A multi-agent architecture for plug and produce on an industrial assembly platform

Nikolas Antzoulatos, Elkin Castro, Daniele Scrimieri and Svetan Ratchev

Faculty of Engineering, University of Nottingham, University Park, Nottingham, NG7 2RD, United Kingdom

{eaxna6, elkin.castro, daniele.scrimieri, svetan.ratchev}@nottingham.ac.uk

Abstract

Modern manufacturing companies face increased pressures to adapt to shorter product life cycles and the need to reconfigure more frequently their production systems to offer new product variants. This paper proposes a new multi-agent architecture utilising "plug and produce" principles for configuration and reconfiguration of production systems with minimum human intervention. A new decision-making approach for system reconfiguration based on tasks re-allocation is presented using goal driven methods. The application of the proposed architecture is described with a number of architectural views and its deployment is illustrated using a validation scenario implemented on an industrial assembly platform. The proposed methodology provides an innovative application of a multi-agent control environment and architecture with the objective of significantly reducing the time for deployment and ramp-up of small footprint assembly systems.

Keywords: Plug and Produce, Multi-Agent System, Architecture, Assembly

1 Introduction

In the past decades, manufacturing companies had to cope with increasingly unpredictable market trends and growing customer demands for high quality customised and personalised products. Consequently, modern markets are characterised by shorter product lifecycles, increased product diversity and shorter

ust guarantee high levels of responsiveness to changes in product design using new manufacturing technologies.

The need for more flexible and adaptable manufacturing systems has long been recognised by industry and there has been a significant volume of research conducted in this area. Since the advent of the first reports on Flexible Manufacturing Systems (FMS), a number of paradigms have been introduced with

the aim of achieving higher levels of flexibility for cost effective manufacture. These include concepts such as Bionic Manufacturing Systems [1], Holonic Manufacturing Systems [2,3], Reconfigurable Manufacturing Systems [4,5], Evolvable Production Systems [6] and Evolvable Assembly Systems [7]. A common denominator of these trends has been the encapsulation of individual production functions into independent production units, like workstations or machines, that can be combined to build new, often modular, manufacturing systems or adapt existing ones.

There have been a number of research studies focused on developing common architectural approach to manufacturing system configuration. The concept of "plug and produce" is about interchanging self-contained modules of manufacturing systems and derives from the "plug and play" devices used in computing. In manufacturing, plug and produce aims at enhancing the interoperability and reusability of modules, thus reducing integration times [8]. This helps to satisfy the requirement of rapid system configuration and reconfiguration and to achieve system scalability for cost effective response to product and volume changes. The major difference between "plug and produce" and "plug and play" is the level of complexity, which is higher for manufacturing equipment [9,10]. In particular, production equipment does not always comply with structural or configuration standards and there are complex levels of interactions [11]. For instance, two robots sharing the same workspace need additional constraints to avoid interference and collision [12,13].

A three-layer architecture has been reported in [14] to enable the pluggability of modules. The lowest layer supervises the plug in and plug out of production components with specific electronic datasheets describing their functionalities and parameters. Their parameters are used in the configuration layer to setup the resources and integrate them into the system. The operator can then assign programs to the controller at the application layer. This approach enables plug and produce for robots and reduces configuration time by supporting self-configuration. However, it does not allow new components to be detected by the system or respond to configuration changes by reassigning tasks or rerouting products.

Agent technology has been proposed as a key enabler for modularisation of production resources and efficient communication between them. A modular control system architecture based on agents is presented in [15] to satisfy the requirements of a scalable automation system in terms of variable production rates. The proposed architecture is based on different elements of the production resources controlled by sensor, actor and product agents. These agents send information to a planning agent, which coordinates their interactions (e.g. in relation to the definition of robot paths that avoid collisions). Any planning activity is monitored by a central supervisor agent, which reports to the operator. Whilst partially applicable, the architecture requires low-level control access to individual components such as sensors, which is often infeasible due to the constraints of proprietarily control systems. In our application, it is required to control a production resource only with a PLC and to treat a production resource as an encapsulated module.

The IDEAS (Instantly Deployable Evolvable Assembly Systems) project [16] developed three elements to implement plug and produce assembly systems, namely a new control paradigm for distributed control principles, an agent-based control architecture, and intelligent mechatronic devices. The approach utilises bespoke mechatronic devices to host the agent technology that can change the control logic. This limits the use of standard industrial technology and legacy production systems. The architecture of IDEAS derives from CoBASA, a multi-agent architecture for shop floor control [17]. In CoBASA, agentified manufacturing components form dynamic coalitions that are regulated by contracts. Creation and modification of coalitions do not require programming, but only configuration in terms of changes to the associated contract. However, the coalition formation process is not efficient because it is not automatic and involves a number of interactions with the user, with limited support from the system.

Agent-based systems have also been employed for the reconfiguration of real-time distributed control systems. In [18], an intelligent approach to dynamic reconfiguration is applied in real-time environments based on the IEC 61499

function block model. An interesting characteristic of this solution is that it exploits the holonic nature of IEC 61499 model in terms of modularity and recursiveness. The aim of this research is to shorten commissioning times and guarantee more responsiveness to disruptions. However, the IEC 61499 function block standard is not yet widely adopted in industry and, in particular, it has not been implemented on legacy systems.

The authors in [19] propose a new manufacturing paradigm, called Grid Manufacturing, which allows products to negotiate directly with a grid of reconfigurable manufacturing systems (RMS) for a more flexible reconfiguration. The communication is provided by a multi-agent system, where agents represent the products and the machining systems. The route of products through the grids is calculated by translating mandatory product operations into agent understandable commands to operate the system. However, the process is controlled by the agent society, which is not permitted by our industrial requirements (see section 2). Moreover, the concept accommodates RMS but not any type of legacy system.

Despite the significant research effort in this domain, there is still a significant gap in developing robust architectural models and methods to allow the wider implementation of plug and produce reconfigurable manufacturing systems that can use heterogeneous multi-vendor standard modules and be applicable to legacy environments. Furthermore, there is a lack of architectural approaches that can support seamless system adaptation after a plug and produce activity including reassignment of tasks after a machine breakdown or a machine replacement [20]. Most of the reported approaches for system adaptation rely on modified hardware to host the agent technology, whilst modern industrial environments require more streamlined approaches where, for instance, processes relying on real time execution are controlled by PLCs.

The paper reports on a new agent-based architecture to enable plug and produce configuration of industrial production systems, which supports a reconfiguration methodology for task allocation. The main architectural requirements informing the research are presented in Section 2. The plug and produce reconfiguration

methodology and agent-based architecture are discussed in Section 3. The reconfiguration scenario applied to an industrial demonstrator is described in Section 4. Conclusions and future work are in Section 5.

2 Industrial Requirements for Plug and Produce System Architecture

This section describes the requirements that shaped the software architecture. Within the European project PRIME [21], a three-day facilitated method was used to capture the requirements of three industrial partners. This method is called Requirements Workshop and it is designed to engage system stakeholders to elicit the driving industrial requirements. Stakeholders are individuals on whom the system has a significant impact [22]. The workshop was an opportunity to gather stakeholders together to provide input about their needs and expectations, with respect to key requirements that were of particular concern to them [22]. Requirements for a system come in a variety of forms, such as: textual requirements, mock-ups, existing systems, use cases and user stories. No matter the source, all requirements encompass the three categories [23] of Functional requirements, Constraints and Quality attribute requirements.

The **Functional requirements** state what the system must do, and how it must behave or react to runtime stimuli. The functional requirements of the three companies are summarised below:

- Production components, like machines or robots, are closed production components. In other words, the sensor and actuator structure is not visible by an external system (e.g. the agent system). The production components are controlled by the PLC and the control logic is not modifiable by an external system. However, it is possible to download an offline-tested PLC configuration from a database to the controller.
- Once a production component is plugged into or out of the system, the agent society has to detect the change. This information must be made available within the system, so that agents can react (e.g. by changing the PLC configuration).

- The system has to cope with *hot* and *cold* plug and produce activities. Hot plug and produce takes place during normal operation. This could happen, for example, in case of machine breakdown or cable break. Cold plug and produce is the scenario where the system has been informed by the operator before the physical plug (e.g. to prevent injury).
- The production system has to reconfigure itself after the introduction or removal of production components. This includes the assignment of production tasks to production components.

The **Constraints** are design decisions with zero degrees of freedom. The principal constraint is that the architecture for enable plug and produce must be applicable to industrial standard technology. In the case of legacy production components, which use hardware that is not capable of hosting the agent technology, an external controller with the agent society running on it has to be integrated to interface the production components.

The **Quality attribute requirements** (Non-functional requirements) are qualifications of the functional requirements. They detail specific scenarios in which certain characteristics of the system are expected. Table I contains examples of quality attribute requirements.

TABLE I: Quality attribute requirements

Quality attributes for the system integrators case	
Stimulus	Plug/unplug a module (physical or logical)
Response	The system reconfigures the assembly system in terms of reassignment of tasks and the layout representation on the HMI
Source of stimulus	Cold plug and produce: Operator Hot plug and produce: Any failure, e.g. machine breakdown
State of the Environment	The production system is operating
Quality attributes	- The HMI refreshes the system layout within 2 sec - The assembly system's components are configured within 15 sec

	- The integration of new production components is quicker than without plug and produce - After a machine breakdown the system reconfigures itself quicker than without plug and produce

3 Design of the multi agent-based architecture for plug and produce

3.1 Systems reconfiguration approach and methodology

An assembly system is defined in terms of its resources, subsystems and the capabilities associated with those resources. The products to assemble are defined by a product specification, which includes the definition of the sequence of assembly operations. Resources are defined here as pieces of equipment with skills to perform tasks of the assembly process: (1) by putting parts together, (2) by enabling other resources to put parts together or (3) by providing information about an assembly or subassembly, for instance, by testing it. These can comprise shuttle systems, robotic arms, grippers, etc.

Let r_k denote a resource k in a production system $\boldsymbol{R}$. Each resource has an associated set of capabilities. A capability is defined as a set of operations the resource can perform to complete an assembly task. A capability of a resource r_k is formally denoted as $\boldsymbol{C_K}$ and comprises a set of (sub)capabilities $c_i^k \in \boldsymbol{C_k}$. A capability is a name and a set of associated parameters that instantiate that capability for a particular resource. This set contains hard parameters and soft parameters, where hard parameter must be fulfilled to perform a task like grip, while soft parameter can correspond to the quality of the task and are not mandatory, like compliance of tolerances.

To represent the complete capabilities of the system, we define a subsystem as a set of resources needed to perform a common set of operations required to complete an assembly task. For example, a robot arm and a tool rack holding grippers used by the robot form a subsystem. Subsystems and their associated

capabilities can be depicted in a tree structure, where leaves are the capabilities of the resources and the root is the capability of the subsystem.

Once, a resource is plugged in or out of the system, the agent society detects the resource as described in Section 3.3 and aggregates subsystems by the capability management agent, which then updates the available capability list. To reallocate tasks to available capabilities, tasks are matched against subsystem capabilities by both name and parameters set, where the hard parameter of the task must match the parameters of the capability. For the remaining capabilities that fulfil the hard parameter, we introduce a fitness function in order to select the fittest capability among a set of competing capabilities that can accomplish a given task. We define $\widetilde{\boldsymbol{D}}_{\boldsymbol{j,m}}^{\boldsymbol{l}}$ (with $d \in \widetilde{\boldsymbol{D}}_{\boldsymbol{l}}$) as the set of soft parameters of capability c_j of subsystem s_l with respect to a task t_m, for which c_j satisfies all its hard capabilities. The fitness function of capability c_j relative to task t_m is defined as follows:

$$F(c_j, t_m, s_l) = \frac{1}{|\widetilde{D}_{j,m}^{l}|} \sum_{d \in \widetilde{D}_{j,m}^{l}} f_d \tag{1}$$

Where $f_d \in [0, 1]$ is the fitness value of soft parameters d of capability c_j relative to task t_m on subsystem s_l. The capability with the highest fitness is chosen.

The overall reconfiguration methodology for a production system is shown in Figure 1.

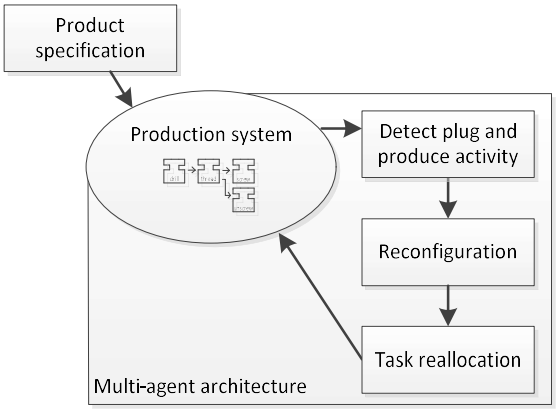

Figure 1 Reconfiguration methodology

The steps involved in a common plug and produce activity and system reconfiguration are summarised below (the agents are introduced in the next

section):

Step 1: Define the full set of the product specification and inform the system
Step 2: Select a product on the HMI and start production
Step 3: Calculate the fitness of the available capabilities for the tasks in the product specification, using (1)
Step 4: Allocate tasks to the “fittest” capabilities and inform the associated agents
Step 5: The agents download user defined PLC configurations from a database to the controllers associated with the selected capabilities
Step 6: If the agent society detect a plug and produce activity, it informs the operator on the HMI
Step 7: Repeat steps 3-6

An example of the application of the above algorithm for a plug and produce scenario is described in section 4.

3.2 General overview of the architecture

The software architecture for a plug and produce assembly system is formalised as a set of structures needed to reason about the system, which comprises software elements as well as their interrelations and properties. The proposed architecture is presented in a decomposition architectural style [24] where the production components and the underlying agent functionality is abstracted into modules. Each module contains a group of interacting agents, which provide its functionality.

The presented architecture includes four agent-based core modules which are described below (see Figure 2).

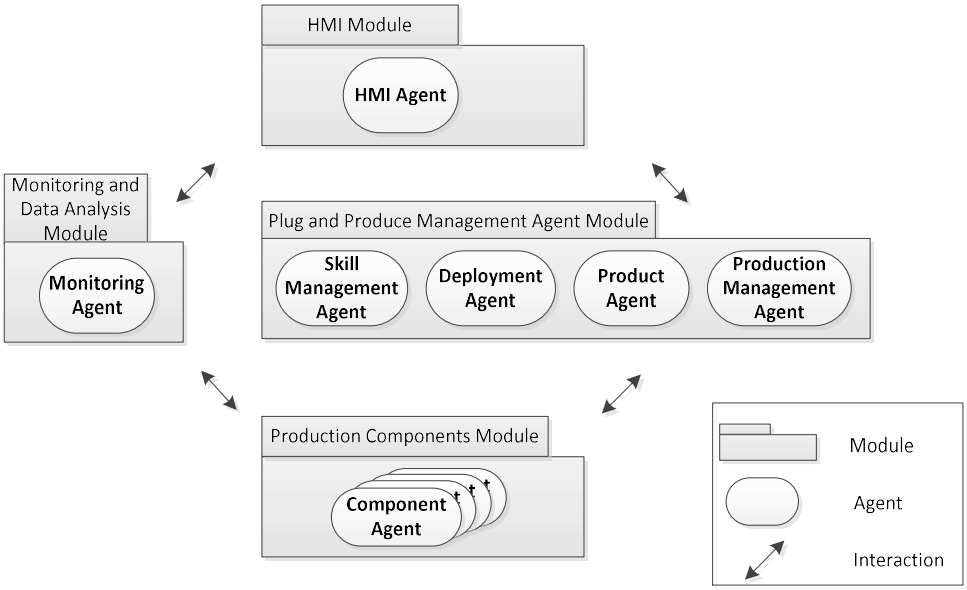

Figure 2 Decomposition view of the architecture

3.2.1 Production Components Module

The Production Components Module is a representation of the production components of a given production system. The agents that populate this module are the Production Component Agents. Any individual Component Agent in this module is associated with a production resource. A Component Agent is the interface between its associated production resource and the multi-agent system.

A Component Agent is linked to its associated production component via the production component's controller. Once this linking is successful, the Component Agent informs relevant modules. Then, it passes relevant data from the production component to the multi-agent system and the other way around. If the communication link to the controller is interrupted, the Component Agent is responsible for reporting this event to relevant modules.

3.2.2 Monitoring and Data Analysis Module

This module monitors and analyses data from the production components. It processes data that come from the Component Agents by calculating averages, medians, and energy consumptions for example. The module is able to write data to a database or directly display information on the HMI. The data in a database can be used for optimisation of key performance indicators for maintenance and

production. In this paper, the agents in this module are not considered further because they do not play any role during a plug and produce operation.

3.2.3 Plug and Produce Management Module

The Plug and Produce Management Module provides the interface between the HMI Module and the Production Component Module. It manages the plug and produce operations of a production component or a product. Therefore, it is aware of the Component Agents that are plugged in to the system and keeps a representation of the current plant layout. Once a new production resource is plugged in, a Component Agent is deployed, according to a saved production component description. The production component description provides access to configuration parameters of a resource. In case of a successful deployment, relevant modules are informed about the newly plugged resources. In addition to a plug and produce activity of a resource, it is possible to introduce a product change. Therefore, the Plug and Produce Management Module can compute feasible products, which can be produced with the given set of plugged production components. If a new product is introduced, the module is able to compute all feasible production layouts and thus, it can support the operator by providing advice and help in managing production resources. In case of a plug and produce reconfiguration activity, the module is responsible for calculating the fitness function guiding the decision making process. The key responsibilities of the Plug and Produce Management Module are delivered by four agents: Capability Management Agent, Deployment Agent, Product Agent, and Production Management Agent. The main responsibilities of each of the agents are outlined in Figure 3.

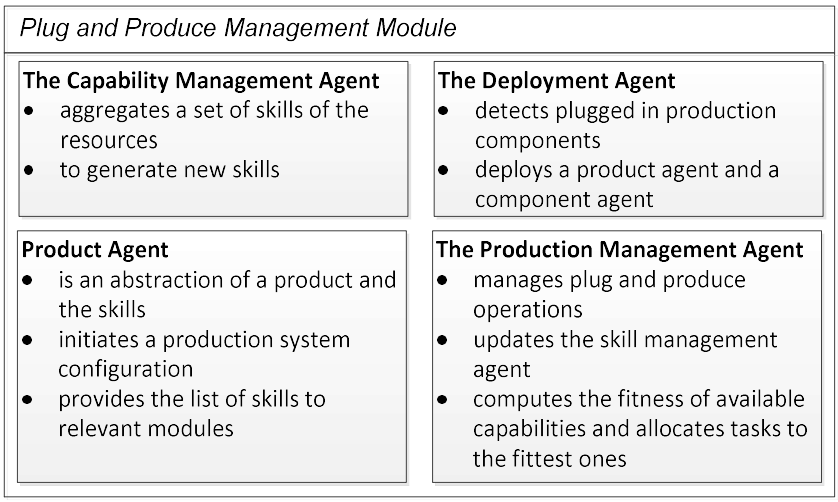

Figure 3 Agents in the Plug and Produce Management Module

3.2.4 HMI Module

The HMI Module accommodates the HMI agent, which interacts with the end user by taking input and delivering information from and to them. Once a production layout changes, the HMI agent delivers updated screens corresponding to the change.

3.3 Agent behaviour for plug and produce

The plug and produce configuration process is based on agent interactions driven by the goal of fulfilling the system functional requirements (see section 2). The agent interactions are described using communication diagrams representing graphs of interacting agents and other elements and annotating each interaction with a number denoting order (see Figure 4). The left hand side communication diagram corresponds to a plug and produce operation. This operation starts by plugging in a resource to the production system, (Step (1), refer to Figure 4). The Plug and Produce Management Module monitors the system, and becomes aware of the just plugged resource (2). The Deployment Agent within this module deploys a Component Agent according to some resource description (3). The Component Agent connects to the resource via its associated controller (4). If the connection is successfully established, the Plug and Produce Management Module gets informed (5) so it can update the capabilities of the production system (6). The HMI Agent asks for the current plant layout (7) and it updates its relevant screens (8).

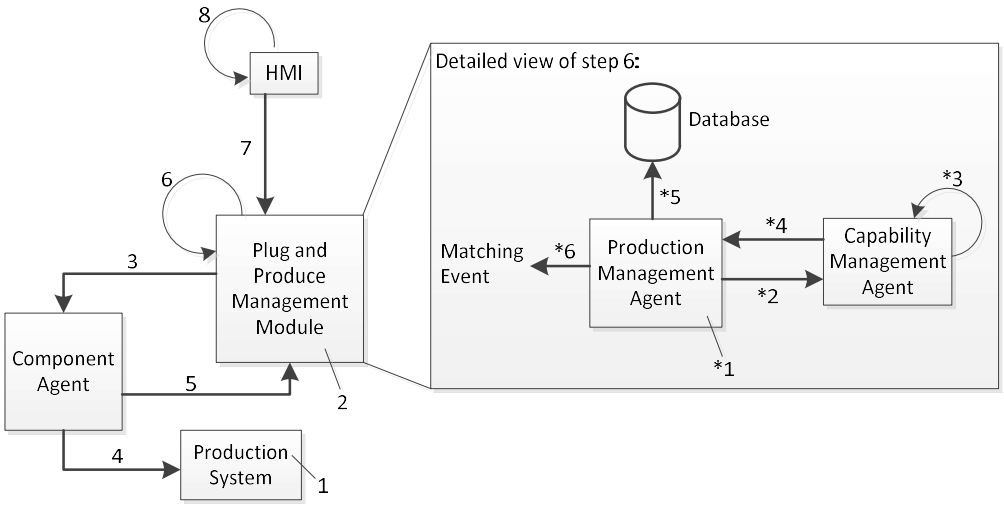

Figure 4: Communication diagram to integrate a new resource and update the capabilities

The right hand side of Figure 4 corresponds to step (6) above. First, the Production Management Agent is informed about the just plugged in resource (*1). Then, the Production Management Agent updates the Capability Management Agent (*2). This update prompts the Capability Management Agent to aggregate the just acquired capabilities into new ones (*3). After this, the Production Management Agent is notified about the aggregated capabilities (*4). Finally, the available capabilities of the production system are compared with the necessary product capabilities on the database to generate production options (*5).

4 A scenario for plug and produce on an industrial assembly platform

We will use a real industrial production system from Feintool Automation, in order to implement the proposed plug and produce architecture. The production system is shown in Figure 5. Although the proposed architecture will be deployed on this assembly platform, it can, however, be implemented on other production systems and layouts.

4.1 The production system layout

The demonstrator to implement the agent architecture on is shown in Figure 5. It shows the production resources and layout. The production system assembles detent hinges, which are used for commercial trucks. It is possible to manufacture product variants, which shows the behaviour of the system by a product change.

The demonstrator contains eight stations; each station accommodates a production component or a fixture. Stations are connected through a transportation system, which is able to carry the product. The assembly platform is modular, which provides an independent mechanical and electrical structure for each station and the transportation system. This independence allows for convenient plugging and unplugging of stations.

Figure 5: Modutec assembly platform

Each station is controlled by a Beckhoff CX5010 embedded PC running TwinCAT 3 Automation Software. The platform also has a Beckhoff CX2030, which is responsible for the transportation system and the overall control of the production system. All controllers within the system are connected to an Ethernet switch.

Each of the eight stations are allocated on a specific plug-and-produce unit (see Figure 6). Stations 1 and 8 are empty. The station 2 contains a fixture to hold the product during assembly. Robot 1 (a Kuka KR 5 six-axis robot) placed at station 3 uses the fixture at station 2 to perform assembly operations. Robot 1 uses five different interchangeable tools required to assemble the entire product, which are placed in a tool changing rack at Station 4. An overview is provided in Table II.

TABLE II: Tool rack overview

Rack Position	Tool	Description
1	Gripper	Mechanical, small
2	Gripper	Pneumatic, small
3	Pusher	Mechanical, medium
4	Gripper	Mechanical, large
5	Gripper	Mechanical, large

Station 5 comprises of Robot 2, which is identical to Robot 1 in Station 3. The fixture to hold the product for robot 2 is located in the Station 6. Both robots share the same tools from the tool changing rack at Station 4, as a result, either robot can execute all assembly steps by utilising the appropriate tools for each operation. Station 7 is an inspection station with two different types of testing units. The first unit uses machine vision to perform a visual test for product integrity and checking the completeness of the assembly. The second unit tests the mechanical properties of the product by applying a set of predefined forces. The stations are served by a transportation system with the positions for loading the parts and unloading the assembled product indicated in the Figure 6. An operator manually loads and unloads the parts and supervises the platform via a Beckhoff CP6901 HMI unit.

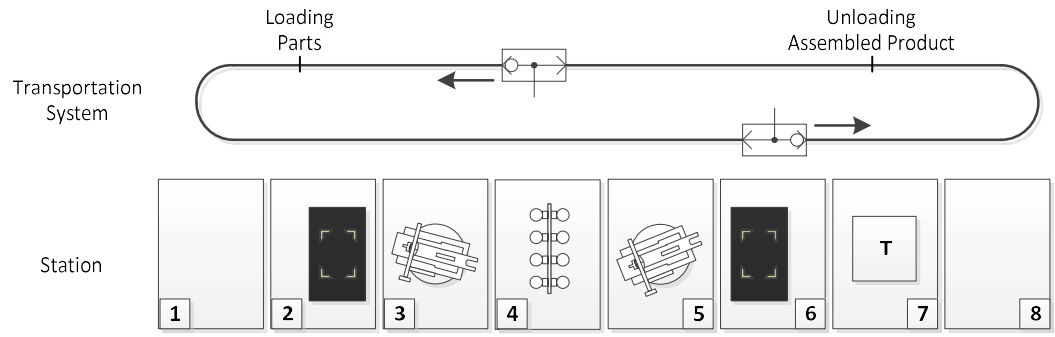

Figure 6: System structure from the top view

The multi-agent system architecture is implemented as a communication and control infrastructure in JADE [25]. JADE has been selected as it supports a peer-to-peer based communication approach for the agents using FIPA semantics as well as providing some basic technical operations to generate, execute, manage and terminate agents.

The multi-agent society is implemented in a distributed fashion on each controller attached to a station. The communication between stations is transmitted over a connected Ethernet network.
The controller CX2030 hosts the Component Agent for the transportation system, the Plug and Produce Management Module, the Monitoring and Data Analysis Module and the HMI Module. The Component Agents for the remaining resources run on the controllers of the associated stations. As stated in the requirements, the system must cope with hot and cold plug and produce activities. Hot plug and produce activities are performed to handle unforeseen events, e.g. machine breakdowns. Cold plug and produce activities are performed after the system has been notified by the user about which change is going to happen.
For example, if a resource is plugged out, the agents in the Plug and Produce Management Module analyse the scenario in order to determine if a suitable configuration of the remaining resources exists to continue production. If this is not possible, in the “cold” use case the operator is informed beforehand. While in the "hot" use case, the agents turn into a fail-safe state and halt production.

4.2 A task assignment calculation scenario

A plug and produce scenario has been implemented to demonstrate the reconfiguration process, including the reallocation of tasks to resources, when a plug and produce activity is carried out with the methodology presented in section 3.

After the production components that are plugged into the system have been identified, the Production Management Agent calculates the value of the fitness function. These values are used to establish if the gripper (tool 1) can perform task “Grip” with the hard parameters Payload (min. 0.32 g) and Width (min. 0.5 mm). Depending on the application, this model can be extended to accommodate additional hard parameters or model them as functions. For example, the acceleration parameter of a robot can be a function of the payload. The soft parameters for the tool are listed and calculated in Table III.

TABLE III: Fitness value for task “Grip”

Task/capability	Task parameter	Capability parameter	Fitness	Accumulated Fitness

Gripper closing time	[0.1 s, 0.2 s]	Min. 0.25 sec	(1/3)*0.5 = 0.167	0.167
Finger position repeatability	0.05 mm	0.07 mm	(1/3)*0.2 = 0.67	0.234
Fail safe	Closes at failure	Opens at failure	(1/3)*0 = 0	0.234

The fitness value for the task “Grip” performed with the capability of tool 1 is 0.234. After calculating the fitness of all current tools, this task is assigned to tool 1, assuming that there is no other resource available in the system that has a higher fitness for this task.

5 Conclusions

A new multi-agent architecture has been introduced to enable plug and produce based configuration and reconfiguration of assembly systems. The architecture and methodology allow the system to detect the removal and inclusion of modules and adapt its capability and behaviour accordingly. The methodology is driven by a fitness function to assess the changes in the capability of the system and allocate tasks to resources. The methodology and architecture have been illustrated using a validation scenario implemented on an industrial assembly platform. The proposed methodology provides an inside into an innovative application of a multi-agent control environment and architecture with the objective of significantly reducing the time for ramp up and deployment of small footprint assembly systems.

Acknowledgments The reported research has been part of the EU FP7 research project “PRIME” (www.prime-eu.com) the support of which is gratefully acknowledged.

References

1. Ueda K A concept for bionic manufacturing systems based on DNA-type information. In: Proceedings of the IFIP TC5/WG5. 3 Eight International PROLAMAT Conference on Human Aspects in Computer Integrated Manufacturing, 1992. North-Holland Publishing Co., pp 853-863

2. Babiceanu RF, Chen FF (2006) Development and applications of holonic manufacturing systems: a survey. Journal of Intelligent Manufacturing 17 (1):111-131

3. Arai T, Aiyama Y, Maeda Y, Sugi M, Ota J (2000) Agile assembly system by `Plug and Produce'. CIRP Annals - Manufacturing Technology 49 (1):1-4

4. Mehrabi MG, Ulsoy AG, Koren Y (2000) Reconfigurable manufacturing systems: key to future manufacturing. Journal of Intelligent Manufacturing 11 (4):403-419

5. Bi Z, Wang L, Lang SY (2007) Current status of reconfigurable assembly systems. International Journal of Manufacturing Research 2 (3):303-328

6. Barata J, Onori M, Frei R, Leitão P (2007) Evolvable production systems: enabling research domains.

7. Onori M, Barata J, Frei R (2006) Evolvable assembly systems basic principles. In: Information Technology For Balanced Manufacturing Systems. Springer, pp 317-328

8. Arai T, Aiyama Y, Sugi M, Ota J (2001) Holonic assembly system with Plug and Produce. Computers in Industry 46 (3):289-299

9. Reinhart G, Krug S, Hüttner S, Mari Z, Riedelbauch F, Schlögel M Automatic configuration (Plug & Produce) of Industrial Ethernet networks. In, Sao Paulo, 2010. 2010 9th IEEE/IAS International Conference on Industry Applications, INDUSCON 2010.

10. Naumann M, Schraft R, Wegener K, Lachello L (2006) Robot cell integration by means of application-P'n'P. VDI BERICHTE 1956:93

11. Marstio I, Haag M, Vaatainen O Self-configuring control system for modular assembly system. In: Assembly and Manufacturing, 2009. ISAM 2009. IEEE International Symposium on, 2009. IEEE, pp 379-382

12. Sugi M, Maeda Y, Aiyama Y, Harada T, Arai T (2003) A holonic architecture for easy reconfiguration of robotic assembly systems. IEEE Transactions on Robotics and Automation 19 (3):457-464

13. Antzoulatos N, Castro E, Scrimieri D, Ratchev S (2014) A Multi-Agent System Architecture for Self-Configuration. Paper presented at the 7th International Precision Assembly Seminar,

14. Zimmermann UE, Bischoff R, Grunwald G, Plank G, Reintsema D Communication, configuration, application - The three layer concept for Plug-and-Produce. In, Funchal, Madeira, 2008. ICINCO 2008 - 5th International Conference on Informatics in Control, Automation and Robotics. pp 255-262

15. Rossmann J, Schluse M, Schlette C, Hoppen M A modular system architecture for the distributed simulation and control of assembly lines based on active databases. In: Iwate, Japan: 9th WSEAS Interna-tional Conference on System Science and Simulation in Engineer-ing, 2010.

16. Onori M, Lohse N, Barata J, Hanisch C (2012) The IDEAS project: plug & produce at shop-floor level. Assembly Automation 32 (2):124-134

17. Barata J (2003) Coalition based approach for shop floor agility–a multiagent approach.

18. Brennan RW, Fletcher M, Norrie DH (2002) An agent-based approach to reconfiguration of real-time distributed control systems. Robotics and Automation, IEEE Transactions on 18 (4):444-451

19. Telgen D, Moergestel Lv, Puik E, Zanten Av, Abdulamir A, Meyer J-J Automatic Structured Decomposition of Manufacturing Actions in an Agent-Based Manufacturing System. In: Web Intelligence (WI) and Intelligent Agent

Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on, 2013. IEEE, pp 155-162

20. Marik V, McFarlane D (2005) Industrial adoption of agent-based technologies. IEEE Intelligent Systems 20 (1):27-35

21. PRIME (2014) PRIME project website. http://www.prime-eu.com/. Accessed 03.03 2014

22. Barbacci MR, Ellison RJ, Weinstock CB, Wood WG (2000) Quality Attribute Workshop Participants Handbook. DTIC Document,

23. Bass L, Clements P, Kazman R (2012) Software architecture in practice. Addison-Wesley,

24. Clements P, Bachmann F, Bass L, Garlan D (2010) Documenting Software Architectures: Views And Beyond Author: Paul Clements, Felix Bachmann, Len Bass, David Ga.

25. JADE (2013) Java Agent DEvelopment Framework. http://jade.tilab.com/. 2013