Towards a Complex Interaction Scenario in Worker-cobot Reconfigurable Collaborative Manufacturing via Reactive Agent Ontology

Case-study: Two Workers in Cooperation with One Cobot

Ahmed R. Sadik[1,2] and Bodo Urban[1,2]
[1]*University of Rostock, Universitätsplatz 1, 18055, Rostock, Germany*
[2]*Fraunhofer Institute for Computer Graphic Research IGD, Joachim-Jungius-Str. 11, 18059, Rostock, Germany*

Keywords: Ontology-based Communication, Collaborative Robotics, Reconfigurable Manufacturing System, Holonic Control Architecture, Autonomous Reactive Agent, Multi-Agent System.

Abstract: Close Human-Robot Interaction (HRI) has been a great focus of research for the last decades. The outcomes of this focus is a new field in industrial robotics called collaborative robotics. A collaborative robot (cobot) is usually an industrial robot designed to operate safely in a shared work environment with the human worker. This in contrast to conventional Industrial Robots (IRs) which are operating in isolation from the worker workspace, the cobot is changing the concept of automation from fully automated operations to semi-autonomous operations, where the decisions of the worker will influence the actions of the cobot and vice-versa. Therefore, a communication and information control framework must exist to connect the worker and the cobot together to fulfil this semi-autonomous paradigm. This framework should be able to provide a method to represent the common knowledge which can support the collaborative manufacturing between the worker and the cobot. During this research we are proposing an ontology-based Holonic Control Architecture (HCA) as a proper solution to share and communicate the knowledge needed to achieve complex interaction scenarios between the worker and the cobot.

1 INTRODUCTION

There is no doubt that the collaborative robotics is an innovative solution for the Reconfigurable Manufacturing System (RMS). An RMS is by definition a system where production components and functions can be modified, rearranged and/or interchanged in a timely and cost-effective manner to quickly respond to production requirements (Koren et al., 1999). Three important factors are usually defining the reconfigurability of an RMS, these factors are the production line structure, the product building plans, and the shop floor resources functions. Adding the close physical interaction between the worker and the cobot, is definitely a new important factor in the RMS which we should put more focus of study on it. The RMS concept stands between the Dedicated Manufacturing System (DMS) and the Flexible Manufacturing System (FMS) as shown in Figure 1. Mass production method is applied in the DMS where a high rate of production can be achieved.

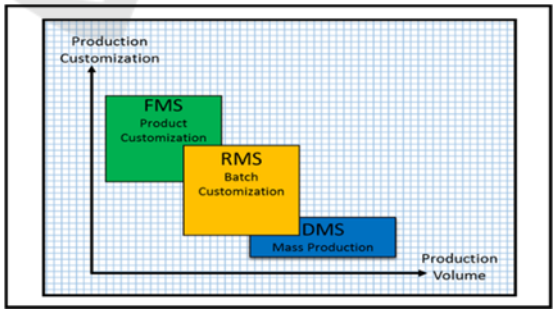

Figure 1: Comparison between DMS, RMS and FMS.

Mass production is using a static pre-planned schedule to manage the production operations, therefore the customization level is very low or impossible. Mass production method is cost efficient only in case of high market demand, which is not a feature of nowadays market (Elmaraghy, 2005). The

Sadik A. and Urban B.
Towards a Complex Interaction Scenario in Worker-cobot Reconfigurable Collaborative Manufacturing via Reactive Agent Ontology - Case-study: Two Workers in Cooperation with One Cobot.
DOI: 10.5220/0006487200270038
In *Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KEOD 2017)*, pages 27-38
ISBN: 978-989-758-272-1

FMS applies the product customization concept where very highly customized products can be manufactured in small quantities. The cost of the FMS is relatively high due to using expensive machines and sophisticated software. Therefore the production rate and capacity of an FMS is very low compared with a DMS, which makes the FMS fails in the most of the cases to efficiently respond to the market demands (Kruger, 2015). Batch customization production method is used in the RMS where the production of several distinct versions of the same product over the same production line is possible. Batch customization method offers less customizability than product customization, therefore a higher production rate and capacity can be obtained. That makes the RMS a compromised solution between the DMS and the FMS.

The rest of this article is structured as follows. In section 2 we address the challenges in the complex worker-cobot interaction under the umbrella of reconfigurable manufacturing. Autonomous reactive agent will be discussed in details in section 3, we will focus in particular on agent communication via ontology as it will be the ground of building the solution concept in section 4. Furthermore, section 5 shows a case-study implementation of this concept. Finally, section 6 will wrap up the work summary with the conclusion and the future research.

2 COMPLEX INTERACTION CHALLENGES

The cooperation between the worker and the cobot in a reconfigurable manufacturing context can be seen as a complex information interaction system for the following reasons (Sadik and Urban, 2017):

- In contrast to a simple interaction scenario which uses basic data types such as Strings, Boolean, or Integers, a complex interaction scenario not only needs to exchange information in form of objects, but also needs to express the relations between these objects.
- In a worker-cobot complex interaction scenario, it is needed to provide the meta-data required for this cooperation. Descriptive meta-data is needed to give a meaning of the tasks and the operations which can be done during the cooperation. Structural meta-data is required to indicate how to compound new objects from existing objects. Finally, administrative meta-data is required to control the task assignment and the cooperation planning, management, and execution.
- In contrast to object serialization convention which translates the data structure into a stream of bytes to be transferred and stored, a worker-cobot complex interaction scenario requires a human readable representation language to perform the same purpose of serialization. In other words, a language which can be understood by both the worker and the cobot should be used to describe the shared work environment.
- The variation in the required production customization level and rate must be comprehended by the communication and information control system. As the complex interaction between the worker and the cobot will be build based on this comprehension.
- In case of one cobot is in cooperation with two workers, the cobot is considered to be a shared operational resources. Considering the fact that the required time from the worker to finish a specific task is always varying, this means that the interaction should be able to tolerate this time variation.

Therefore during this article we focus on explaining the possible approach and technology which is able to overcome the above challenges.

3 AUTONOMOUS AGENT ONTOLOGY

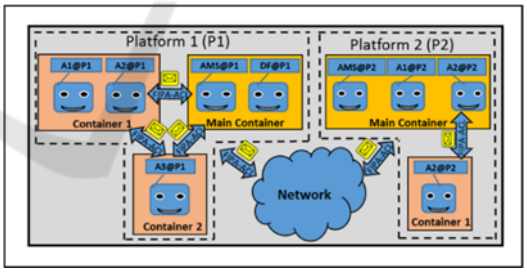

Figure 2: JAVA Agent Development.

A software agent is a computer system situated in a specific environment that is capable of performing autonomous actions in this environment in order to meet its design objective (Jennings and Wooldridge, 1998). An agent is autonomous by nature. It means that an agent operates without a direct intervention of the humans, and has a high degree of controlling its actions and internal states. In order to achieve this autonomy, an agent must be able to fulfil the following characteristics:

- Responsive: an agent is capable of perceiving its environment and respond in a timely fashion to

the changes occurring in it.

- Pro-active: an agent is able to exhibit opportunistic, goal directed behaviour and take initiative.
- Social: an agent can interact with other artificial agents or humans within its environment in order to solve a problem.

Conceptually, an agent is a computing machine which is given a specific problem to solve (Shen et al., 2006). Therefore, it chooses certain set of actions and formulates the proper plans to accomplish the assigned task. The set of actions which are available to be performed by the agent are called a behaviour. The agent behaviours are mainly created by the agent programmer. An agent can execute one or more behaviour to reach its target. The selection of an execution behaviour among others would be based on a certain criteria defined by the agent programmer. Building an execution plan is highly depending on the information which the agent infers from its environment including the other agents. A Multi-Agent System (MAS) is a collective system composed of a group of artificial agents, teaming together in a flexible distributed topology, to solve a problem beyond the capabilities of a single agent.

JAVA Agent Development Environment (JADE) is a distributed MAS middleware framework as it is shown in Figure 2 (JADE, n.d.). Each JADE instance is an independent thread which contains a set of containers. A container is a group of agents run under the same JADE runtime instance. Every platform must contain a main container. A main container contains two necessary agents which are: an Agent Management System (AMS) and a Directory Facilitator (DF). AMS provides a unique ID for every agent under its platform to be used as an agent communication address. While the DF announces the services which agents can offer under its platform, to facilitate the agent services exchange, so that every agent can obtain its specific goal (Teahan, 2010). JADE applies the reactive agent architecture which complies with the Foundation for Intelligent Physical Agent (FIPA) specifications, and provides a graphical interface to deploy and debug a MAS (Bellifemine et al., 2007). FIPA is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies (FIPA, n.d.). JADE agents use FIPA-Agent Communication Language (FIPA-ACL) to exchange messages either inside its own platform or with another platform in a distributed MAS as shown in Figure 2 and Figure 3.

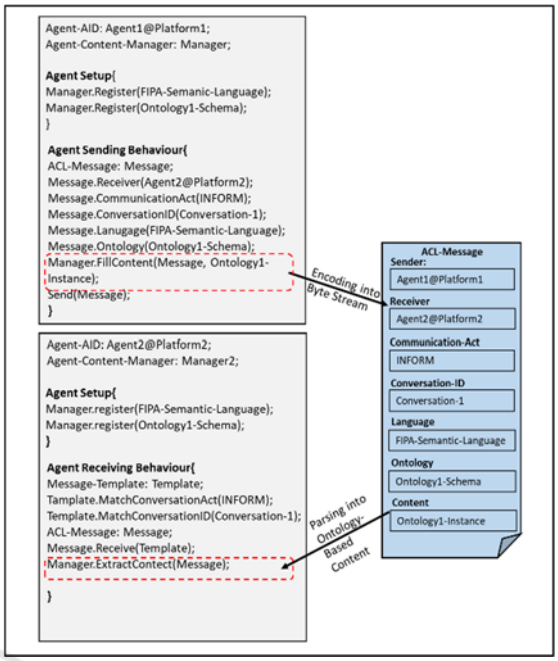

Figure 3: JADE Agents Communication via FIPA-Ontology.

The interaction mechanism for sending and receiving an ontology-based ACL-message between two JADE agents is illustrated in Figure 3. Every JADE agent must have an Agent Identification (AID). The AID is composed of a unique name for the agent over a specific platform, and it is used as an address for the agent to send or receive a message (Poslad, 2007). Also every agent must have a setup method. The setup method is automatically triggered after the agent creation. The main function of the setup method is to initialize the required parameters and behaviours needed for the agent to perform its tasks. In order to complete a communication process between two agents, one agent must have a behaviour which is responsible for constructing and sending an ACL-message, the other agent must have a behaviour which is responsible for receiving the ACL-message. An ACL-message is composed from variety of fields, and it must have at least a Sender and a Receiver AIDs. Other fields such as the Communication-Act and the Conversation-ID are necessary to distinguish the message at the receiver side. The Communication-Acts are defining the ACL-message in terms of standard FIPA actions or functions, for instance a communication-act field can contain INFORM, REQUEST, CONFIRM, etc. The Conversation-ID parameter can be any unique string to distinguish or record a specific conversation topic or thread among the agents.

The content field of an ACL-message contains a String data type in case of simple agent conversation. However in a complex agent conversation, an ontology-based content will be the proper conversation method. In order to communicate via agent ontologies, every agent must deploy a content-manager. The content-manager registers a common language of conversation between the agents (Caire, 2009). The FIPA Semantic Language (FIPA-SL) is not mandatory but preferable in a complex JADE conversation. FIPA-SL is a human readable language which defines the syntactic rules needed to parse or encode an ontology-based content. Also the content-manger registers the common ontology schemas. The ontology schema is defining the abstract structure pattern and the semantics needed to construct or interpret an ontology-based ACL-message. At the sender agent side, the content-manager checks the semantics of the sent ACL-message based on a common ontology schema, and decodes that message into a stream of Bytes via FIPA-SL. At the receiver side, the content-manager parses the received ACL-message into a human readable content via FIPA-SL, and structures it based on a common ontology schema.

4 SOLUTION CONCEPT

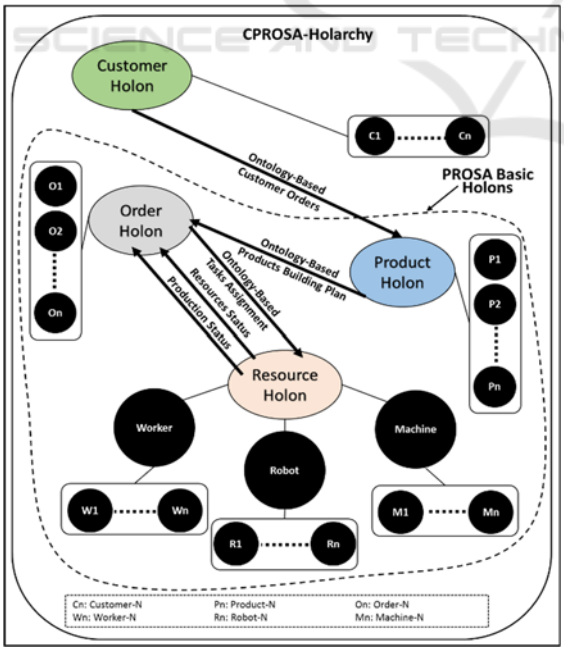

Figure 4: CPROSA-holarchy.

In the late sixties, the term holon has been introduced for the first time by philosopher Koestler (Koestler, 1967). Koestler developed the term as a basic unit in his explanation of the evolution of the biological and social structures. Based on his observations that organisms (e.g., biological cells) are autonomous self-reliance units, which have a certain degree of independent control of their actions, yet they still subject to higher level of control instructions. His conclusion was that any organism is a whole “holos” and a part “on” in the same time, which derived the term holon (Botti and Giret, 2008). The concept of holon has been adopted in the early nineties by the intelligent manufacturing systems (IMS) consortium, to define a new paradigm for the factory of the future. The following terminologies has been defined by the IMS to provide a better understanding of the Holonic Control Architecture (HCA):

- Holon: an autonomous cooperative building block of the manufacturing system that can be used to transform, transport, store and/or validate the information and the physical signals (Babiceanu and Chen, 2006).
- Autonomy: the capability of the holon to create and control the execution of its own plans and/or strategies.
- Cooperation: a process whereby a set of holons develop mutually acceptable plans and execute these plans together.
- Holarchy: a system of holons which cooperate to achieve a goal or objective. The holarchy defines the basic rules for cooperation of the holons and thereby limits their autonomy.

The HCA is basically a distributed control and communication topology which divides the manufacturing process tasks and responsibilities over different holon categories. Product-Resource-Order-Staff-Architecture (PROSA) is the most known HCA model (Van Brussel et al, 2003). The PROSA model defines three basic holons which can be seen in Figure 4. The PROSA holons are as following:

- Product Holon (PH): is responsible for processing and storing the different production plans required to insure the correct manufacturing of a certain product.
- Order Holon (OH): is responsible for composing, managing the production orders. Furthermore, in a small scale enterprise, it should assign the tasks to the existing operating resources and monitor the execution status of the assigned tasks.
- Operational Resource Holon (ORH): is a physical entity within the manufacturing system, it can represent a robot, machine, worker, etc. The ORH

is usually composed of two components. The first component is the physical component which represents the physical input/output (I/O) of a resource. The second component is the communication component which is responsible for translating the I/O events into information and conducting them to the other holons and vice-versa.

Another extra holon can exist in the PROSA model which is the Supervisor Holon (SH). The SH will be implemented in a large scale enterprise as its main function is to coordinate with the other SHs for scaling and expanding the manufacturing system. In the context of reconfigurable manufacturing we found that it is necessary to modify the PROSA model (Sadik and Urban, 2016). Therefore, a new holon is introduced by this research which is the Customer Holon (CH). A CH is deployed on the customer platform to provide a User Interface (UI) for the customer to select and customize the product order. Furthermore, it interacts with the PH to trigger a new production order. Therefore, we are going to refer to our PROSA modified holarchy as CPROSA.

While the HCA is a conceptual model focuses on the holons functionalities and responsibilities, it does not specify a certain technology to apply that concept. On the other hand, artificial agent technology is a general purpose solution which can apply the HCA. Thus, during this research, JADE agent framework has been used to implement the HCA. JADE empowers the object oriented concepts such as abstraction and inheritance, which makes it very suitable for applying the HCA. For example, a Worker Holon (WH) can have many different instances originate from it, yet every WH instance can act differently than the others. Figure 4 illustrates the main concept to implement a reconfigurable cooperative manufacturing workcell, which can contain different workers and cobots as operational resources, the manufacturing workcell can simultaneously process different customized orders from variety of customers.

Using JADE to implement the reconfigurable cooperative manufacturing workcell empowers another very strong concept which is the agent communication via ontology (Alsafi and Vyatkin, 2010). As it has been discussed earlier, HCA is a practical solution for the reconfigurable cooperative manufacturing workcell (Balakirsky, 2015). The HCA implementation includes many different objects. This is not only obligating the agents to send or receive objects, but also it obligates them to express relations between these objects and perform actions over them (Fiorini et al., 2015), which brings us to use the concept of ontology to communicate between the agents. Examples of these communication can be seen in Figure 5.

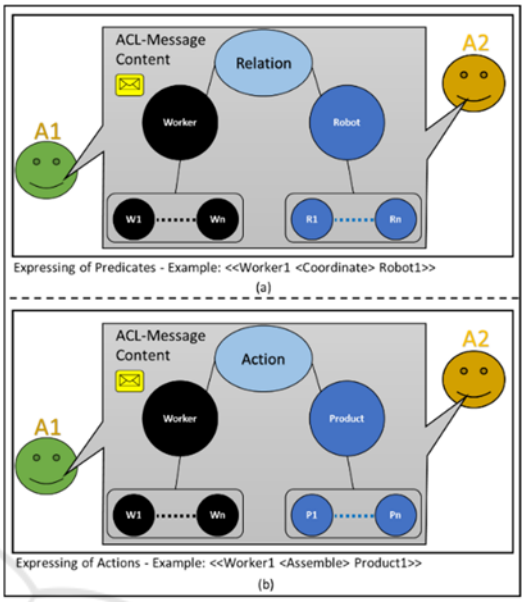

Figure 5: Agents Communication via Ontology.

The term ontology can be considered sometimes vague and not precise, therefore we state below the most suitable definitions of ontologies for our research (Rodrigues, 2012).

- An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well the rules for combining terms and relations to define extensions for this vocabulary (Neches et al., 1991).
- An ontology is a formal, explicit specification of a shared conceptualization (Gruber, 1995).
- An ontology is a logical theory accounting for the intended meaning of a formal vocabulary (Wang et al., 2012). i.e. “it is a commitment to a particular conceptualization of the world”.
- An ontology provides the meta-information to describe the data semantics, represent knowledge, and communicate with various types of entities (e.g. software agents and humans) (Fensel, 2004).
- An ontology can be described as “means of enabling communication and knowledge sharing by capturing a shared understanding of terms that can be used both by humans and machine software” (Lai, 2007).

All the previous definitions lead to the complete understanding of the meaning of ontology in the context of our research. Thus an ontology is a

conceptual tool to represent and create a common understanding for the manufacturing workcell entities (i.e., holons). Furthermore this common understanding would enable to exchange, reuse and extend the manufacturing knowledge. JADE supports the ontology-based MAS by defining three different types of schemas (Leitao et al., 1991):

- Terms: expressions that indicate entities (abstract or concrete) that exist in the MAS and that agents may reason about. Terms can be seen as primitives which are atomic data types such as strings or integers, and concepts which are complex structure such as objects.
- Predicates: expressions that describe the status of the world such as the relationships between the concepts.
- Actions: expressions that describe mechanisms or operations that can be executed by an agent.

5 CASE-STUDY

5.1 Case-study Description

During this research we selected a specific case-study where two workers are in cooperation with one cobot. The goal of the case-study is to implement the solution concept. The CPROSA holarchy deployment can be seen in Figure 6-a, JADE framework contains four containers which present the previously described CPROSA holons. Two CHs can be found in this case-study as it can be seen in Figure 6-b. Both the CHs have a similar UI. The UI of the CH is providing a tool for ordering a specific product with certain features (i.e., parts). The customer selects the very basic features and identifies the needed amount of the product then sends the order to the PH.

Two products can be manufactured in this case-study, the first is a centrifugal pump and the second is a screw compressor. The UIs of the pump and the compressor holons can be seen in Figure 6-c. The two products share some features such as the casing and the electrical motor. The pump has two unique features which are the impeller and the shaft, while the compressor has other two unique features which are the male-rotor and the female-rotor. When a PH receives a product order from the CH, it constructs the building plans for this product order as it will be discussed later in details. The PH also has the ability to rearrange the orders or modify them before sending them to the OH.

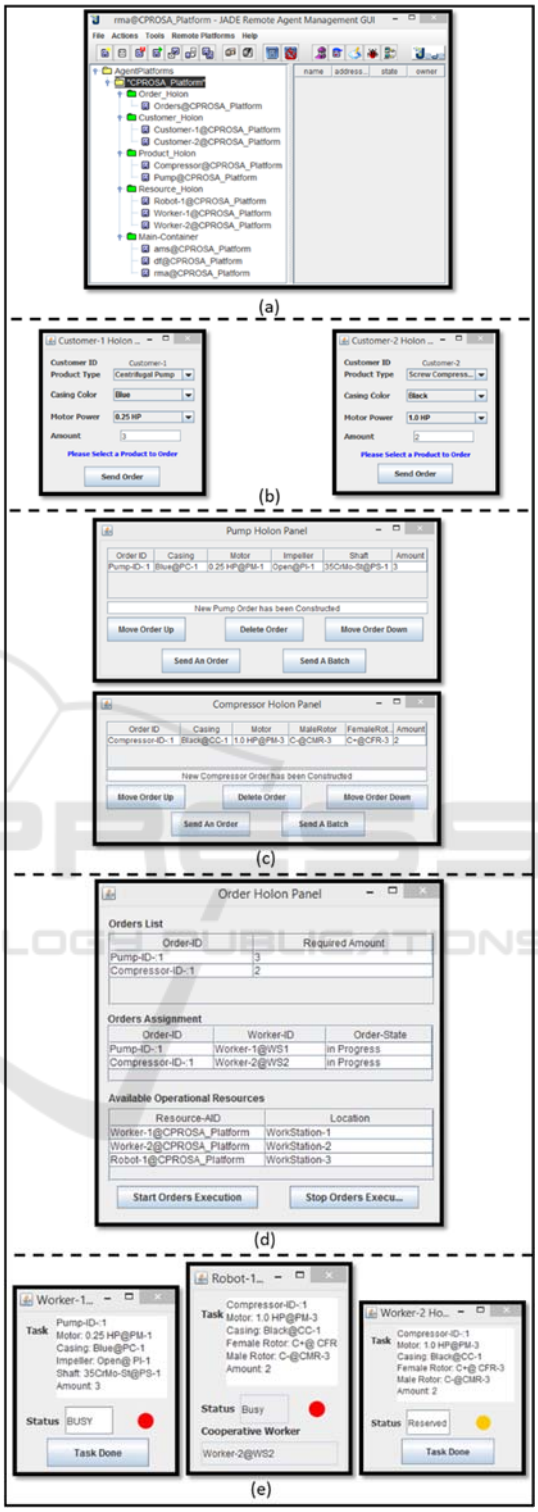

Figure 6: (a) CPROSA Model Implementation – (b) Customers Holons UI – (c) Products Holons UI – (d) Order Holon UI – (e) Operation Resources Holons UI.

The OH is responsible for collecting the product orders from all the other PHs as it is shown in Figure 6-d. Simultaneously the OH discovers the existence of the operation resources. Furthermore, it starts and stops the production process. Two WHs (W1H, W2H) and one Robot Holon (RH) are the operation resources in this implementation as it is shown in Figure 6-e. The function of the workers within this case-study is to perform an assembly operation for the customized product orders, while the function of the cobot is to pick and place the customized features of every production order to the worker workstation.

As we do not have a robot hardware during this implementation, we assumed that the cobot will always take two seconds to pick and place one product. Therefore the RH will multiply the number of products by two to obtain the overall time needed for the whole pick and place operations. Accordingly the RH can have two statuses, either busy or free. Another status is required for the WH which is the reserve status. In the reserve status the WH is waiting the cobot to load at least one product to the worker, therefore the worker can start the assembly operation and subsequently the WH status turns to be busy. The WH stays in the busy status till the worker presses the task-done button, then the WH status would be free.

5.2 Interaction Ontologies

5.2.1 Building-Operations-Ontology

As has been discussed earlier at the solution concept. JADE is using three different types of schemas to construct its ontology. Figure 7-a shows the required schemas to build a product order from a customer-order. The first set of JADE schemas which can be seen in the figure are the terms (i.e., concepts and primitives):

- Compressor-Customer-Order: a schema which encapsulates some attributes such as the required compressor color, the needed hydraulic power, and the required amount. Also it contains an AID as every customer- order is a life agent which needs a unique ID.
- Pump-Customer-Order: a schema which encapsulates some attributes such as the required pump color, the needed hydraulic power, and the required amount. Also it contains an AID as every customer-order is a life agent which needs a unique ID.
- Casing: a shared feature between the pump and the compressor. The casing schema contains two attributes which are the casing color and position at the features workspace or storage.

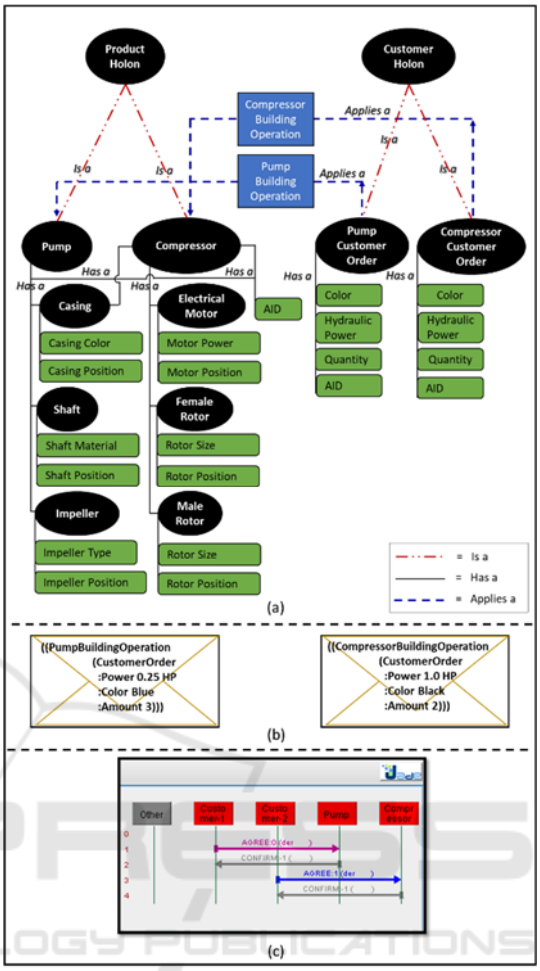

Figure 7: (a) Building-Operations-Ontology – (b) ACL-message Content for the Building-Operations-Ontology – (c) JADE Interaction between Customers Holons and Products Holons.

- Electrical-Motor: a shared feature between the pump and the compressor. The motor schema contains two attributes which are the motor electrical power, and position at the features workspace or storage.
- Shaft: a unique feature of the pump. The shaft schema contains two attributes which are the shaft material, and position at the features workspace or storage.
- Impeller: a unique feature of the pump. The impeller schema contains two attributes which are the impeller type, and position at the features workspace or storage.
- Female-Rotor: a unique feature of the compressor. The female-rotor schema contains two attributes which are the rotor size, and position at the features workspace or storage.

- Male-Rotor: a unique feature of the compressor. The male-rotor schema contains two attributes which are the rotor size, and position at the features workspace or storage.
- Compressor: a concept schema which encapsulates many other schemas under it, those schemas are the casing, electrical-motor, female-rotor, and male-rotor. Every compressor is a life agent, therefore it must contain an AID attribute as well.
- Pump: a concept schema which encapsulates many other schemas under it, those schemas are the casing, electrical-motor, shaft, and impeller. Every pump is a life agent, therefore it must contain an AID attribute as well.

The second set of JADE predicate schemas that have been used in this implementation can be addressed as the following:

- (concept-x) <Is-a> (concept-y): usually a relation between two concept schemas. This relation is similar to the object oriented abstraction. Thus, this predicate expression has been used to express the parent-child relationship between the concepts.
- (concept-x) <Has-a> (attribute-x): usually a relation between a concept and an attribute, an attribute can be a concept schema or a primitive. This relation is similar to object oriented inheritance. Thus, this predicate expression has been used to form sophisticated objects from simpler ones.
- (agent-x) <Applies-a> (action-x): usually a relation between a concept and an action schema. A concept uses this predicate expression to trigger one or more than one actions at the same time. The action schemas will be discussed below in details.

The third set of JADE action schemas that have been used in this implementation can be addressed as the following:

- Compressor-Building-Operation: this action schema expects a Compressor-Customer-Order concept as an input, and it can be deployed by either customer-1 or customer-2 agents. An example of this operation can be seen at the ACL-message content in at the right side of Figure 7-b.
- Pump-Building-Operation: this action schema expects a Pump-Customer-Order concept as an input, and it can be deployed by either customer-1 or customer-2 agents. An example of this operation can be seen at the ACL-message at the left side of Figure 7-b.

Figure 7-c shows JADE interaction scenario between the CHs (i.e., customer-1 agent and customer-2 agent) and the PHs (i.e., pump agent and compressor agent). In this scenario, customer-1 agent sends an ACL-message with an AGREE communicative act. The AGREE-message contains a Pump-Building-Operation and a Pump-Customer-Order. The AGREE-message is received by the pump agent. Therefore, the pump agent confirms the receiving by sending back a CONFIRM-message to customer-1 agent. Simultaneously the pump agent constructs a pump instance based on the incoming customer-order. The same building mechanism is used between cusomter-2 agent and the compressor agent to construct a new instance of a compressor associated with a customer-2 order.

5.2.2 Planning-Operations-Ontology

Figure 8-a shows the required schemas to construct the production planning from a product orders. Similar to the building-operations-ontology, the planning-operations-ontology contains three different kinds of schemas. The concept schemas are as following:

- Compressor-Order: a schema which extends the compressor schema by adding the required amount of compressor units.
- Pump-Order: a schema which extends the pump schema by adding the required amount of pump units.
- Operations-List: a schema which includes a list of operations which can be used to manufacture either a pump or a compressor. The schema can be used to manufacture a product which needs three operations or less.
- Compressor-Manufacturing-Order: a schema which combines a Compressor-Order schema and an Operations-List schema. Also it has an AID attribute as it acts as an agent.
- Pump-Manufacturing-Order: a schema which combines a Pump-Order schema and an Operations-List schema. Also it has an AID attribute as it acts as an agent.

The predicate schemas which has been used in this part of the implementation are exactly the same to the ones that has been used in the building-operations-ontology. While the action schemas are as following:

- Compressor-Manufacturing-Operation: this action schema expects a Compressor-Order and a Compressor-Operations-List concept schema as an input, and it is deployed by the compressor agent. A detailed example of this operation can be

seen at the ACL-message at the right side of Figure 8-b.

- Pump-Manufacturing-Operation: this action schema expects a Pump-Order and a Pump-Operations-List concept schema as an input, and it is deployed by the pump agent. A detailed example of this operation can be seen at the ACL-message content in Figure 8-b at the left.

Figure 8-c shows JADE interaction scenario between the PHs (i.e. pump agent and compressor agent) and the OH. This interaction is following the same mechanism used in the building-operations-ontology. Except that we changed the AGREE-messages with a PROPAGATE-messages.

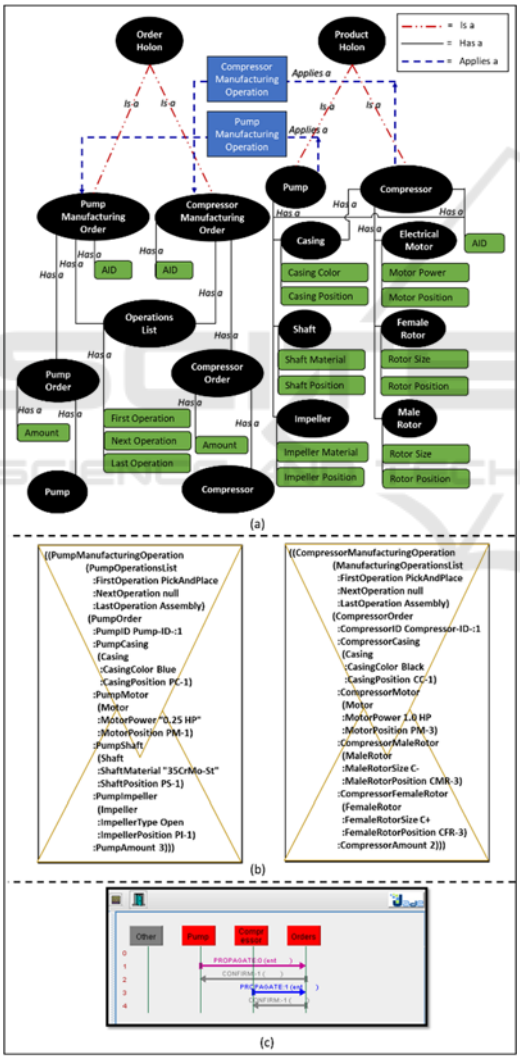

Figure 8: (a) Planning-Operations-Ontology – (b) ACL-message Content for the Planning-Operations-Ontology – (c) JADE Interaction between Products Holons and Order Holon.

5.2.3 Manufacturing-Operations-Ontology

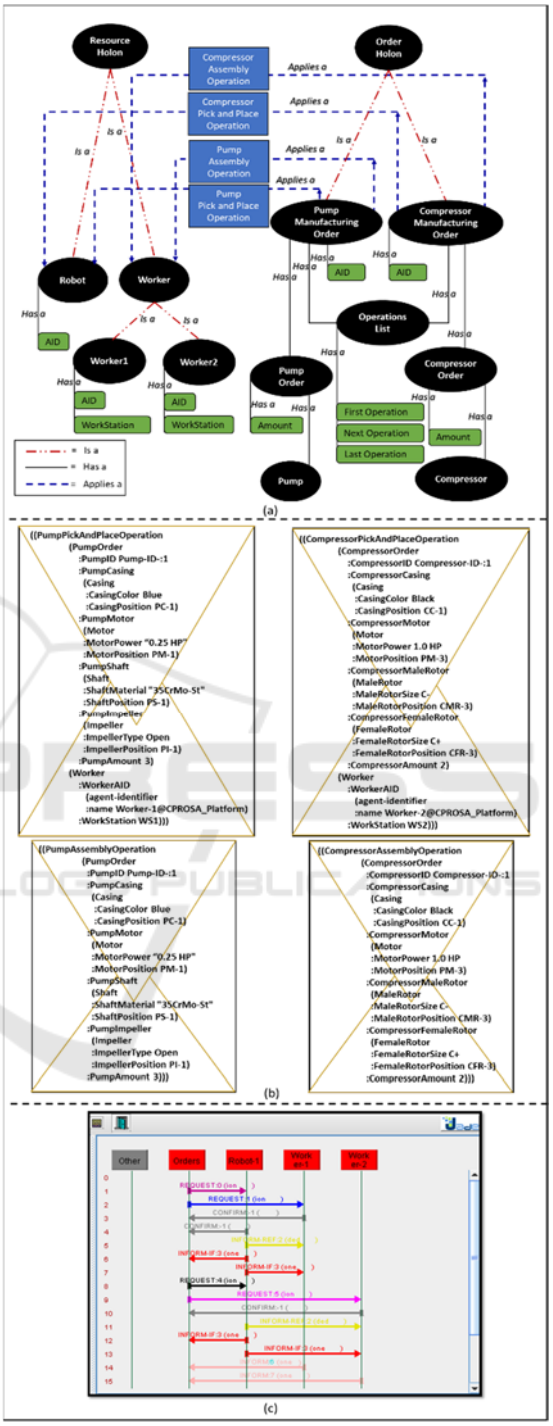

Figure 9: (a) Manufacturing-Operations-Ontology – (b) ACL-message Content for the Manufacturing-Operations-Ontology – (c) JADE Interaction between Order Holon and Resources Holons.

Figure 9-a shows the required schemas to execute the manufacturing operations from the production plans. Similar to the building-operations-ontology and the planning-operations-ontology, the manufacturing-operations-ontology contains three different kinds of schemas. The concept schemas are as following:

- Worker: a schema which contains two attributes, the first one is the worker AID as it acts as a life agent, and the second is the worker location within the workcell (i.e., workstation). The worker agent is providing an UI for the worker for providing the assigned task and inquiring the task done event (see Figure 6-e). Two instances of the worker agent exist in this case-study scenario. As has been mentioned before in section 5.1, the worker can have three statuses. A free status when there is no product orders or the production is not started. A reserve status when the worker is waiting the first product unit to be placed by the cobot. A busy status while the cobot is still handling the orders and till the worker triggers the task-done button.
- Robot: a schema which contains one attribute, which is the robot AID as it acts as a life agent. The robot schema does not have a workstation attribute because in this specific case-study, we have one cobot which is responsible for the pick and place. Therefore the location of the cobot is not necessary required, however in case of more than one cobot this attribute could be important. The robot agent is providing an UI to show the assigned task and the status of the cobot (see Figure 6-e). As has been mentioned before in section 5.1, the cobot can have two statuses. A free status when there is no product orders or the production is not started. A busy status when the cobot is picking and placing the production orders. A timer of two second has been assigned to every pick and place operation.

The predicate schemas which has been used in this part of the implementation is exactly the same to the ones that has been used in the building-operations-ontology and planning-operations-ontology. While the action schemas are as following:

- Compressor-Pick-And-Place-Operation: this action schema expects two concept schema inputs; the first concept schema input is the Compressor-Order which contains the detailed specifications of the compressor. Therefore, the cobot can use these information especially the compressor features positions to perform the pick operation. The second concept schema input is the target worker. Therefore, the cobot can use the worker workstation location to place the compressor features at this location. This action schema is deployed by the orders agent to interact with the robot agent. A detailed example of this operation can be seen at the ACL-message content at the top right of Figure 9-b.
- Pump-Pick-And-Place-Operation: this action schema expects two concept schema inputs; the first concept schema input is the Pump-Order which contains the detailed specifications of the pump. Therefore, the cobot can use this information especially the pump features positions to perform the pick operation. The second concept schema input is the target worker. Therefore, the cobot can use the worker workstation location to place the pump features at this location. This action schema is deployed by the orders agent to assign a task to the robot agent. A detailed example of this operation can be seen at the ACL-message content at the top left of Figure 9-b.
- Compressor-Assembly-Operation: this action schema expects one concept schema input which is the Compressor-Order. This operation is beneficial for the worker to provide him with the required features to build a customized compressor. Moreover, it provides the amount of required units. This action schema is deployed by the orders agent to assign a task to any of the worker agents based on their status. A detailed example of this operation can be seen at the ACL-message content at the bottom right of Figure 9-b.
- Pump-Assembly-Operation: this action schema expects one concept schema input which is the Pump-Order. This operation is beneficial for the worker to provide him with the required features to build a customized pump. Moreover, it provides the amount of required units. This action schema is deployed by the orders agent to assign a task to any of the worker agents based on their status. A detailed example of this operation can be seen at the ACL-message content at the bottom left of Figure 9-b.

Figure 9-c shows JADE interaction scenario among the OH and the ORHs (i.e., worker1 agent, worker2 agent, and robot agent). During this interaction, the manufacturing operations are assigned to the operational resources based on their status. As it can be seen in lines1, 2, 3, and 4 of Figure 9-c, the orders agent sends two REQUEST-messages which are replied by two CONFIRM-messages. The first REQUEST-message assigns a Pump-Pick-And-Place-Operation to the robot agent. The second

REQUEST-message assigns a Pump-Assembly-Operation to worker1 agent. The reason that the pump-order has been processed first by the orders agent is that it is the first product order at the order list (refer to Figure 6-d). In line 5 of Figure 9-c, the robot agent sends an INFORM-REF-message to worker1 agent to tell that it placed the first pump unit. Then, the robot agent sends two INFORM-IF-messages to the orders agent and worker1 agent to tell that it finished handling all the required pump amounts (i.e., three pump units by referring to Figure 6). The two INFORM-IF-messages can be seen in lines 6, and 7 of Figure 9-c. The same interaction mechanism can be seen in lines 9,10,11,12, and 13 to assign the compressor-order manufacturing operations to the worker2 agent and the robot agent. Lines 14, and 15 of Figure 9-c shows the INFORM-messages to express done-signal which is generated from worker1 and worker2 agents to the orders agent.

6 SUMMARY, CONCLUSION AND FUTURE WORK

During this research, we introduced the collaboration between the worker and the cobot as a new flexible trend in the reconfigurable manufacturing. As this subject has many different perspectives, we focused our research attention on the complex information interaction between the worker and the cobot. The challenges in this interaction are getting more sophisticated when we put it under the reconfigurable manufacturing umbrella.

CPROSA holarchy has been proposed as a solution for this dilemma. It is a modified version of PROSA model which adds the customer as a new basic holon in the HCA. Furthermore, our CPROSA is not only describing the interaction among the basic holons in order to accomplish the worker-cobot collaboration, but also it uses the powerful concept of ontology to create a shared understanding of the entities within the collaborative manufacturing environment. Furthermore, this shared understanding is used to achieve the complex interaction scenarios.

JADE framework has been used to implement the CPROSA solution model and the case-study scenario. During the case-study we proposed a simple scenario where two customers customize two different orders. The first order is three units of a customised centrifugal pump, and the second order is two units of a customized screw compressor. Moreover, we assumed that we have two workers in cooperation with one cobot. The products orders should be processed by the OH and then assigned to the ORHs based on their status. The reason for selecting this scenario is to show how much complex the interaction can be - even in a simple case-study. During the implementation of the case-study we showed three connected ontologies. The first ontology was responsible for building the product plans from the customer-orders, the second ontology was responsible for planning the manufacturing process, and the third ontology was responsible for executing the manufacturing operations.

The privileges of using the ontology concept were so influential in this implementation. For instance, the conceptualization and the modularity were very clear when at the concepts, predicates, and actions schemas. The object oriented abstraction can be seen in the parent-child relationships between the concept schemas. The inheritance can be implemented from one concept to another using Has-a predicate. All the predicate schemas have been reused at the different parts of the implementation which guarantees the reusability of the research concept. The extensibility was clear when extending one concept schema to another more complicated schemas, or when using many different ontologies to work together in the same interaction context.

Regardless the simplicity of the case-study, the solution concept shows that it is totally able to solve the addressed research challenges. However during the future work, we intend to implement the CPROSA model over more collaborative operational resources. Even it would be a very hard to demonstrate this kind of research in a single article. Other technologies such as XML and industrial web-services could be a good approach to solve this research challenges, which could be a focus in our future research. Also during the future research we will use a real hardware for the robot instead of the UI only.

REFERENCES

Alsafi, Y. and Vyatkin, V., 2010. Ontology-based reconfiguration agent for intelligent mechatronic systems in flexible manufacturing. *Robotics and Computer-Integrated Manufacturing*, Vol. 26, Issue 4, pp.381-391.

Babiceanu, R. and Chen, F. (2006). Development and Applications of Holonic Manufacturing Systems: *A Survey. Journal of Intelligent Manufacturing*, Vol. 17, Issue 1, pp.111-131.

Balakirsky, S., 2015. Ontology based action planning and verification for agile manufacturing. Robotics and Computer-Integrated Manufacturing, Vol. 33, pp.21-28.

Bellifemine, F., Caire, G., Greenwood, D., *Developing Multi-Agent Systems with JADE*, Vol. 7 ed. John Wiley & Sons, 2007.

Botti, V., Giret, A., Holonic Manufacturing Systems, *ANEMONA - A Multi-agent Methodology for Holonic Manufacturing Systems, Springer Series in Advanced Manufacturing*, pp. 7-20, 2008.

Caire, I., *JADE Tutorial: JADE Programming for Beginners*, TILAB & West Sussex, 2009.

Elmaraghy, Hoda A., Flexible and reconfigurable manufacturing systems paradigms, *International Journal of Flexible Manufacturing Systems*, Vol. 17, Issue 4, pp 261-276, October 2005.

Fensel, D., 2004. Ontologies. 1st ed. Berlin, Heidelberg: Springer Berlin Heidelberg.

Gruber, T., 1995. Toward principles for the design of ontologies used for knowledge sharing, *International Journal of Human-Computer Studies*, Vol. 43, Issue 5, pp.907-928.

Fiorini, S., Carbonera, J., Gonçalves, P., Jorge, V., Rey, V., Haidegger, T., Abel, M., Redfield, S., Balakirsky, S., Ragavan, V., Li, H., Schlenoff, C. and Prestes, E., 2015. Extensions to the core ontology for robotics and automation. *Robotics and Computer-Integrated Manufacturing*, Vol. 33, pp.3-11.

FIPA, Fipa.org, 2016. [Online]. Available: http://www.fipa.org/. [Accessed: 06- Oct- 2016].

JADE | Java Agent DEvelopment Framework, *Jade.tilab.com*. [Online]. Available: http://jade.tilab.com/. [Accessed: 1- Oct- 2016].

Jennings, N.R., Wooldridge, M.J., Applications of intelligent agents, *Agent Technology: Foundations, Applications, and Markets*, Springer, pp. 3–28, 1998.

Koestler, A., *The Ghost in the Machine*, Hutchinson, 1967.

Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritschow, G., Ulsoy, G., Brussel H., 1999, Reconfigurable Manufacturing Systems, CIRP Annals, Vol. 48, pp. 527-540.

Kruger, K., Basson, A., 2015, Implementation of an Erlang-Based Resource Holon for a Holonic Manufacturing Cell, *Studies in Computational Intelligence*, Vol. 594, pp. 49-58.

Lai, L., 2007. A knowledge engineering approach to knowledge management. *Information Sciences*, Vol. 177, Issue 19, pp.4072-4094.

Leitao, P., Rodrigues, N., Turrin, C., Pagani, A., Petrali, P., 2012. GRACE ontology integrating process and quality control, *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society.*

Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T., Swartout, W., 1991. Enabling Technology for Knowledge Sharing, *Al Mag.*, Vol. 12, pp.36-56.

Poslad, S., Specifying Protocols for Multi-Agent Systems Interaction, 2007. *ACM Transactions on Autonomous and Adaptive Systems*, Vol. 2.

Rodrigues, N., 2012. Development of an Ontology for a Multi-Agent System Controlling a Production Line, MSc thesis, Instituto Politécnico de Bragança.

Sadik A.R., Urban B., 2016, A Novel Implementation Approach for Resource Holons in Reconfigurable Product Manufacturing Cell, *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics*, pp. 130-139.

Sadik A.R., Urban B., 2017, Applying the PROSA Reference Architecture to Enable the Interaction between the Worker and the Industrial Robot - Case Study: One Worker Interaction with a Dual-Arm Industrial Robot, *Proceedings of the 9th International Conference on Agents and Artificial Intelligence - ICAART*, Vol 1, pp. 190-199.

Shen, W., Hao Q., Yoon, H. J., Norrie, D. H., 2006. Applications of agent - based systems in intelligent manufacturing: An updated review, *Advanced Engineering Informatics*, Vol. 20, pp. 415-431.

Teahan, W., *Artificial Intelligence – Agent Behaviour*, BookBoon, 2010.

Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Petters, P. Reference Architecture for Holonic Manufacturing Systems: PROSA. *Computers in Industry*, Vol. 37, pp. 95–108, 2003.

Wang, H., Gibbins, N., Payne, T. and Redavid, D., 2012. A formal model of the Semantic Web Service Ontology (WSMO). Information Systems, Vol. 37, Issue 1, pp.33-60.