Comparison of IEC 61499 and Agent Based Control for a Reconfigurable Manufacturing Subsystem

C. Mulubika, A. H. Basson
Department of Mechanical and Mechatronic Engineering,
University of Stellenbosch, South Africa

Abstract
This paper considers two control approaches for a subsystem of a Reconfigurable Manufacturing System (RMS), i.e. IEC 61499 function blocks and agent based control. The two control approaches are compared for the high-level controller, implemented on a personal computer, of a modular Cartesian robot. The paper considers the properties that enhance reconfigurability and shows that the advantages of IEC 61499 are, firstly, relatively easy application when the purpose of the application is well delineated ab initio and, secondly, the modularity in the software units (the latter makes software reconfiguration easy). Agent platforms have the advantage of better development platforms. A challenge for both control approaches lies in directly interfacing with hardware since the implementations used here are Java based.

Keywords
Reconfigurable manufacturing systems, IEC 61499, Agent based control

1 INTRODUCTION

Today's global manufacturing economy is characterized by customers' need for products with higher quality, longer durability, short delivery times, more customization, etc. Too remain competitive, manufacturing companies have to deliver more variations of new products in shorter lead times. The reconfigurable manufacturing system (RMS) paradigm seeks to meet these challenges by offering capacity and functionality changes to a part family when needed [1], with reconfigurable machines as the hardware subsystems at machine or device level [2]. Some technologies in software and hardware have been identified as reconfiguration enablers: in software, modular and open-architecture controls that aim at allowing reconfiguration of the controller; in hardware, modular machine tools aiming at giving the customer more machine options [3].

Considerable research effort in control of RMSs has focused on the use of multi-agent systems and some on the IEC 61499 standard. The multi-agent approach brings the advantages of modularity, decentralization, autonomy, scalability and re-usability, and has been recognized as an enabling technology for designing and implementing distributed and intelligent manufacturing systems ([4], [5], [6]). However, much research work in the agent-based control domain focuses on areas like production planning [7], resource allocation scheduling [8], and application to distributed material-routing control [9]. On the other hand IEC 61499, which enables encapsulation, portability, interoperability and configurability [10], was developed to incorporate demands for flexibility and adaptability in production systems' industrial controls [11]. Many studies using IEC 61499 focused on improving reconfigurability at lower control levels, but a comparison of the two control approaches in enhancing reconfiguration at the same control level has rarely been considered. This paper therefore seeks to evaluate the two control approaches to establish which attributes of each approach enhances reconfiguration in hardware and software components for high-level control (HLC) of an RMS subsystem.

The paper is structured as follows: Section 2 looks at the evaluation criteria for the two control approaches, and Section 3 gives details of the case study. Sections 4 and 5 describe the application of IEC 61499 and agent-based control in the HLC of a modular Cartesian robot. Section 6 evaluates the results of the case study.

2 EVALUATION CRITERIA

The core of the RMS paradigm is an approach to reconfiguration based on system design combined with the simultaneous design of open-architecture reconfigurable controllers with reconfigurable modular machines that can be designed by synthesis of motion modules [3]. The ultimate goal of the RMS is therefore to utilize a system approach in the design of the manufacturing system that allows simultaneous reconfiguration of the entire system, the machine hardware and control software [1],[3].

Reconfiguration in the structure, hardware and software are therefore some of the key evaluation criteria for RMSs. In hardware, machine modules should have clearly defined interfaces in terms of mechanical (e.g. connectors, fasteners), power (e.g. electricity, hydraulics, pneumatics), and informational or control (control network) aspects.

In software, an open-architecture and modularity are the two main criteria [3]. For modularity to be supported in software, the control system itself must be based on the principles of an open architecture. IEEE defines open architecture as: “an open system providing capabilities that enable properly implemented applications to run on a wide variety of platforms from multiple vendors, inter-operate with other system applications, and present a consistent style of interaction with the user” [3].

The overall emphasis in this paper is to enhance reconfiguration. Therefore it is imperative that a control approach for an RMS on any level of control should support the six core characteristics of RMSs [1]: customization (flexibility limited to part family), convertibility (design for functionality changes), scalability (design for capacity changes), modularity (components are modular), integrability (interfaces for rapid integration) and diagonisability (design for easy diagnostics).

3 CASE STUDY

The case study used for the research is part of a reconfigurable assembly cell, which is aimed at producing spot-welded assemblies for domestic circuit breakers. The cell comprises a pallet magazine, a feeder station, a welding station, a pallet-based conveyor, and inspection and removal stations. This paper only considers the HLC of the modular Cartesian robot of the welding station. The robot uses three Festo EGC linear drives, with three Festo CMMP-AS motor controllers controlling three different Festo EMMS-AS motors for the three axes.

A holonic control architecture, based on PROSA ([12], [13]), was used for the cell controller (Figure 1). In this architecture, the welding station is seen as a resource holon, which provides physical manufacturing services to the cell. Some of the information processing part of the holon resides in the cell controller, but most is in the welding station's HLC and low-level controller (LLC), while the physical processing part includes the robot and weld head. The product, order and staff agents of PROSA reside in the cell controller, which is on a different platform. The products' manufacturing process information resides in the product agent and the manufacturing instructions are communicated to the resource agents when required. The resource agents do not retain product knowledge, but offer services that can be used to produce a product to the cell controller. The holons in the cell controller communicate with the welding resource holon using an XML-based protocol over a TCP/IP connection.

The research considered used IEC 61499 function blocks and agents, as alternatives, for the robot's HLC. Both control approaches were implemented on a personal computer, thereby providing a standardized platform for comparison and allowing the use of readily available development environments, respectively FBDK [14] and JADE [14].

Since FBDK and JADE are Java based and therefore cannot directly interface with hardware drivers, the LLC was implemented as a C# program which handled all interfacing with hardware and the associated protocols. The LLC and HLC communicated via a TCP/IP socket connection.

The LLC was used to communicate with the sensors and actuators. The interface is accomplished via the digital inputs and outputs of an Eagle μDAC Data Acquisition unit, which was connected to the computer using USB-2, via the dll provided with the μDAC. The C# program accepts messages from the HLC and activates the appropriate digital output. Similarly, after receiving a digital input from one of the axes, the signal is sent to the HLC.

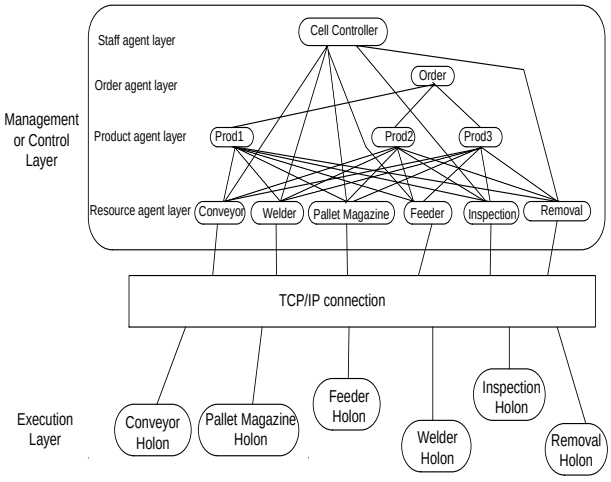

Figure 1 - Holarchy of assembly cell

4 IEC 61499 CONTROL

One option for implementing the HLC is using IEC 61499 function blocks. An IEC 61499 *device* is an abstract model that captures the information processing properties of control devices. These *devices* are hosts to *resources*, which in turn contain the *function block networks* [10]. The function block networks are where the control algorithms are implemented. The IEC 61499 standard defines three classes of function blocks (FBs): basic FBs, composite FBs and service interface FBs [10]. An FB is the building block and encapsulates some behaviour. Like a state machine, the FB has an Execution Control Chart (ECC) which defines the reaction of a FB to each of its possible *input events,* using the *data inputs* associated with each event. The reaction can consist of an algorithm within the FB using also *internal variables* and terminating with *output data* and *output events*, which in turn are connected to other FB's input events and input data. A composite FB encapsulates a network of both basic and other composite FBs connected to external data and event sources. The standard does

not provide for global data, making FBs more reusable thereby easing reconfiguration [11].

Figure 2 shows a layered architecture, with the five layers, used for the control of the modular Cartesian weld robot.

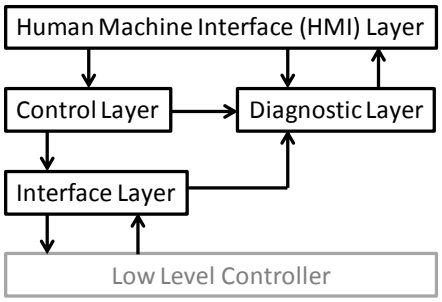

Figure 2 - Layered architecture for welder control

In the human machine interface (HMI) layer functionally similar frame-devices and panel resources from FBDK for each axis were provided for the user to interact with the controller. The frame-devices were used to model the system for each axis. The Control Layer was formed using basic FBs, composite FBs and service interface FBs. For instance, to pass messages within an application, publish and subscriber service interface FBs from the IEC 61499 library were used. Server and client FBs from the library were also used to pass messages between the HLC and the cell controller. Interconnections of FBs were then combined in a composite function block. Three instances of the set of function blocks responsible for the control of each axis were also included in the control layer. For example, the nTask FB receives a command for an axis to perform (e.g. to home). The command is then encoded into XML format in the XML-ENCODER FB and then sent through the COMM FB to the interface layer. The XML message is parsed by the LLC layer to determine which axis is being commanded and what the command is.

The interface layer was used to link the HLC layer, which is composed of the IEC 61499 FBs, with the LLC layer implemented as a C# program. Figure 3 shows the function block used to interface HLC layer with the LLC layer. The LLC and HLC communicated via a TCP/IP socket connection, which introduced two concerns: firstly, the service interface FBs from the IEC 61499 library encodes string messages as ASN.1 expressions [14] (which was not understood by the LLC), and secondly the data input types to the service interface FBs are in WSTRING format. To solve this problem, an algorithm (Figure 4) was developed and used in a COMM FB. The algorithm needs the port and host name separated by a colon in WSTRING format, with data inputs and outputs connected to enable receiving and transmission of data to the LLC. The LLC is a TCP/IP server and therefore must accept a connection from the HLC as a client.

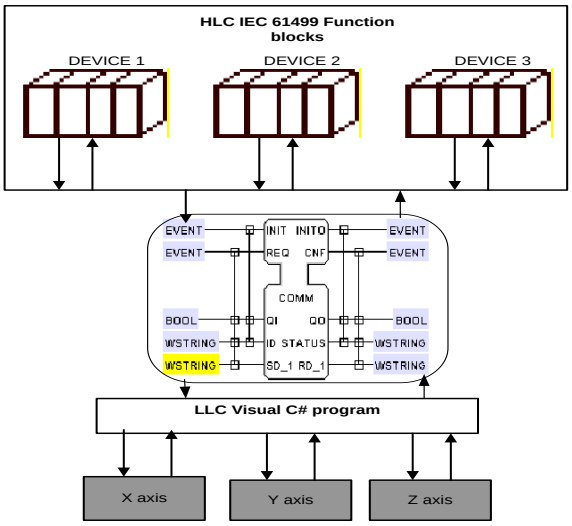

Figure 3 - Interface between HLC and LLC

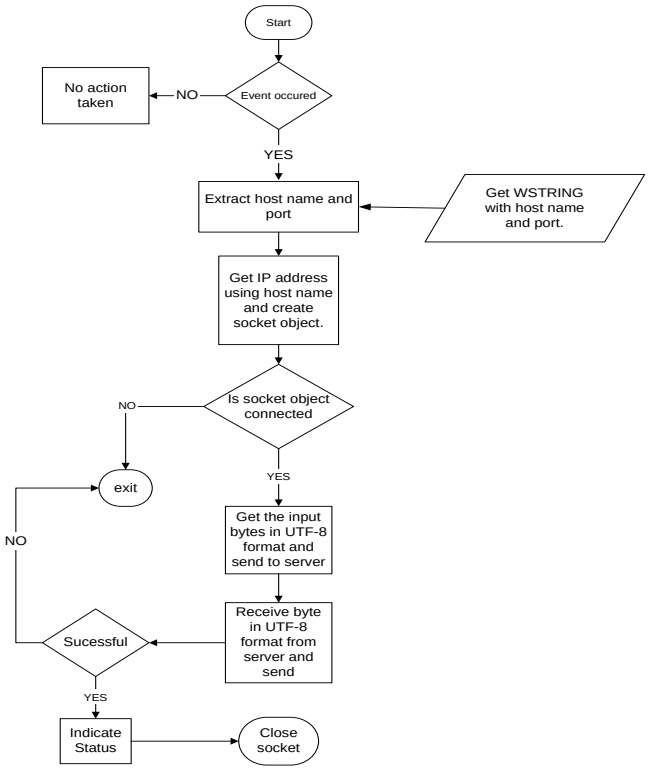

Figure 4 - LLC and HLC communication interface algorithm

5 AGENT BASED CONTROL APPROACH

Agent based control is an alternative approach to using IEC 61499 function blocks for the HLC. Since the welder holon is fairly simple, many of the complex services of multi-agent systems were not needed and the HLC was implemented in a single *CwelderAgent*.

The Java Agent Development (JADE) [15] platform was used to develop and run the agent based control. JADE was chosen because it is well

supported and widely used, thus aiding development and maintenance. JADE is fully distributed in nature and is platform independent since it is Java-based. JADE agents are Foundation for Intelligent Physical Agents (FIPA) complaint making it interoperable with other agents. Moreover, the platform has fully developed debugging tools making testing, debugging and launching of the agents easy.

JADE agents use behaviours to carry out tasks [16] and three basic behaviours are available in JADE, i.e. *OneShotBehaviour*, *CyclicBehaviour* and *GenericBehaviour*. A one-shot behaviour is designed to run once in an execution phase. A cyclic behaviour is designed to repeatedly rerun and never to complete, while a generic behaviour embeds a status trigger and executes different operations depending on the status value. When a behaviour has been defined, it can be added initially in the *setup()* method of the agent class or within another behaviour using the *addBehaviour()* method.

The *CwelderAgent* has to communicate with both the cell controller and the LLC. It is therefore a TCP/IP client of both the LLC and the cell controller. In order to create two client connections in one agent, two *OneShotBehaviour* classes were used as inner classes of an agent. To invoke a behaviour without using the *reset()* method, the *OneShotBehaviour* class is extended and a constructor made. In this way, the behaviour is only invoked when a message is passed to it, unlike using a cyclic behaviour where there is no control when it starts to run. Furthermore, if the *block()* method were to be used with the cyclic behaviour, the whole agent would go to “sleep”.

The constructors for the two extended *OneShotBehaviour* classes take a string passed to it by the message received from the LLC server and a HLC server as *ToInternalServer(String)* and *FromExternalServer(String)* respectively. Then the Java socket communication in blocking mode is implemented in the *action()* method of each *OneShotBehaviour*. The *action()* and *done()* methods are the two abstract methods to be implemented for a class extending the behaviour class. The *action()* method defines the operations to be performed by the behaviour, while the *done()* method returns a boolean value indicating the state of the behaviour. To initiate communication with the LLC server, the operator can click on the button of the GUI, passing a message to the LLC server. The response from the LLC server is then passed to the behaviour within the *action()* to the constructor the handles the HLC server.

6 EVALUATION

Both IEC 61499 FBs and JADE agents are feasible technologies to implement the HLC for the modular Cartesian robot. Since the axes of the robot were not required to be coordinated while moving (linear interpolation was not required), the demands on the HLC was quite moderate. The evaluation therefore can focus on each approach's support for reconfiguration.

As mentioned in Section 2, an open architecture is an important consideration for controllers. Both FBDK and JADE are open systems that are based on Java, thus meeting the open architecture requirement. However, it is an advantage from an interoperability and maintenance perspective if the HLC software is either IEC 61499 or FIPA compliant.

As also mentioned in Section 2, RMSs should exhibit six core characteristics, i.e. customization, convertibility, scalability, modularity, integrability and diagonisability. The evaluation of the control approaches is therefore based on their ability to enhance these six core characteristics. It should be noted that the latter three RMS characteristics are preconditions to achieving the first three characteristics. The latter characteristics are therefore considered first.

Regarding modularity, both IEC 61499 FBs and agents are inherently modular. FBs and agents can both be defined (e.g. for an axis for a Cartesian robot) and then multiple instances can be launched (e.g. one set of instances for each axis). However, since IEC 61499 makes no provision for global variables, the standard is superior to agents in terms of modularity. Moreover, the design of a control device is more standardized in IEC 61499 since the functionality of the different FBs, resources and devices are already specified.

When considering integrability, agents' standardized interaction protocols are an advantage for more complex control systems, but in a simple controller, such as considered in this paper, it is not relevant.

Communication is a central concern in integrability. In this respect, IEC 61499 suffers a setback when used as a HLC since the ASN.1 encoding it uses for string communication over Ethernet is not widely used by other high level languages. When used to connect to the cell controller or LLC, compatible encodings have to be used and therefore custom FBs had to be created. However, when used to integrate with other IEC 61499 FBs, the standard has interfaces available in its library and can be used easily.

Two phases of diagnosability should be considered: firstly during development (including major reconfiguration that requires changes to the control software), and secondly during operation. Regarding the development phase, the authors found it is very difficult to diagnose FB networks since FBDK's (and other available IEC 61499 platforms) debugging tools are rudimentary. To get debug output from a FB network, one has to include print statements in the algorithm or use another network of human

machine interface FBs. Moreover, being an event driven architecture, the flow of events within the FB network is fast and difficult to visualise. On the other hand, agent platforms are much more mature and have good debugging tools.

To diagnose control systems during operations, agent platforms' available tools also help tracing where the problem might have occurred. Further, it is simpler (although not trivial) to include diagnostic provisions (like timeouts) in agent based controllers than in FB networks due to the latter's strict event driven operation. The asynchronous nature of agent communication and non-deterministic operation of agents complicate diagnosing problems during operation of multi-agent platforms, but for the simple control system considered here, this is not a major concern.

Customization to accommodate changes within a product family is inherent in the PROSA architecture used for the cell. Since product information does not reside with the device holons, the robot only had to be able to interpret manufacturing instructions from the cell controller, which were passed in an XML protocol. Both the FB and the agent approaches could handle this communication, and the communication aspects mentioned above are the main distinctions between the two control approaches.

Convertibility for RMSs is more of a concern for hardware than software. Both FBDK and JADE allow for easy conversion of the HLC and the main concerns here are diagnosability, as described above.

Both approaches, due to their modularity, are easily scalable. In IEC 61499 standard, a FB can be re-used by assigning a unique name to each instance of the FB, while with JADE agents, multiple instances of the same agent code can be used.

Although it is not directly related to reconfigurability, a limitation of both approaches should be noted: since they are implemented in Java in FBDK and JADE, respectively, they present some challenges when interfacing with hardware. Java does not support serial port communication in its recent versions and to interface with device drivers in the form of dll files, requires significant effort using JNI tools. In the case study presented here, the Java-related limitations were overcome by introducing the LLC programmed in C#, but that added the effort of developing a communication interface between the HLC and the LLC. An approach to consider in future research is to implement the whole controller in C# using object oriented programming principles.

7 CONCLUSIONS

This paper compares IEC 61499 function blocks and JADE agents as alternative approaches to implement the high-level controller of a modular Cartesian robot. Since the robot is part of a reconfigurable manufacturing system, the reconfigurability of its controller is the main consideration in the comparison of the two approaches.

For the case considered here, which is a relatively simple controller compared to most of the cases published for agent based controllers, it was found that both IEC 61499 function blocks and JADE agents meet the requirements of reconfigurable systems. JADE agents' main advantages are in string manipulation when communicating with other holons and in diagnostic tools. IEC 61499's main advantage is that the standard is explicitly aimed at control with a prescribed structure. Both FBDK and JADE inherit the limited hardware interfacing capabilities of Java.

The authors' experience is further that the development of a typical IEC 61499 application is relatively easy when the purpose of the application is already known. Moreover, modularity in the software units makes software reconfiguration easy and FBDK is simpler to master than JADE. On the other hand, JADE software agents' standard interaction protocols make implementation of complex systems easier. Agent based control has therefore a significant advantage for complex systems, but for a simpler controller considered here where much of agent systems' advantages do not play a role, it still has the advantages of better development and diagnostic tools.

8 ACKNOWLEDGEMENTS

This research was financially supported by Technology Innovation Agency grant AMTS-07-01-P and the gracious access to case study information from CBI Electric is acknowledged.

9 REFERENCES

[1] Koren, Y. & Shiptalni, M., 2010. ‘Design of Reconfigurable Manufacturing Systems.’, *Journal Of Manufacturing Systems,* Volume 26, pp. 130-141.

[2] Bi, Z. M., Lang, S. Y. T., Verner, M. & Orban, P., 2008. ‘Development of Reconfigurable Machines’. *International Journal of Advanced Manufacturing Technology,* Volume 39, pp. 1227-1251.

[3] Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritschow, G., Ulsoy, G. & Van Brussel, H., 1999. 'Recongurable Manufacturing Systems'. *Annals of the CIRP,* 48(2), pp. 527-540.

[4] Monostori, L., Vancza, J. & Kumara, S. R. T., 2006. ‘Agent-Based Systems for Manufacturing.’ *Annals of the CIRP,* 55(2), pp. 697-720.

[5] Jennings, N. & Bussman, S., 2003. ‘Agent-Based Control Systems: Why Are They Suited to Engineering Complex Systems?’ 23(3), pp. 61-73.

[6] Meng, F., Tan, D. & Wang, Y., 2006. 'Development of Agent for Reconfigurable Assembly System with JADE'*. Proceedings of the 6th World Congress on Intelligent Control and Automation*, Dalian, China, pp. 7915-7919.

[7] Pechoucek, M., Thompson, S.G., Baxter, J.W.w, Horn, G.S., Kok, K., Warmer, C., Kamphuis, R., Maric, V., Vrba, P., Hall, K.H., Maturana, F.P., Dorer, K. & Calisti, M., 2006. 'Agents in Industry-The Best from the AAMAS 2005 Industry Track'. *IEEE Intelligent Systems,* 21(2), pp. 86-95.

[8] Shen, W., 2002. 'Distributed Manufacturing Scheduling Using Intelligent agents'. *IEEE Intelligent Systems,* 17(1), pp. 88-94.

[9] Vrba, P. & Marik, V., 2010. 'Capabilities of Dynamic Reconfiguration of Multiagent-Based Industrial Control systems'. *IEEE Transactions on Systems,Man, and Cybernetics-Part A:Systems and Humans,* 40(2), pp. 213-223.

[10] Vyatkin, V., 2007. 'IEC 61499 Function blocks for Embedded and Distributed Control Systems Design'. New Zealand: O3 Neida.

[11] Lepuschitz, W., Zoilt, A., Vallee, M. & Merdan, M., 2011. 'Toward Self-Reconfiguration of Manufacturing Systems Using Automation Agents'. *IEEE Transactions on Systems, Man, and Cybernetics-Part C:Applications and Reviews ,* 41(1), pp. 52-68.

[12] Van Brussel, H. Wyns, J., Valckenaers, P., Bongaerts, L. & Peeters, P., 1998. 'Reference Architecture for Holonic Manufacturing Systems: PROSA'. *Computers in Industry,* Volume 37, pp. 255-274.

[13] Valckenaers, P., Van Brussel, H., Wyns, J., Bongaerts, L. & Peeters, P., 1998. 'Designing Holonic Manufacturing Systems'. *Robotics and Computer-Integrated Manufacturing,* Volume 14, pp. 455-464.

[14] Holobloc. http://www.holobloc.com

[15] JADE (Java Agent Development Environment). http:// jade.tilab.com

[16] Bellifemine, F., Caire, G. & Greenwood, D., 2007. 'Developing Multi-agent Systems with JADE.' West Sussex: John Wiley and Sons Limited.

10 BIOGRAPHY

Chibaye Mulubika obtained his BEng degree in Electromechanical Engineering at the Copperbelt University in Zambia in 2008. He is currently an MScEng Mechanical student at the University of Stellenbosch, South Africa.

Anton Basson obtained his PhD in Aerospace Engineering at Penn State University in 1992. He is a Professor in Mechanical Engineering and co-leader of the Mechatronics, Automation and Design Research Group at University of Stellenbosch, South Africa.