Agent-based Reconfiguration in a Micro-flow Production Cell

José Dias*, Johan Vallhagen† José Barbosa*‡, Paulo Leitão*§,
* Polytechnic Institute of Bragança, Campus Sta Apolónia, 5300-253 Bragança, Portugal,
Email: {jose.dias, jbarbosa, pleitao}@ipb.pt
† GKN Aerospace Sweden AB, 46181 Trollhaettan, Sweden
Email: johan.vallhagen@gknaerospace.com
‡ INESC-TEC, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
§ LIACC – Artificial Intelligence and Computer Science Laboratory, Rua Campo Alegre 1021, 4169-007 Porto, Portugal

***Abstract*—The world is moving towards to the fourth industrial revolution, usually linked with the Industrie 4.0 initiative, enables the digitization of manufacturing factories by using Cyber-Physical Systems and emergent technologies like Internet of Things and Internet of Services. The seamless reconfiguration of these complex industrial cyber-physical systems is an important challenge for the complete implementation of this revolution, being necessary to re-think the way such mechanisms can be designed and engineered. This paper presents an agent-based reconfiguration system for the dynamic and seamless re-configuration of a physically-reconfigurable modular micro-flow production system in the area of manufacturing of aerospace engine components.**

I. Introduction

The fast mutation in the costumer needs forces the manufacturing companies to introduce and promote a continuous improvement in their production processes, adapting them in a timely manner. An increase of the products customization and complexity levels are among the key factors that contributes to this rapid mutation.

As the world faces its 4th industrial revolution, many research and development work is being conducted worldwide, pushing the current state-of-the-art forward with the vision of empowering companies with the necessary set of tools to better face these highly complex demands. To foster this, several national, e.g., the Industrie 4.0 initiative [1], and global financing programs, e.g., the European Horizon 2020 program, are creating the appropriate stimulus for the digitization of industry, aiming to achieve more modular, reconfigurable, sustainable and adaptive systems.

This new vision is aligned with the key features found in the well-known Reconfigurable Manufacturing Systems (RMS) paradigm [2], where modularity, integrability, customization, convertibility and diagonalisability are cornerstones. The RMS is a concept that suggests the rapid change in the factory's structure using changes in hardware and/or software to adjust the production capacity and functionality [3]. Therefore, it becomes clear that the agility introduced in the adaptability and/or reconfigurability of the software and hardware layer is crucial to fulfil the nowadays manufacturing requirements, enabling a truly plug-and-produce manufacturing system (see as example the EU FP7 PRIME project [4]).

The aeronautical industry is subjected to similar challenges, and is forced to follow the same innovative approaches to face the need to reach product customization and productivity in highly complex low volume production. In suitable domains, the innovative micro cell concept can be adopted in the aeronautical industry, allowing an easy system reconfiguration according to the production needs. The micro-flow concept is a part of the PERFoRM (Production harmonizEd Reconfiguration of Flexible Robots and Machinery) project that intends to address this problem by establishing a seamless reconfiguration architecture to achieve a flexible manufacturing environment based on the rapid and seamless reconfiguration of machinery and robots as response to operational or business events.

This paper focuses the development of a scalable agent-based reconfiguration tool that supports the dynamic, seamless and on-the-fly reconfiguration and plugability in a micro-flow production cell comprising several different modular processes for producing aerospace engine components. The use of Multi-Agent Systems (MAS) provides several key features that supports the seamless reconfiguration, modularity, plugability, scalability, cooperation, interoperability and data persistence. The proposed solution was deployed in an industrial prototype, which allowed testing the designed solution.

The rest of the paper is organized as follows: Section II overviews the PERFoRM concept regarding the seamless reconfiguration of robotics and machinery, and Section III introduces the micro-flow production cell use case. Section IV describes the agent-based reconfiguration architecture and Section V presents its deployment in the industrial use case. Finally, Section VI rounds up the paper with the conclusions and points out the future work.

II. PERFoRM Concept to the Seamless Reconfiguration

The main goal of the PERFoRM system architecture [5], illustrated in Figure 1, to allow the transparent and interoperable communication between different hardware and software components, such as PLCs (Programmable Logic Controllers), industrial robots and SCADA (Supervisory Control and Data Acquisition) systems, aiming to reach a seamless production

978-1-5386-0837-1/17/$31.00 ©2017 IEEE

system reconfiguration. These heterogeneous hardware and software production components, addressing the different ISA-95 levels, expose their functionalities as services.

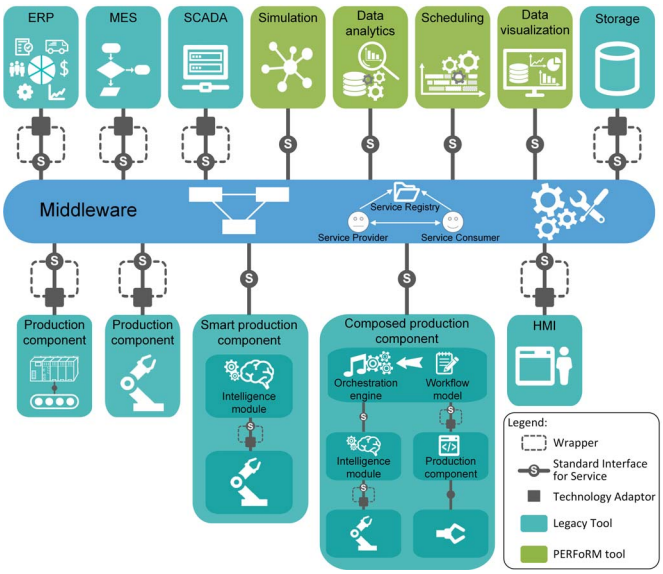

Figure 1. PERFoRM system architecture

The core component of the architecture is the distributed service-based integration layer (known as industrial middleware) that acts as an integration platform for the individual components. The definition of standard interfaces is a crucial architectural design issue to tackle the interoperability in real industrial environments. These interfaces fully expose and describe services in a unique, standardized and transparent way to enhance the seamless interoperability and pluggability. For this purpose, a common data model was adopted, based on AutomationML, serving as the data exchange format shared between the PERFoRM-compliant architectural elements, covering the semantic needs associated to each entity. Technology adapters aim to connect legacy systems to the PERFoRM middleware and to convert the legacy data (e.g., from an existing non-PERFoRM compliant database or robot) into the PERFoRM's data model.

The PERFoRM industrial middleware together with the proper adapters and standard interfaces permit the interconnection of heterogeneous legacy hardware devices, e.g. robots and the respective controllers, and software applications, e.g. databases, SCADA applications and other management, analytics and logistics tools.

III. Micro-flow Production Cell

The industrial case studied in this paper is in the area of manufacturing of aerospace engine components. This is typically metallic components that are machined to final dimensions and need to fulfil very high quality requirements. The process operations often include several steps of surface treatment processes, as well as cleaning, inspection and non-destructive testing (NDT).

As a complement to traditional production cells and resources, a concept for a flexible and reconfigurable production cell has been developed. One goal of the PERFoRM project is to develop and demonstrate more agile and automated production using an integrated system that can complete a short sequence of common operations in the value adding process chain. The initially identified processes to automate are surface treatment operations with robots, e.g., de-burring or polishing, and integrate with the subsequent processes for cleaning and inspection. However, the results may be extended and deployed to other assets and processes in the future. The concept, illustrated in Figure 2, is called micro-flow cell.

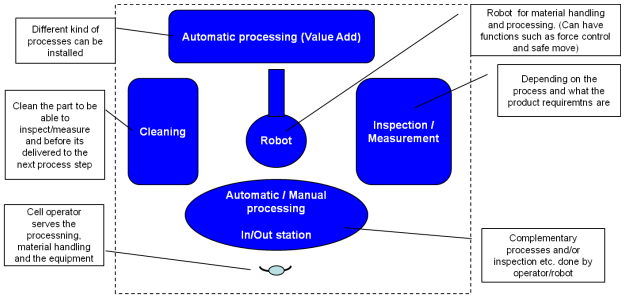

Figure 2. A principle description of the "Micro-Flow Cell concept"

The base configuration and components of the micro-flow cell are a computer, responsible for the cell functions, a PLC for the communication and control, a Human Machine Interface (HMI) for the operator interface, at least a robot system for the part handling and processing, and a safety system for the cell. Each process module has its own PLC running the local control system.

The reconfiguration is triggered according to the schedule: once received the schedule, the process modules are plugged/unplugged to accomplish the work orders defined in the schedule. The programs for the robot manipulation according to the different process modules are stored in a database, being downloaded every time the robot performs an operation.

This kind of system introduces several potential advantages. From a production efficiency perspective, the main goal is to develop and deploy less complex automated solutions for low volume production. In terms of productivity, shorter least time is also expected. To meet the requirements of a cost-effective solution, the goal is to develop a modular production cell concept that can be flexible and reconfigured with different processes. The cell and its process modules should be easily and quickly changed depending on the current production demands aligned with the ideas of plug-and-produce for cyber-physical systems. Therefore it will require a well designed architecture and reconfiguration mechanism.

IV. Agent-based Reconfiguration Mechanism

The agent-based reconfiguration tool, placed within the PERFoRM ecosystem, focuses on the logical re-organization of micro-flow production cells. This section will detail its architecture as well as the description of the reconfiguration

mechanism through the dynamic and automatic plug-in and plug-out of the modular processes.

A. MAS to Support the Reconfiguration

This tool is built upon MAS principles, considering a society of distributed and autonomous agents to regulate the re-organization procedure of the micro-flow cell. The MAS paradigm [6], [7], derived from the distributed artificial intelligence field, is pointed out as a suitable approach to support flexibility, robustness and reconfigurability. The inherent characteristics of MAS can easily support the micro-flow cell requirements, namely in terms of reconfigurability, scalability and plugability. In particular, the use of MAS principles in this tool can bring the following benefits:

- **Scalability**: the addition of new processes in the micro-flow cell is very simple in logical terms, simply requiring the instantiation of developed agent class on the fly (with its proper customization); note that there is no need for additional code from the point of view of the agents, or also the need to stop, re-program and re-start the tool.
- **Distributed data collection**: the MAS approach allows to implement a distributed approach for the collection of data from the process modules and robots supporting the implementation of performance monitoring.
- **Data persistence**: this solution guarantees the persistence of the data in case of failure. The agents store individually and locally the current cell configuration and its own status. In case of breakdown, when the system is turned on, each individual agent accesses to the local database to update the cell configuration and its own status.
- **Cooperation**: the interaction among multiple robots and process modules to exchange information, regarding to schedule and execute their operations after a reconfiguration procedure, is simplified by using MAS. Additionally, in case of multiple micro-flow cells, the creation of a network of interconnected agents increases the seamless reconfiguration, namely by reacting in real time to failures, rescheduling dynamically the system and overcoming problems related to performance bottlenecks.

B. Agent-based Architecture

The designed MAS-based system is composed of two types of agents, namely the "Robot Agent" representing robot resources and the "Process Agent" representing process modules. The creation of two types of agents increases the modularity and flexibility of the system, since the agents have distinct functions, establishing in this way a similarity between the logical and the physical levels, allowing the agents to have a better representation of the system. The robot agent is linked with the cell configuration and safety procedures, while the process agent only cares with its associated physical process. Aiming to keep the cell configuration updated during the reconfiguration procedure, the process and robot agents need to interact, namely to exchange information regarding the plug-in and plug-out of the process modules.

The internal architecture of these agents is illustrated in Figure 3. Briefly, each agent is managing the logical re-organization of its physical device, contributing for the re-organization of the micro-flow-cell. The agents interconnect their physical counterparts, i.e. robot controllers and PLCs, by getting data from OPC-UA (Open Platform Communications - Unified Architecture) servers, acting as adapters to be PERFoRM compliant. The information collected from the physical resources, regarding current configuration status and performance, is stored in their local databases. The agents reason using the historical and current data to perform the reconfiguration procedure, and share information by exchanging messages among themselves according to the FIPA-ACL (Foundation for Intelligent Physical Agents – Agent Communication Language) communication language.

Figure 3. Internal architecture of the agents

The behaviour of each type of agent is dependent of its role and objectives, being in this work formalized using the Petri nets formalism [8]. Petri nets is a formal modelling tool adequate to model and to analyse the behaviour of complex event-driven systems characterised as being concurrent, asynchronous, stochastic and with high level of distribution.

The Petri net behaviour model for the process agent is illustrated in Figure 4. The process agents are created according to the catalogue of modular process modules that can be used in the micro-flow production cell, being created one process agent for each physical process module.

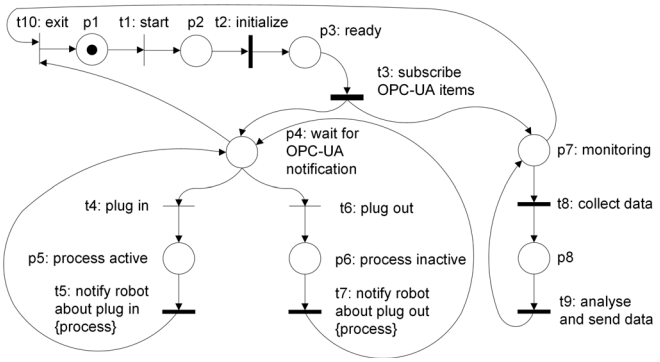

Figure 4. Behaviour model for the process agent

After the initialization phase, where the agent is parametrized according to the details of the process module it represents, namely its name and OPC-UA server address, and subscribes some process's parameters in the OPC-UA server, the agent enter in a sleep mode, waiting for an event (as result of the initial subscription) notifying that its associated process is plugged-in. At this stage, the agent switches to an active mode, updates its position and location within the micro-flow production cell and notifies the robot agent responsible for this cell about its new state, proving information regarding the process and its position. A same procedure is executed when the process agent receives an event notifying that the process is plugged-out. This mechanism allows the seamless reconfiguration of the cell in a distributed manner, since the process agents are individually detecting when are plugging-in and -out, and exchanging information with each other and with the robot agents to maintain the proper knowledge about the cell configuration.

In active mode, each process agent is also continuously collecting data related to the performance of its process module, e.g., processing time, number of processed parts and status (idle or execution). This data is analysed and aggregated and sent to an external Key Performance Indicator (KPI) monitoring tool.

The Petri net behaviour model for the Robot agent is illustrated in Figure 5.

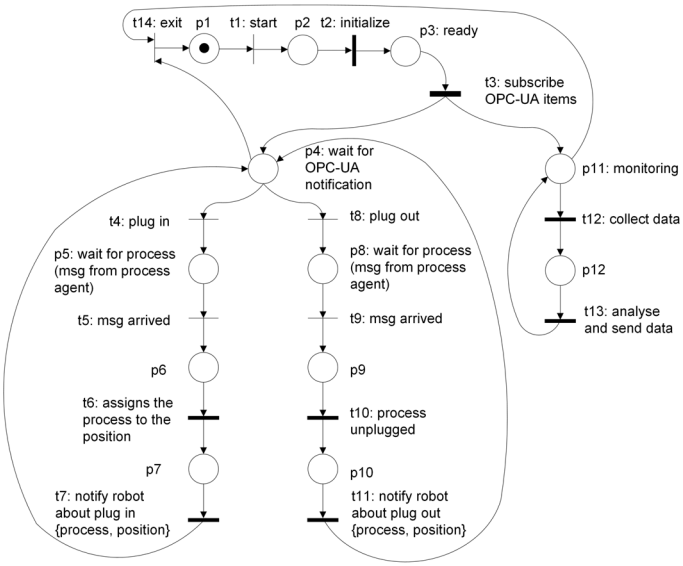

Figure 5. Behaviour model for the robot agent

The robot agent is launched at the same time as the process agents, being created one robot agent for each physical robot placed in the micro-flow production cell (usually, a micro-flow cell has only one robot and only one robot agent is created, but if it contains more robots, additional robot agents are created).

After the initialization phase, similar to the one described for the process agent, where the agent is parametrized according to the robot type, the agent remains in active mode, monitoring the performance of its physical counterpart and waiting for an event, as result of the initial subscription: a plug-in or plug-out of a modular process. When a process module is plugged-in, the robot agent is automatically notified about the change in the cell configuration, storing the info related to the plugged-in process and its location in its local database. This information is assigned to the robot controller that should adapt itself to work with this new modular process at this location, namely downloading the proper robot program and changing its tools. A same procedure occurs when the robot agent receives an event notifying that a process is plugged-out. In this case, the agent updates the unavailability of the process and informs the robot controller that this process is not anymore active.

After the reconfiguration of the micro-flow production cell, the cell can return to the processing state, if the cell safety procedures are guaranteed. In fact, aiming to ensure the safety, robot agents should check the information related to the cell safety systems, e.g., light curtains and emergency stops, before allowing to return to the normal processing operation.

C. Plug-in and Plug-out Mechanisms

As previously referred, process and robot agents need to interact to reach the seamless reconfiguration procedure. For this purpose, the reconfiguration can be triggered by plugging in and/or plugging out modular processes. Figure 6 describes the sequence diagram to reconfigure the micro-flow cell due to the plug-in of a process module.

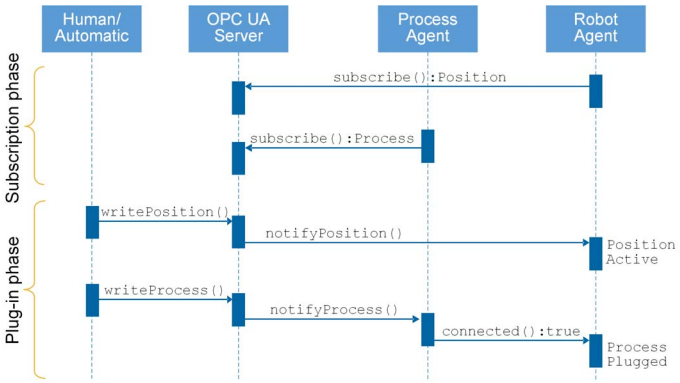

Figure 6. Sequence diagram for the plug-in of a process module

This procedure is divided in two phases, namely the subscription phase and the plug-in phase. On the subscription phase, the robot agent subscribes the cell positions related to its operation scope, while the process agents subscribe their available process modules.

In the plug-in phase, the system is waiting for a reconfiguration, that occurs when the operator plugs-in a process module into a specific position. When a position becomes active, by plugging-in a process module, the robot agent is automatically notified (since it had subscribed this event) that there is a process module incoming into that position. The process agent is also notified that the process module associated to it is plugged, switching its status from *sleep mode* to *active mode* and informing the robot agent (by sending a message), that

the process it is connected. In this way, the robot agent knows which process is plugged in each position.

This connectivity can be done in two different ways: automatically or manually. If the cell is equipped with sensors that allows to know which positions are active, and the process modules have RFiD (Radio-Frequency IDentification) tags that identifies themselves when attaching a position (e.g., using a RFiD reader), the process identification can be performed in automatic way. Otherwise, this identification can be performed by the operator that plugs-in physically the process module and inserts the position and the process identification through a HMI.

Independently of the physical connection approach, the process identification information is stored in an OPC-UA server, which variables are subscribed by the several robot and process agents. This allows to completely separate the agent-based system from the physical technological implementation of the micro-flow production cell.

Regarding the plug-out of a process, a similar interaction is performed by the process and robot agents, with the robot agent subscribing the position and the process agent the process modules. When a process module is plugged-out, the robot agent is notified that there is a process out-coming from that position and the process agent is notified that the process is being unplugged. After receiving the notification, the process agent informs the robot agent that the process is disconnected, and switches its status to *sleep mode*. In this way, the robot knows automatically which process module is not available anymore and which position becomes inactive.

With this distributed and loose coupled behaviour, the agent-based solution can discover in a dynamic and automatic manner the plug-in and plug-out of process modules, allowing to support the seamless reconfiguration of the micro-flow production cell composition.

V. Deployment of the Agent-based Reconfiguration Mechanism

The designed agent-based reconfiguration tool was deployed to support the dynamic reconfiguration of the production cell.

A. Agent-based Infrastructure

The agent-based infrastructure was implemented using the Java Agent Development Environment (JADE) framework [9]. The agents communicate to implement their cooperation patterns through the use of messages formatted according to the FIPA-ACL specification [9]. The agents are created according to the number of available process modules and robots. The several instantiated agents are parametrized by parsing an AML (Automation Markup Language) file (see Section V.C for more details), which contains, for each agent, the name, type of device (i.e. process or robot), characteristics of the process/robot, and connection info to the physical device. In this work, seven process agents (brushing, dimension, marking, roughness, grinding, polish and deburring) and one robot agent (managing an ABB industrial robot) were created to manage the reconfiguration of the micro-flow production cell. All the agents register their skills in the Directory Facilitator (DF), which works like yellow pages services for the agents. It maintains an accurate, complete and timely list of the agents, that can be easily and on-line discovered by the other agents to support the seamless reconfiguration.

B. Interconnecting the Physical Devices

The agents are interconnecting robots and machinery, by using the OPC-UA standard (IEC62541) [10], as illustrated in Figure 7, which is aligned with the industry trends in factory automation to exchange real-time data information. The connection between the OPC-UA server and the PLC is established through Modbus TCP/IP Ethernet and with the robot through Ethernet.

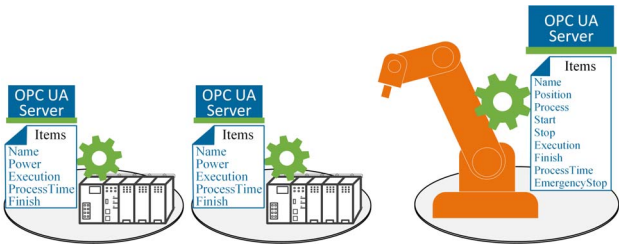

Figure 7. Interfacing agents with physical hardware devices using OPC-UA

The base modelling of the OPC-UA is the concept of *Nodes*, which represent instances. A *NodeId* uniquely identifies a Node in the OPC-UA server and is used to address the Node [10]. An example of a used *NodeId* is listed below, namely for the position "P1" of the micro-flow cell (other nodes types are illustrated in Figure 7).

```
NodeId nodeIdCellP1 = NodeId.parseNodeId("ns=2;s=
    Robot.ABB.IRB1400.bool.P1");
```

The agents are running in the cell computer and exchange data with the OPC-UA server by means of standard interfaces and using mainly three methods, namely, *read*, *write* and *subscribe*. The subscribe method is of particular importance to ensure the event notification of desired events. In this case, a subscription can monitor multiple Items, where a Monitored Item is used to define the Attribute of a Node that should be monitored for data changes [10]. The monitoring of the changes on the Node related to the position P1 is described in the next excerpt of code.

```
public void uaSubscribe() throws ServiceException,
      StatusException {
       subscription = new Subscription();
       MonitoredDataItem itemPlugP1 = new
            MonitoredDataItem(opcUaClient.
            nodeIdPlugP1, Attributes.Value,
            MonitoringMode.Reporting);
       subscription.addItem(itemPlugP1);
       itemPlugP1.setDataChangeListener(
            dataChangeListenerPlugP1);
       opcUaClient.client.addSubscription(
            subscription);
}
```

In this way, the Monitored Item is used to subscribe for data changes on the Attribute value of a Node. According to

a publish interval, when the value changes, the client who has subscribed the event will receive a notification.

C. Data Model Instantiation

The part of the data manipulated by the agent-based reconfiguration tool is described using the PERFoRMML (PERFoRM Markup Language) data model specified in [11], which is based on the AutomationML [12]. PERFORMML uses OPC-UA as data format /transport protocol. The *PMLEntity* class is the generic representation of shop floor entities, encapsulating the information of components and subsystems. On the *PMLComponent* or *PMLSubsystem* it is defined the associated entities, skills and values. The PLMValue elements enable the basic representation of information pertaining to data machinery level, namely in terms of parameters, e.g., ***ID*** - a string that serve as unique identifier for this element, ***Description*** - a string containing the description of the element, ***Address*** - a string containing the address of a value (OPC-UA).

The part of the data model containing the description of the topology of the micro flow cell (see Figure 8) results in an AML file. AML is a XML-based data format built upon other well established, open standards spanning several engineering areas, e.g., the Computer Aided Engineering Exchange (CAEX) that serves as the basis of hierarchical plant structures aiming at interconnecting them [13].

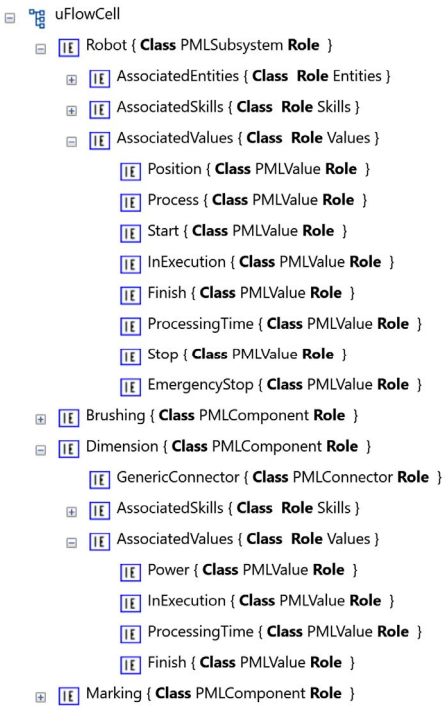

Figure 8. Topology of the micro-flow cell represented using PERFORMML

In this AML file, the Instance Hierarchy comprises the topology of the cell, being built by adding the "elements" from the System unit class, which contains all the needed elements to describe the micro-flow cell.

VI. Conclusion

The current industrial revolution is demanding for a paradigm shift, across all manufacturing domains, regarding the integration of hardware (physical) and software (cyber) counterparts, in the form of CPS. The aeronautical domain is no exception and is being pushed to follow this trend.

This paper presented an agent-based reconfiguration tool to address the seamless reconfiguration of micro-flow production cells, providing descriptions of the system architecture, reconfiguration mechanisms and technical deployment. The deployment in the use case prototype allowed the validation of the proposed approach, particularly verifying the dynamic and on-the-fly reconfiguration of the micro-flow production cell.

Future work will be devoted to the complete deployment of the agent-based solution in the industrial micro-flow production cell, as well as the the smooth migration of the existing cell to the new CPS and PERFoRM compliant ecosystem.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 680435.

References

[1] H. Kagermann and W. Wahlster and J. Helbig, "Securing the future of German manufacturing industry: Recommendations for implementing the strategic initiative INDUSTRIE 4.0," ACATECH – German National Academy of Science and Engineering, Tech. Rep., 2013.

[2] Y. Koren, F. Jovane, U. Heisel, T. Moriwaki, G. Pritschow, G. Ulsoy, and H. Van Brussel, "Reconfigurable Manufacturing Systems," *CIRP Annals*, vol. 48, No. 2, pp. 6–12, 1999.

[3] H. ElMaraghy, "Flexible and Reconfigurable Manufacturing Systems Paradigms," *Proceedings of the International Journal of Flexible Manufacturing System*, vol. 17, pp. 261–271, 2006.

[4] A. Rocha, G. Di Orio, J. Barata, N. Antzoulatos, E. Castro, D. Scrimieri, S. Ratchev, and L. Ribeiro, "An agent based framework to support plug and produce," in *12th IEEE International Conference on Industrial Informatics (INDIN)*, 2014, pp. 504–510.

[5] P. Leitão, J. Barbosa, A. Pereira, J. Barata, and A. Colombo, "Specification of the PERFoRM architecture for seamless production system reconfiguration," in *Proceedings of the 42nd Annual Conference of IEEE Industrial Electronics Society (IECON'16)*, 2016, pp. 5729–5734.

[6] J. Ferber, *Multi-Agent Systems, An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, 1999.

[7] M. Wooldridge, *An Introduction to Multi-Agent Systems*. John Wiley and Sons, 2002.

[8] T. Murata, "Petri Nets: Properties, Analysis and Applications," *IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[9] F. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*. Wiley, 2007.

[10] W. Mahnke, S. H. Leitner, and M. Damm, *OPC Unified Architecture*. Sringer Verlag, 2009.

[11] R. S. Peres, M. Parreira-Rocha, A. D. Rocha, J. Barbosa, P. Leitão, and J. Barata, "Selection of a data exchange format for industry 4.0 manufacturing systems," in *42nd Annual Conference of the IEEE Industrial Electronics Society (IECON'16)*, 2016, pp. 5723–5728.

[12] R. Drath, A. Lüder, J. Peschke, and L. Hundt, "AutomationML - The glue for seamless Automation engineering," in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2008, pp. 616–623.

[13] S. Faltinski, O. Niggemann, N. Moriz, and A. Mankowski, "AutomationML: From data exchange to system planning and simulation," in *IEEE International Conf. on Industrial Technology*, 2012, pp. 378–383.