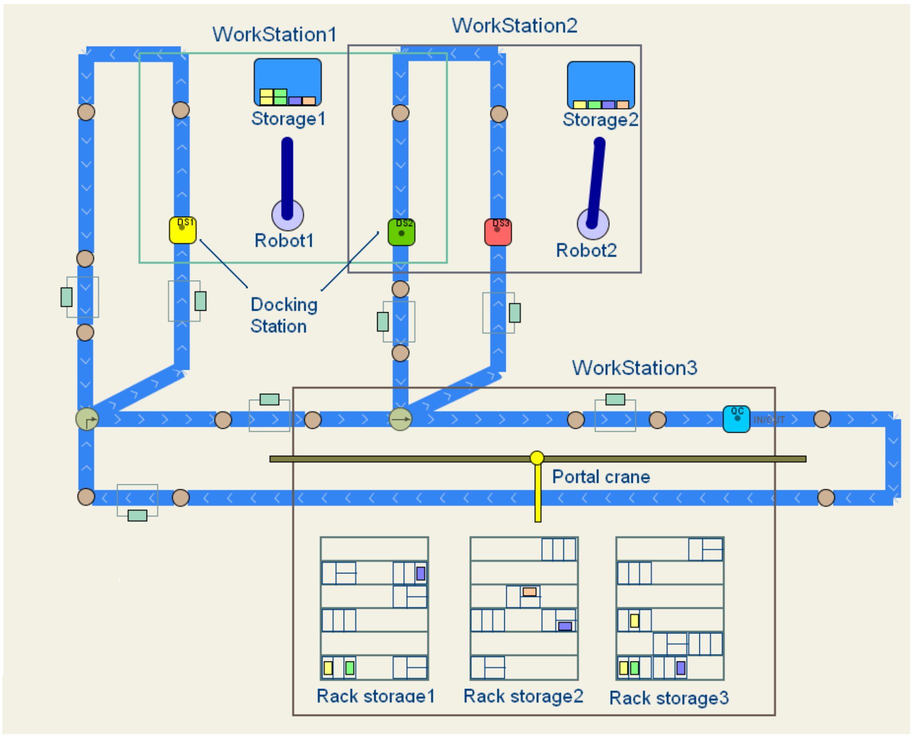

Fig. 3. Prototype packing line of University of Cambridge's DIAL lab simulated in MAST system

4.2 Ontology-Based Implementation of DIAL Scenario

To demonstrate the usability of the developed manufacturing ontology presented in Sect. 2, the DIAL scenario-specific extension has been created on top of the general concepts provided by the ontology. There is a new `PackedBox` class extending the general `Product` class that is used to select corresponding production plan. Additionally, there is `BoxItemType` class representing the type of item inserted into the box with subclasses `Deodorant`, `Gel`, `Razor` and `Foam`. These subclasses are used in the specification of order as parameters specifying what type of item is requested for each slot in a box. Furthermore, the production plan for making the gift box has been created – it is illustrated in Fig. 4. It is composed of five steps; the first one is for retrieving an empty box from ASRS (associated with `BoxRetrieving` type of operation); next three steps, which can be executed in any order, concerning inserting items into particular slots in the box (operation `BoxItemPacking`) and in the last step the finished box is stored back into the ASRS (operation `BoxStoring`).

The orders are placed through the OrderManager component in the graphical user interface. Its task is to compose a semantic description of the order based on parameters specified by the user in the form of an RDF description compatible with the order ontology. The order agent created for the order then instantiates new product agent.

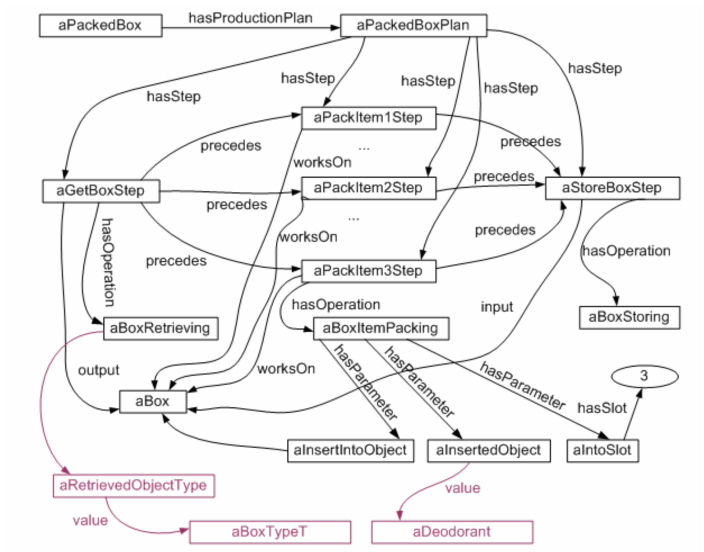

Fig. 4. Ontology describing production process (box packing) in DIAL scenario. At the bottom there are schematically shown parameters (aBoxTypeT and aDeodorant) copied from user order when Production Plan Agent prepares tailored production process description.

The product agent receives the order specification from the order agent. The product agent then asks a special Production Plan Agent for an instance of production plan ontology for box packing. This agent modifies the general production plan according to the user's order. This modification includes association of parameters from the order to particular production steps (type of box and types of inserted items – can be seen at the bottom of Fig. 4). Optionally, deletion of some of the three packing steps is done in case particular slot is requested to be empty.

According to this production description, the product agent plans execution of these steps, i.e. searches the Directory Facilitator for agents providing operations associated with steps (`BoxRetrieving`, `BoxItemPacking` and `BoxStoring`). The cost-based model, extended contract-net protocol [11] and RDF content language are used for negotiation about resource allocation between the product agent and the workstation agents. In bidding on cost of providing operation the workstation agent considers the availability of raw material (boxes and cosmetic items), capacity and load of workstation and product priority. The product agent selects the lowest cost and subsequently negotiates with the transporter agents representing shuttles about the delivery of product to the selected destination. Once the operation is finished the product agent starts planning of the next step defined in the plan ontology.

5 Key Characteristics and Advantages of Combining Agents with Semantics

The shift from non-semantic multi-agent control systems to the next generation of semantically enriched solutions incorporating ontologies for representing and sharing knowledge can be compared to the transition from classical centralized control to distributed agent-based systems. This section summarizes some key aspects of combining agents with semantic technologies and discusses potentials and advantages of such an integration.

Duality of knowledge. It has been discussed that the ontology provides a semantic description of the particular domain knowledge, like for instance manufacturing. It defines basic concepts or facts, like machines and operations and captures relations between those concepts like for instance capturing the fact that a machine provides an operation. The ontology thus provides a *declarative* description of the knowledge. It is used by an agent to give the semantic meaning to the data the agent obtains by perceiving the environment or by receiving from other agents via messages. The knowledge about some domain has to be however considered as twofold. Especially in the applications where the agents act in a real environment, like they are controlling machines in factories, the agent has to know not only *what does it mean* but also *what to do* when a new observation is done. In other words there might be an action required to be taken as a response to a new information in agent's repository. In fact, this is also a king of knowledge, although represented by an algorithm, procedure or process to be executed. Thus we call this *procedural knowledge* and argue it is necessary to interlink it with the declarative one. This dual character of knowledge is not discussed so much in the literature. Rzevski et al. in [20] present the Magenta agent toolkit where the ontology provides also an associated procedural description of the agent behavior.

Modularity. The usual concept of inheritance coming from object oriented programming is deployed when building the ontologies. It means that the existing classes representing general entities are used to create subclasses describing more specialized entities (like drilling being a subclass of an operation). This concept is naturally used to build a hierarchy of domain or process specific knowledge modules, describing for instance production planning and scheduling aspects, material routing, ordering, machining, etc. A key point is that an agent, designed for a particular task, does not need to be equipped with the overall ontology; it gets only an appropriate set of knowledge modules that are needed to describe the agent's domain of expertise. Complementary to the a tailored subset of ontology, i.e., the declarative knowledge, the agent should also be provided with a corresponding subset of procedural knowledge. This mechanism allows creating minimal, tailored agents in a modular way, having only necessary amount of knowledge and functionality. This aspect is important for instance when considering deploying agents on embedded devices or controllers with strict memory and computing power constraints.

We have developed a pilot implementation of these principles in MAST agents. As presented in more detail in [19], the procedural knowledge is captured in the modules of behaviors programmed and stored in separate Java classes held apart from the main agent body. When the agent is instantiated it is given the tailored subset of these modules needed for its common functionality. When the agent notes an unusual event for which it does not have appropriate behavior it can obtain it on-line from dedicated Behavior Repository agent(s). This agent manages the database of behavior modules for different types of agents and on request sends these behaviors to the agents in a binary form. This mechanism can be also used for sending updates of existing behaviors without a need to stop the running agents and reprogram them off-line. The Magenta toolkit presented in [2] also allows sending updates of agent's behavior (program code) at runtime as the ontology is being modified or extended.

We envision that eventually the agents will not be programmed like today, where there are specific agent classes representing specific manufacturing components or tasks like conveyor belt agent, diverter agent or product agent. Instead, there will be one agent class representing a general agent skeleton. When instantiated, the agent downloads, based on its type, all the needed ontology as well as behavior modules.

Flexibility. Even if we will not consider modifying or updating agent program code at runtime, the use of ontologies increases the flexibility of the agent-based control system. In the packing cell scenario described in Section 4, the previously implemented product agent controlling the packing of box had its behavior fixed to this particular product type. Although it could discover new operation providers added to the system or find alternative routing in case of conveyor failures, the same product agent class could not be used for controlling different type of production process, like for instance box unpacking. In the new, semantically enriched solution, the production process is not described in the program code of the agent, but it is specified explicitly using the ontology. The agent is then able to process such a recipe automatically, in such a way it schedules the execution of steps by negotiating with workstation agents about providing the operations associated with the steps. The great advantage is that the same product agent can be simply used to control any kind of production process, if it is described in the same ontology. In the packing cell scenario the plan is to develop the RDF description of the box unpacking process and use the existing product agent to fulfill the order for unpacking a box.

6 Conclusions

The main objective of the semantic extension of the agent-based control that is discussed in this paper is to design a coherent framework providing general and extendable ontologies for semantic description of manufacturing tasks. In addition, the instructions for integrating these ontologies in agent solutions for discrete production control are provided. The agent simulation system MAST together with the DIAL packing line has been selected as platforms for verification of the developed framework. Compared to previous non-semantic solutions the major advantage of the new ontology-based system is the introduction of a general product agent that does not have behavior fixed to any concrete product type. It is able to control production of any product type following the semantic description of the production plan. In future work, the focus will be on implementation of a general workstation agent (currently it is still fixed to particular equipment) that is able to govern the functionality of arbitrary aggregation of equipment. It also uses the production plan ontologies defining how the externally advertised operation breaks down into elementary actions executed by the equipment. Other plans include using ontologies and rules for basic reasoning of agents instead of pure Java code, for example for matchmaking, for reasoning in transportation or for generation of tailored production plan from general one based on customer order parameters.

Acknowledgements

This research has been supported by Rockwell Automation, by the Ministry of Education of the Czech Republic within the Research Programme No.MSM6840770038: Decision Making and Control for Manufacturing III and by the FIT-IT: Semantic Systems program, an initiative of the Austrian Federal Ministry of Transport, Innovation, and Technology (bm:vit) under contract FFG 815132. The authors would like to thank to Prof. Duncan McFarlane, the head of the Distributed Information & Automation Laboratory within the Institute for Manufacturing, University of Cambridge for his support and cooperation.

References

1. Al-Safi, Y., Vyatkin, V.: An Ontology-Based Reconfiguration Agent for Intelligent Mechatronic Systems. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 114–126. Springer, Heidelberg (2007)
2. Andreev, V., Rzevski, G., Skobelev, P., Shveykin, P.: Adaptive Planning for Supply Chain Networks. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 215–224. Springer, Heidelberg (2007)
3. ANSI/ISA-88.01.1995. Batch Control Part 1: Models and Terminology. American National Standard. The Instrumentation, Systems and Automation Society (1995)
4. Brennan, R., Vrba, P., Tichy, P., Zoitl, A., Sünder, C., Strasser, T., Mařík, V.: Developments in Dynamic and Intelligent Reconfiguration of Industrial Automation. Computers in Industry 59/6, 533–547 (2008)
5. Camarinha-Matos, L.M.: Multi-Agent Systems In Virtual Enterprises. In: Proceedings of International Conference on AI, Simulation and Planning in High Autonomy Systems, pp. 27–36. SCS publication, Lisbon (2002)
6. Cândido, G., Barata, J.: A Mutliagent Control System for Shop Floor Assembly. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 293–302. Springer, Heidelberg (2007)
7. Dean, M., Schreiber, G.: OWL Web Ontology Language reference (2004), `http://www.w3.org/TR/owl-ref/`
8. Fletcher, M., Brusey, J.: The Story of the Holonic Packing Cell. In: Proceedings of 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems. ACM Press, Melbourne (2003)
9. Fletcher, M., Vrba, P.: A Brace of Agent Simulation Scenarios. In: Proceedings of IEEE Workshop on Distributed Intelligent Systems, pp. 169–176. Prague (2006)
10. Foundation For Intelligent Physical Agents: FIPA ACL Message Structure Specification, SC00061G (2002)
11. Kadera, P., Tichy, P.: Plan, Commit, Execute Protocol in Multi-agent Systems. In: HoloMAS 2009, Linz (submitted, 2009)
12. Lemaignan, S., Siadat, A., Dantan, J.-Y., Semenenko, A.: MASON: A proposal for an ontology of manufacturing domain. In: IEEE Workshop on Distributed Intelligent Systems, pp. 195–200. IEEE Computer Society Press, Los Alamitos (2006)
13. Lopez, O., Martinez Lastra, J.L.: Using Semantic Web Technologies to Describe Automation Objects. International Journal of Manufacturing Research 1(4), 482–503 (2006)
14. Manola, F., Miller, E.: RDF primer (2004), `http://www.w3.org/TR/rdf-primer/`

15. McLean, C., Lee, Y., Shao, G., Riddick, F.: Shop Data Model and Interface Specification, NISTIR 7198 (2005)
16. Meyer, G., Främling, K., Holmström, J.: Intelligent Products: A survey. Computers In Industry 30(3), 137–148 (2009)
17. Obitko, M., Vrba, P., Mařík, V., Radakovič, M.: Semantics in Industrial Distributed Systems. In: Proceedings of the 17th IFAC World Congress, pp. 13880–13887 (2008)
18. Pěchouček, M., Rehák, M., Charvát, P., Vlček, T., Kolář, M.: Agent-Based Approach to Mass-Oriented Production Planning: Case Study. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews 37(3), 386–395 (2007)
19. Radakovič, M., Vrba, P., Obitko, M.: Architecture for Explicit Specification of Agent Behavior. In: 13th IFAC Symposium on Information Control Problems in Manufacturing, Moscow, Russia (accepted 2009)
20. Rzevski, G., Skobelev, P., Andreev, V.: MagentaToolkit: A Set of Multi-agent Tools for Developing Adaptive Real-Time Applications. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS, vol. 4659, pp. 303–313. Springer, Heidelberg (2007)
21. Sallez, Y., Berger, T., Tretensaux, D.: A Stigmergic Approach for Dynamic Routing of Active Products in FMS. Computers in Industry 30(3), 204–216 (2009)
22. Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeter, P.: Reference Architecture for Holonic Manufacturing Systems: PROSA. Computers in Industry 37(3), 255–274 (1998)
23. Vrba, P., Mařík, V.: Simulation in Agent-Based Control Systems: MAST Case Study. In: Proceedings of 16th IFAC World Congress, Prague (2005)
24. Vyatkin, V., Christensen, J., Lastra, J.: OOONEIDA: An Open, Object-Oriented Knowledge Economy for Intelligent Industrial Automation. IEEE Transactions on Industrial Informatics 1(1), 4–17 (2005)