Interfacing Agents with an Industrial Assembly System for “Plug and Produce”

(Demonstration)

Nikolas Antzoulatos Elkin Castro Lavindra de Silva Svetan Ratchev
Institute for Advanced Manufacturing, University of Nottingham, Nottingham, UK
{nikolas.antzoulatos, elkin.castro, lavindra.desilva, svetan.ratchev}@nottingham.ac.uk

ABSTRACT

We present a fully implemented industrial production system that is reconfigured by a multi-agent system and is controlled by standard industrial control technology. We focus on certain design principles that we found to be crucial in such integrations, and the main challenges we faced when interfacing agents with a real-world production system.

Categories and Subject Descriptors: C.3 [Special-purpose and application-based systems]: Process control systems

Keywords: Manufacturing; capability-based framework; plug and produce; production reconfiguration

1. INTRODUCTION

We have developed a JADE-based multi-agent framework for implementing “plug and produce” on production systems in order to handle changing product specifications, production devices and system capabilities [1]. We define plug and produce based on [2], as a methodology for automatically managing the introduction/removal of assembly devices into/from an assembly system, as well as the introduction of products or product variants. In general, there are many approaches for assisting with the reconfiguration and adaptation of production systems. These frameworks can be classified into methodologies that are based on agent and AI systems, capability and knowledge management systems, and generic frameworks for reconfiguration planning [5]. This paper focuses on the implementation of a capability-based approach, which represents capabilities of assembly devices in a suitably expressive language, and then uses them as input into a decision making process, carried out by a multi-agent system. Our implementation was designed for use on standard industrial control technology such as PLCs (programmable logic controllers), robot controllers and RFID (radio-frequency identification) systems. To the best of our knowledge, plug and produce approaches in which agents are interfaced with off-the-shelf (non-customised) PLCs—the most widely used technology for industrial process control—in a fully implemented, real-world production system are lacking in the literature.[1]

Figure 1: The Modutec highly flexible assembly platform

2. THE ASSEMBLY PLATFORM

The production system, first presented in [1], is shown in Figure 1. This system has been set up by combining eight independent workstations using their provided interfaces; the platform is used to assemble detent hinges for lorry-cab furniture. A linear transfer system serves the workstations via a pallet carrier that holds a pallet with the individual parts to be assembled, as well as the partially/fully assembled hinge. We are currently using six of the eight workstations: two hold a KUKA robot each; two accommodate one workspace each; one holds a tool changing rack; and one workstation hosts an inspection station. We have added RFID tags which identify the various assembly tools to enable a PLC to ascertain which tools are on the rack, and we have extended the platform with the ability to operate it remotely.

The tool changing rack is placed between the KUKA arms, both of which have access to the rack as well as to the workspaces for carrying out the assembly operations. The rack contains six slots that can hold up to six different types of end effectors. We use a selection of six multi-purpose tools—two pneumatic grippers and four two-finger grippers—to perform various pick-and-place operations as well as for inserting certain parts into a hinge being assembled. The KUKA arms are able to dynamically lock/unlock themselves to/from the multi-purpose tools during assembly, yielding a robust, flexible and reconfigurable production line. In particular, if one of the arms fails during production, the other is able to take over and continue production, albeit perhaps at a slower pace. Finally, the inspection station is used to perform a mechanical and a vision test. The former determines whether the force that needs to be applied to break the detent matches the hinge being assembled, and the vision test checks whether the assembly was successful.

The detent hinge that is assembled is composed of two separate leafs held together by a metal pin. Three metal

---

[1] ROS-Industrial (www.rosindustrial.org) seems to be an exception, and a recent positive step in this direction.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

balls need to be placed into adjacent cylindrical slots in the centre of the hinge, three springs need to be placed into the same slots, and a retainer is used to close the hinge. By using only a subset of these parts to assemble a hinge, we can have four product variants, each having a different detent force.

Assembly starts when an operator chooses a hinge to be assembled via a Human-Machine Interface (HMI). This product selection creates a Product Agent (PA), which contains its associated assembly tasks such as “pick-and-place”. The System Agent (SA) matches the assembly tasks of the PA against the current complex capabilities of the assembly system, which are represented by multiple Production Management Agents (PMAs). Complex capabilities are matched as follows. On plugging in an assembly device such as a robot, its associated Component Agent (CA) is launched. This agent registers itself and its associated atomic capabilities with a PMA, which groups CAs that work together and aggregates CAs’ atomic capabilities into complex ones. Tasks achieved by these complex capabilities are at the same level of granularity as the assembly tasks of the PA, which allows the SA to match them against each other. We refer the reader to [6] for full details of the multi-agent system.

2.1 Interfacing with PLCs

Unlike traditional interfaces between agent systems and low-level robot controllers, such as the interface between OpenPRS and Genom [3], and between Jason and ROS [7], which are based on high level communication mechanisms such as “request-response” and “publish-subscribe”, the interface to PLCs is based on more primitive forms of communication. Specifically, the standard way to “send” a request to a PLC, and to “receive” a response, is by respectively writing to and reading from specific data structures in the PLC’s memory. There is the additional issue when interfacing with PLCs in that different hardware vendors have different vendor-specific variants of PLC programming languages. Thus, as a first step, and to gain insights into how we could develop a single, principled interface to such PLC variants, we have chosen the popular Beckhoff PLCs to control the assembly platform. These are industrial PLCs, having the advantage of running on standard CPUs (e.g. Intel) and using standard networking technology such as Ethernet TCP/IP. Beckhoff PLCs can execute C/C++ and Java programs, thereby facilitating integration of agent systems. To this end, we have written an interface that allows an agent developer to read/write some common data types/structures such as integers, arrays, and strings from/to Beckhoff PLCs.

Each workstation has one PLC, which manages all the devices within it. A JADE agent, referred to as the Deployment Agent (DA), runs on each PLC. The DA periodically monitors specific PLC variables to determine whether any devices/tools have been “plugged” into or out of the platform by a (human) operator, or whether the state of the assembly platform has changed in other ways (e.g. security doors in Figure 1 have just been closed). The agent system adapts its behaviour to such changes as necessary. For example, if the DA detects that a device has just been plugged in, it will instantiate a CA for the device. Likewise, if a device is removed from the assembly platform the DA would then “kill” the associated CA. This agent models the associated assembly device such as a robot or a gripper and is responsible for writing configuration commands, via our interface, to the relevant data structures monitored by the PLC, which in turn reads and executes the commands to do the assembly operations.

A novel feature of our framework is that it *uses* standard industrial control technology. This design decision is crucial as it allows operating the assembly system without the plug and produce framework if the need arises, e.g. in the event of a software crash. A related feature in our design is that the agent system only requests tasks from the PLCs down to the level of abstraction of high-level assembly operations. For example, while a CA can request its PLC to perform a force test, it is solely the PLC that refines this task into the more specific steps of moving the hinge/pallet into the test enclosure and pushing the hinge to test its detent force. This is because PLC programs are trusted by industrialists, and generally more robust than agent programs, as the former are often the product of years of industrial use and testing.

2.2 Plugging and unplugging devices

Plugging/unplugging devices correspond to distinct sets of events on the workstations. We outline these events for two devices. Plugging in the linear transfer system amounts to two consecutive events: connecting its Ethernet cable into the associated PLC, and the carrier completing one full round along the track, which is detected via sensors along it. Conversely, the linear transfer system is deemed to have been unplugged when its Ethernet cable has been unplugged. Different events are used to detect whether a tool (e.g. a gripper) has been manually plugged/placed into a slot in the tool rack: the tool must be detected in the rack, all security doors must have been closed, and the emergency lock must have been released afterward. Once the tool rack’s DA detects one or more tools in it, a CA is launched for each.

3. DISCUSSION

We are exploring an alternative architecture similar to [4], in which DAs and CAs run on a separate development board (e.g. a Raspberry Pi) associated to each PLC—rather than on the PLC itself. Such an architecture would, for example, be suitable for PLCs that do not support high-level programming languages such as Java. We have also started to explore a more comprehensive interface to multiple industrial PLCs (e.g. both Beckhoff and Siemens PLCs).

REFERENCES

[1] N. Antzoulatos, E. Castro, D. Scrimieri, and S. Ratchev. A multi-agent architecture for plug and produce on an industrial assembly platform. *Production Engineering*, 8(6):773–781, 2014.

[2] T. Arai, Y. Aiyama, Y. Maeda, M. Sugi, and J. Ota. Agile assembly system by “plug and produce”. *CIRP Annals-Manufacturing Technology*, 49(1):1–4, 2000.

[3] S. Bensalem, L. de Silva, F. Ingrand, and R. Yan. A verifiable and correct-by-construction controller for robot functional levels. *JOSER*, 1(2):1–19, 2011.

[4] E. Hohwieler, E. Uhlmann, M. Niedermayer, and B. Hirsch. Self-organizing production with distributed intelligence. In *International Researchers Symposium on Innovative Production Machines and Systems*, pages 163–167, 2009.

[5] E. Järvenpää. *Capability-based Adaptation of Production Systems in a Changing Environment.* PhD thesis, Tampere University of Technology, 2012.

[6] 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 *IEEE Int. Conf. on Industrial Informatics (INDIN)*, pages 504–510, 2014.

[7] R. Wesz. Integrating robot control into the AgentSpeak(L) programming language. In *Software Agents, Environments and Applications School (WESAAC)*, pages 197–203, 2014.