Final draft of: Albert Jakobus Hoffman & Anton Herman Basson (2016) IEC 61131-3-based holonic control of a reconfigurable manufacturing subsystem, International Journal of Computer Integrated Manufacturing, 29:5, 520-534, DOI: 10.1080/0951192X.2015.1067915

IEC 61131-3-Based Holonic Control of a Reconfigurable Manufacturing Subsystem

Albert Jakobus Hoffman and Anton Herman Basson[1]

Department of Mechanical and Mechatronic Engineering, Stellenbosch University, Stellenbosch, South Africa

[b]E-mail: ahb@sun.ac.za, Tel +27 21 808 4250 (corresponding author)Researchers in reconfigurable manufacturing systems (RMSs) have generally used the agent-based control (ABC). Due to industry's hesitance to adopt ABC, this paper evaluates the reconfigurability of a control system developed with industry accepted technologies, i.e. IEC 61131-3 programming languages, a Beckhoff embedded PC and Beckhoff's programming software, TwinCAT. The evaluation focusses on a station controller that controls a reconfigurable subsystem in an RMS. The control system, implemented in an ADACOR-based holonic architecture, was evaluated by conducting reconfiguration experiments using a laboratory case study. This paper shows that a reconfigurable station controller can be implemented using IEC 61131-3 and industry accepted technologies if a hardware platform is used that allows multiple virtual PLCs to be run in individual threads. The control approach presented here can be used to create station control systems that offer optimised cycle times, the benefits of an RMS and the benefits of industry accepted technology.

Keywords: Reconfigurable Manufacturing System (RMS), Holonic Manufacturing System (HMS), IEC 61131, Beckhoff PLC

1. Introduction

In smaller developing countries, like South Africa, industry has an increasing need for manufacturing automation to remain competitive in terms of quality and

[1] Corresponding author: email ahb@sun.ac.za

cost. However, the classical forms of automation are not cost effective for the low volumes and high variance of products that are often produced there. The reconfigurable manufacturing system (RMS) concept potentially provides a solution for this dilemma.

Most research in RMSs has used the agent-based control (ABC) approach. Some researchers even state that ABC is the only way of implementing RMSs (Leitão 2009). RMS researchers use agents because the capabilities of agent-based software and their development platforms make it easier and quicker to test their research (Leitão and Restivo 2006). However, ABC has not found general acceptance in manufacturing industries (Leitão 2009). Although agent software has features that benefit RMSs, they also have features that are unnecessary or unattractive in the industrial environment (Marik 2005; Leitão 2009; Almeida et al. 2010). Since industry is reluctant to adopt ABC, alternatives more acceptable to industry for controlling RMSs, should be sought.

One alternative to ABC that has been researched as platform for RMS control systems, is that based on IEC 61499 function blocks. Mulubika and Basson (2013), as well as Kruger and Basson (2013), have compared this approach to ABC. Although IEC 61499 holds promise for RMS applications, as recently confirmed by Valentea, Mazzolinib and Carpanzanoa (2015), there has been a severe lack of support for this standard by major automation controller vendors and the development platforms to support this approach are not mature enough to be attractive to industry. IEC 61499 further does not make provision for dynamic instantiation, which inhibits its ability to implement holonic control architectures, while these architectures have many advantages for RMS applications, as discussed in this paper.

Industry commonly relies on industrial controllers, such as programmable logic controllers (PLC), to control the machines in the production lines. IEC 61131-3 programming languages, often with OEM-specific customisations, are the industry standard for PLCs. Although these programming languages are widely used and accepted by the industry, researchers have apparently not used them to develop RMSs, presumably because they considered PLCs to be unsuitable for RMS controllers.

The objective of this paper is to reconsider this position, particularly in the context of a station controller in an RMS: the paper evaluates the reconfigurability of a station control system based on industry accepted technologies, i.e. IEC 61131-3 programming languages, a Beckhoff embedded PC and Beckhoff's programming software, TwinCAT.

The next sections describe the case study used as the context for this paper and formulate the main requirements for the control system. Thereafter, Section 4 describes the control approach used here and Section 5 evaluates the control system, followed by an overall evaluation of the approach presented in this paper, in comparison to ABC, and concluding remarks.

2. Case Study

CBI Electric: Low Voltage, a South African company, produces a wide range of electro-mechanical circuit breakers. The parts are mostly manufactured in Johannesburg, while assembly is performed in Lesotho to make use of the low labour rates in that country. Since circuit breakers are safety devices, each circuit breaker is tested as part of the assembly process. The assembly operations and handling operations during testing are currently done manually, but CBI is considering automating some of these operations (particularly testing and labelling). However, due to the large product variety compared to modest (by

international standards) production volumes, conventional automation approaches are infeasible.

A reconfigurable quality assurance cell (RQA cell), aimed at CBI's needs, is being developed by the Mechatronics, Automation and Design Research Group at Stellenbosch University (Hoffman 2014). The RQA cell will do visual inspections, electrical testing, riveting and labelling of circuit breakers. The RQA cell is simultaneously being used as a case study for developing RMS controllers based on industry accepted technologies, so that the industry will be more likely to adopt the RMS concept.

The case study for this paper is the Electronic Test Station (ETS) of the RQA cell. The ETS is the main testing station of the RQA cell and is responsible for electrical testing of all the circuit breakers that pass through the RQA cell. The design criteria and mechanical aspects of the ETS are described in detail by Hoffman (2014).

Figure 1 shows the layout of the ETS. The assembled circuit breakers arrive at and depart from the ETS in fixtures on pallets carried by a conveyor. A six degree of freedom articulated arm robot transports the circuit breakers between the pallets and the test racks. Each test rack houses a number of test slots. A parallel conveyor added to the central conveyor line allows buffering of pallets so that the ETS will not have to wait for pallets. The parallel conveyor has an In Pallet position where the robot can pick up untested circuit breakers and an Out Pallet position where circuit breakers that have passed the tests are placed. To ensure that there will always be a pallet available to place tested circuit breakers, the parallel conveyor has a Buffer Pallet position between the In Pallet and the

Out Pallet. To achieve the required cycle time, the robot uses a gripper that is capable of transporting multiple circuit breakers simultaneously (two for the case study implementation).

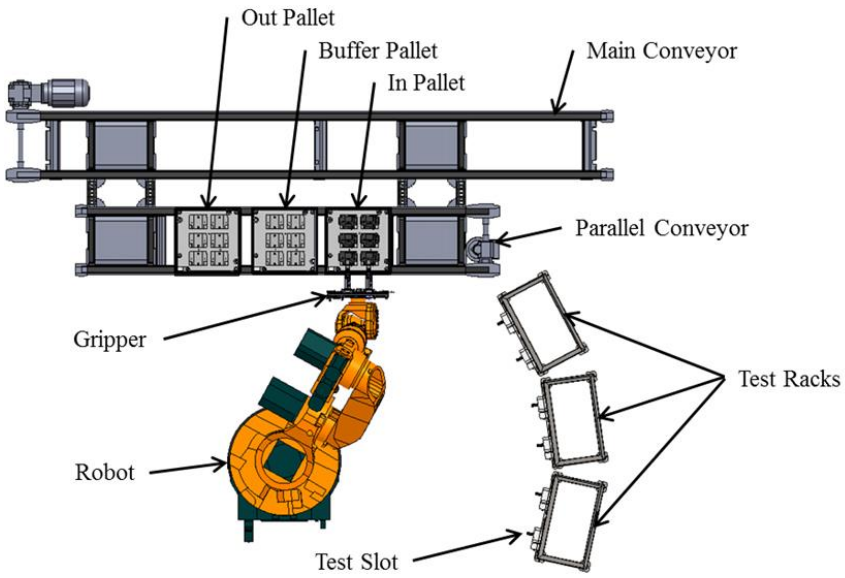

Figure 1. Layout of the ETS.

3. Requirements

Ideally, an RMS should exhibit the six core characteristics given by Koren and Shpitalni (2010), i.e.:

- Customisation: flexibility limited to a part family.
- Convertibility: designed for future functionality changes.
- Scalability: designed for future capacity changes.
- Modularity: system comprises distinct modules.
- Integrability: modules have interfaces suited for rapid integration.
- Diagnosability: designed for easy diagnostics.

The case study discussed above has additional requirements for the ETS and the

RQA cell, which affect both the hardware and software of the ETS (Hoffman 2014). The following additional design requirements apply to the control system of the ETS:

- The ETS has to allow manual override of the control system. In such a situation the test racks have to be accessible to humans to continue production.
- The ETS has to keep the cycle time as low as possible. Therefore the control system has to be capable of using the robot to move multiple circuit breakers simultaneously.
- The ETS has to fulfil the role of an operational holon (described in Section 4.2) from the perspective of the cell, in accordance with the ADACOR control architecture (Leitão and Restivo 2006). The ETS has to communicate with the cell controller using TCP/IP and XML formatted strings.
- Lastly, the ETS has to keep track of the test results of the individual circuit breakers and report the results to the RQA cell controller.

4. Control System Development

A control system was created to adhere to the requirements in Section 3. The control hardware and holonic architecture are discussed first, followed by communication between the holons. Thereafter the individual holons of the control system are described in detail.

4.1. Control Hardware

To maximize the potential of industry adoption, the control system was created in an IEC 61131-3 standard programming language (structured text) to run on a Beckhoff CX5020 embedded PC with the TwinCAT PLC runtime. The embedded PC extends the capabilities of IEC 61131-3, allowing multiple virtual PLCs to run in separate threads on one embedded PC. This effectively makes a holonic control

implementation possible. The virtual PLCs have conventional digital and analogue inputs and outputs like normal PLCs, but can also have inputs and outputs linked to each other, thus creating shared memory which can take the form of data tables.

4.2. Control Architecture

The manufacturing cell that contains the ETS, as well as the ETS itself, uses a holonic control architecture. Holonic control was chosen because it has similar characteristics to RMSs and holonic control architectures are often used to create RMSs (Tönshoff and Winkler 1996; Heikkilä, Jarviluoma and Juntunen 1997; Van Brussel et al. 1998; Brückner et al. 1998; Liu et al. 2000; Chirn and McFarlane 2000; Monch et al. 2003).

Departing from the concept of a holon introduced by Koestler (1967), Van Brussel et al. (1998) defined holons to be self-contained wholes to their subordinated parts and, simultaneously, dependent parts when viewed from the inverse direction. Bell, Rahimifard and Toh (1999) give a thorough review of holonic manufacturing systems. Holonic control architectures are advantageous for RMSs since they divide the control into independent software modules with relatively simple interfaces, through which the holons communicate and collaborate. Further, the parts of the controller that interacts with hardware (called operational holons in the reference architecture described below) are intuitively mapped to hardware modules and, therefore, when the hardware changes, the corresponding changes in the control system can be done with relative ease.

The ETS in itself can also be considered to be an RMS since it too can be reconfigured. Therefore, the control system of the ETS must be amenable to reconfiguration, which leads to the decision to use a holonic control approach for the ETS itself. The holonic control system for the ETS is based on the ADACOR (Leitão and Restivo 2006) control architecture. PROSA (Van Brussel et al. 1998) was also considered, but ADACOR was chosen because it includes explicit provision for optimisation, which is required to minimise the cycle time of the ETS. Figure 2 shows the control architecture of the ETS, as well as the communication structure in the ETS. The dashed lines in Figure 2 indicate the distinct controllers used to host the various holons. In accordance with ADACOR, the holons and their respective roles are (Leitão and Restivo 2006):

- The product holons each represents a product type, containing all the information required to produce the product.
- The operational holons each represent a physical resource that can be used to produce products. In the ETS controller, the robot holon and tester holons are operational holons.
- The task holons each represent an order for producing a product. A task holon obtains the information required to produce the product from the relevant product holon. Thereafter, the task holon employs the services of the relevant operational holons, in the appropriate sequence, to perform the steps required to produce the product. The implementation of the task holons in the ETS, in the form of a task holon manager shown in Figure 2, departs somewhat from classical ADACOR, as described in Section 4.5.
- The supervisor holon's role is to optimise the production plans by guiding the interaction between the operational and task holons.

IEC 61131-3 languages do not make provision for running multiple software threads, but the holonic approach entails that holons can run independently from one another. Therefore, an extension to IEC 61131-3 offered by the Beckhoff embedded PC architecture was used: each type of holon was assigned to a virtual PLC, thereby allowing the holons to run in separate threads on the embedded PC. A library was created of all the functions that the holons have in common, thus simplifying the creation of new holons and modify existing holons. The main differences between the holons are in their main programs and

holon specific functions. The holons' implementations are discussed in the following subsections.

As mentioned in Section 3, the cell controller views the ETS as an operational holon and will communicate with the ETS's controller using TCP/IP and XML formatted strings, naturally using one address and port. This means that one holon in the ETS has to handle all outside communication. Since the communication with the cell controller will require high-level information and will affect the whole ETS, the outside communication function was assigned to the supervisor holon.

Each test rack in the ETS is managed by its own PLC (as shown in Figure 2) and is viewed as an operational holon in the ETC control architecture. This results in a high level split as defined by Hoffman et al. (2013). An alternative would have been to locate the test racks' operational holons on the embedded PC, but this option was rejected because the high level split approach has the following advantages in terms of reconfigurability:

- Integrability: The ETS's controller only has to communicate with a test rack's PLC and not with the individual tests slots. This interface will stay the same even if the test racks change.
- Modularity: The PLC, the test rack and the test slots connected to it form a distinct module.
- Diagnosability: The PLC manages the test racks connected to it and is capable of fault finding the test racks if necessary.

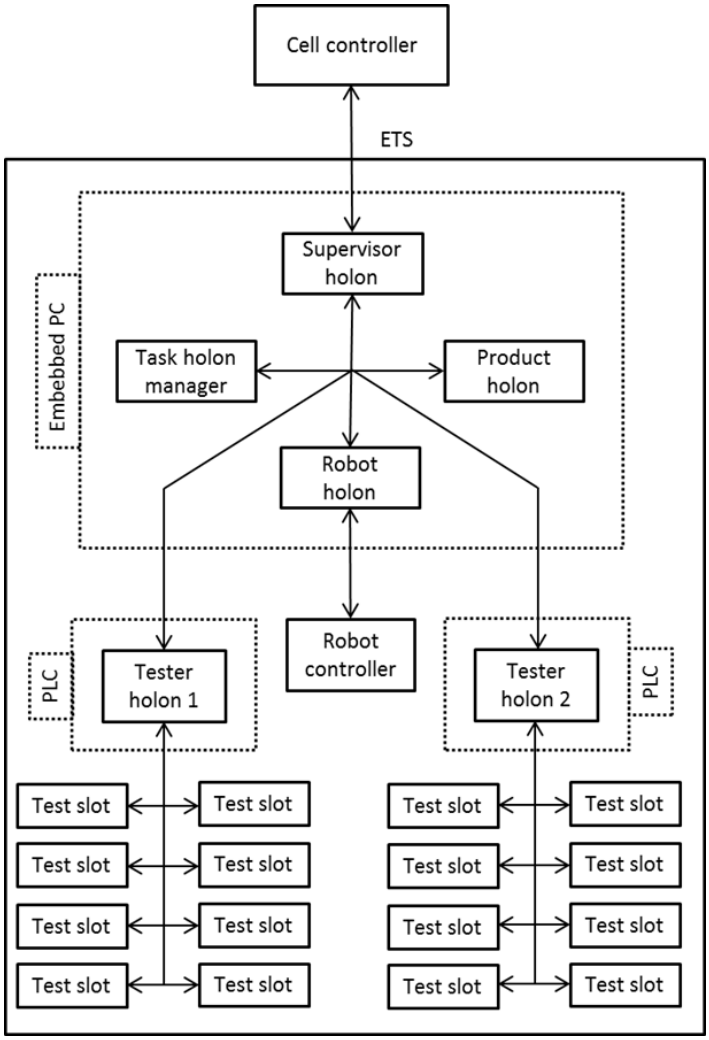

Figure 2. Control architecture of the ETS.

A complete test rack can be reconfigured and tested outside of the ETS, to aid reconfigurability, and to allow the test racks to be used without the task or supervisor holon when the station is in manual override mode.

The location of each test slot, relative to the test rack's origin, is stored in a .csv file on the test rack's PLC. These locations are used by the supervisor holon to determine which test slots to use. When a test slot is changed during reconfiguration, the operator has to update the .csv file on the PLC. Further, when a test rack is moved or a new test rack is introduced, the origin of the test rack will

have to be matched with that of the workspace that the robot designates to the test rack. After doing this, all the locations of the test slots can be used by the robot.

4.3. Inter-holon Communication

Two approaches were used for inter-holon communication: exchanging messages and IO linking. Message exchange is the conventional means in holonic architectures, but since holons are per definition independent, exchanging message will incur latencies: if holon A sends a request for information to holon B, A may have to wait until B has responded to the message, without knowing how long B will take to do so. The ETS controller uses IO linking to avoid these latencies in routine inter-holon communication where the responding holon merely has to send information immediately at its disposal (and make no decisions). The IO linking approach uses data tables, where a data table is an output of one holon and an input of another holon.

The remainder of this section will first describe the IO linking implementations, i.e. the "Work in Progress Table" and the "Parity and Priority Table", followed by the message exchange implementation. All the holons use the message based approach for inter-holon communication, whereas the IO linking is only used by the task holon manager and the supervisor holon.

4.3.1. Work in Progress Table

Since IEC 61131-3 does not allow the dynamic creation of object instances, nor dynamic allocation of memory, a different approach was used to implement task holons, as described in Section 4.5. In this approach, each row in the Work in Progress Table (WIP table) contains the data of a task holon. The size of the table is determined by a variable and can be changed when the controller is in programming mode. Only the task holon manager can make changes to the WIP table.

Each task holon data set in the WIP table, in addition to other information, contains a "command string" data field and an "active command" data field. The command string is an ASCII string variable built by concatenating short strings that represent the process steps required to test the circuit breaker. Since in IEC

61131-3 the maximum string length is 255 characters, the individual process steps must be encoded in short strings. In the case study, the command string comprised a series of characters (e.g. BCZ), where each character (e.g. B) describes a process step that the task holon has to perform. When the task holon starts to execute a command, the first (remaining) character in the command string, which represents the particular process step, is removed from the command string data field and written in the active command data field. When that process step is completed, its character is erased from the active command data field and the task holon remains dormant until the supervisor holon has assigned a priority (in the Parity and Priority Table) to that task holon. The supervisor holon reads the active command fields in the WIP table and will only assign priorities to task holons that have no entries in those fields.

4.3.2. Parity and Priority Table

The parity and priority table (PPT) determines the order in which the task holons are executed, as well as the groups in which they are to work. Grouping of task holons is used to allow the robot to move multiple circuit breakers simultaneously, thereby reducing the cycle time. In addition, the PPT also indicates the optimal positions for the respective task holons to move their circuit breakers to. The grouping of task holons and the methods used to determine the priority of the circuit breakers are described by Hoffman (2014). The PPT is an output of the supervisor holon and is linked as an input to the task holon manager. The task holon manager uses the priority numbers in the PPT to determine the order in which the task holons are to be executed.

4.3.3. Message Exchanges

In the message based approach, the holons communicate with each other by sending XML formatted strings via TCP/IP sockets. An alternative would have been to use OPC for message exchanges (i.e. using IO linking through an OPC server), but XML formatted strings via TCP/IP was chosen because it simplified the implementation of a FIFO buffer for each holon. XML formatting was also

chosen because of its wide acceptance in industry and because the cell controller uses it. Care has to be taken when using this formatting with IEC 61131-3 as the maximum string length is 255 characters. The tags of the XML structure therefore have to be kept short and the sending of unnecessary information should be avoided.

When a holon starts up, it reads a file that contains the IP addresses, ports and device ID's of all the other devices it needs to connect to. This information is then stored internally in a device list table that is created when configuring the holon. The holon then starts its multi-client server and attempts to open sockets to all the devices in the device list. The holon attempts to open the sockets every 2 seconds and will create timeout errors if any connection cannot be established in 20 seconds. This means that each holon has a server part and a client part, which act like an inbox and an outbox, respectively. The server replies to any client that sends messages to it with the same message as confirmation of receiving the message. The communication functions have timeouts and error logging buffers to improve the robustness and assist in debugging.

As illustrated in Figure 3, each holon has one inbox FIFO buffer that accumulates all the incoming messages received via the holon's multi-client server. Further, an outbox FIFO buffer is created for every device in the holon's device list, with each outbox FIFO representing a client connection to a different holon's multi-client server. Figure 3 shows the links between the outbox FIFOs and inbox FIFOs of the various holons. The inbox and outbox FIFOs are checked at the beginning of every program cycle. For example:

- When a part of holon A's control program wants to send a message to holon B, that part adds the message to holon A's outbox FIFO for B.

- At the next start of holon A's program cycle, it checks all its outbox FIFOs and will send the message to holon B, with holon A as a client connected to holon B's multi-client server socket.
- At the next start of holon B's program cycle, it checks its inbox FIFO and, finding Holon A's message there, reads the message and reacts accordingly.

Unique IDs are used to relate a response to a previously sent request, which is essential because several task holons may send similar requests to the same holon, while all task holon communication is handled by the task holon manager.

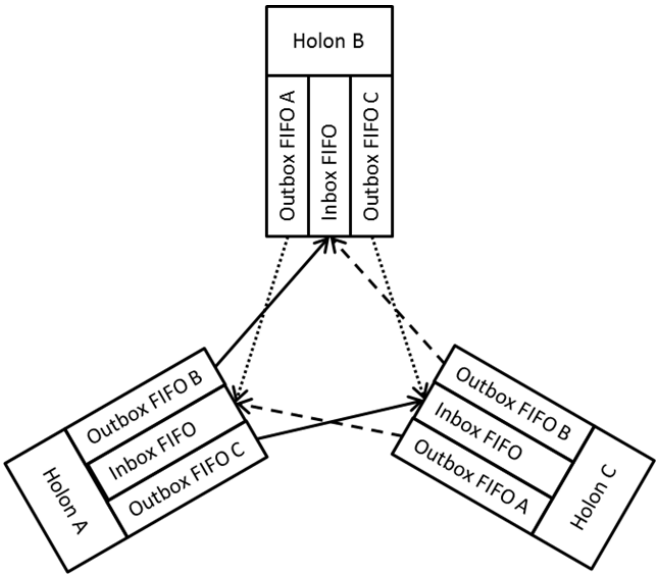

Figure 3. Message exchange architecture.

4.4. Supervisor Holon

The supervisor holon serves as a gateway between the internal holons of the ETS and the cell controller (Figure 2). If a holon in the ETS needs to send a message to

the cell controller, the message is sent via the supervisor holon since the cell controller views the ETS as one of the cell controller's operational holons.

The supervisor holon is also responsible for optimising the performance of the ETS: the supervisor holon uses information from the WIP table (Section 4.3.1) and workspaces table (described below) to maintain the PPT (Section 4.3.2), indicating which circuit breakers to move together, the order in which they should be moved, as well as where they should be moved to. Details are given by Hoffman (2014).

To minimise the ETS cycle time, which is dominated by the robot's movement times, the supervisor holon uses the workspaces table. A workspace is assigned to each operational holon with a physical interface where the robot can pick and place circuit breakers. The origin of each workspace is saved internally in the robot controller when the workspace is calibrated using the robot during ETS reconfiguration. During program start-up, the supervisor holon obtains the origin of each workspace from the robot holon, as well as the position of each slot in the workspace, relative to its origin, from the holon that contains the workspace. This information only changes during reconfiguration, which can only be done offline since the robot calibration has to be done by an operator.

The supervisor holon updates the other information in its workspaces table (except for the positions) once per second by sending requests to all the holons that contain workspaces. The information includes, e.g. whether a slot is open, has been booked or is occupied. If the slot has been booked or is occupied, the identity of the corresponding circuit breaker is also obtained.

At the start of each program cycle, the supervisor holon services the Inbox and Outbox FIFO buffers (as described in Section 4.3.3), including the messages to and from the cell controller. The supervisor holon then updates the workspaces table. When the task holon manager has indicated to the supervisor holon that it is waiting for an updated PPT, the supervisor holon starts its optimisation procedure: the supervisor holon first groups new task holons that have not been grouped yet. The supervisor holon then cycles through the workspaces looking for the group of circuit breakers that are in the best positions to be moved from. When the supervisor holon has found a group, the supervisor holon finds the best position to move the group of circuit breakers to. The supervisor holon then assigns a priority to each of the circuit breakers' task holons in the PPT. Once the supervisor holon has cycled through all the workspaces, it indicates to the task holon manager that the PPT has been updated, which triggers the task holon manager to activate the designated task holons.

4.5. Task Holon Manager

In accordance with the ADACOR architecture, the task holons drive production, while the supervisor holon optimises operations by assigning priorities to the task holons. In the ETS, each circuit breaker has its own task holon that ensures that all the necessary production steps are carried out for that circuit breaker. However, the ETS also requires the grouping of task holons since, to increase the throughput of the ETS, the gripper on the robot was designed to transport multiple circuit breakers at the same time. The supervisor holon determines the grouping of the circuit breakers that can be moved together. This grouping requires that the task holons of the grouped circuit breakers also must be grouped together, and that the grouped circuit breakers move only when all the holons in the group have the appropriate status.

IEC 61131-3 does not allow instances of task holons to be created dynamically, as is usually done in ADACOR. An approach that was considered,

but not implemented, is to create a sufficiently large number of task holon instances at start-up (e.g. as many as there are entries in the WIP table) and to let the supervisor holon activate task holon instances as needed. As fully fledged holons, these instances would each have their own messaging service, IP address and port. These task holons would use variables similar to the entries in the WIP table. In this approach, however, it would be complex to arrange for groups of circuit breakers (each with its own task holon) to be moved together. This approach would also result in increased communication overhead between the task holons and the supervisor holon.

The approach taken in the ETS controller to fulfil the role of all the task holons, is to use only one holon, the task holon manager, with a WIP table in which each row contains the data associated with a task holon, as mentioned in Section 4.3.1. This approach works well in the ETS, since the task holons all have the same decision making logic and only one task holon can initiate production steps (by sending messages) at a given time, to maintain the optimised order given in the PPT, as determined by the supervisor holon.

The task holon manager program cycles through three consecutive phases, like any other holon, i.e. a communications phase, an information processing phase where the information in the WIP table related to the task holons is updated, and a decision making phase where the task holons' work is performed. The communication phase is described in Section 4.3.3 and the other phases are described in the following paragraphs:

Three types of messages are pertinent to the information processing phase:

- If the message is from the cell controller (via the supervisor holon) requiring the creation of a new task holon, the task holon manager sends a message (on behalf of the task holon) to the product holon, requesting the product information required to test the circuit breaker that the new task holon represents. The task holon manager adds all the information related to the new task holon to the WIP table, thereby effectively creating a new task holon.
- If the message is from the supervisor holon indicating that the PPT has been updated, the task holon manager will execute the decision making phase in the particular program cycle.
- If the message type is a "reply to a request" sent by one of the operational holons, the relevant task holon's information in the WIP table is updated accordingly.

The decision making phase is executed in the particular programme cycle only if the supervisor holon has indicating that the PPT has been updated. In this phase, the task holons are activated by the task holon manager, in the order indicated by the PPT, by loading the relevant data from the WIP table and performing actions dictated by the logic stated in the decision making part. The messages that the active task holon wants to send are added to the relevant outbox and will be sent at the start of the next program cycle. The task holon manager will go through the PPT once, activating all the task holons indicated by the PPT. The task holon manager will then wait until the supervisor holon indicates that the PPT has been updated before activating the task holons again.

Although the task holons are not separate software modules, the other holons in the ETS (except the supervisor holon) are not aware of this and communicate with the task holons in the same way as they would have if the task holons where separate modules. Each task holon's row in the WIP table has a unique message ID field that identifies the last message that the task holon sent to

another holon. A message replying to a request by a task holon would include this message ID, thus allowing the task holon manager to associate the reply with the appropriate holon.

4.6. Robot and Gripper Holon

4.6.1. Robot Functionality

The robot holon provides the task holon manager and the supervisor holon with access to the services offered by the robot. The robot holon converts the messages to and from the robot between the inter-holon message strings and the proprietary format of the robot, which was a Kuka KR 16 in the case study. In the case study, the robot holon communicated with the Kuka controller over a Profibus interface between the Beckhoff embedded controller and the robot controller.

The throughput rate of the ETS is largely determined by the robot and the number of test slots. The distance the robot has to travel between the In Pallet and Out Pallet is small, similar to the distances that the robot has to move between test slots. On the other hand, the distance between the pallets and the test racks is large in comparison and, in order to reduce cycle times, this distance should ideally always be covered with circuit breakers in the robot's gripper. The optimal movement cycle for the robot is therefore: move to the In Pallet, pick up a group of circuit breakers from adjacent positions on the In Pallet, move to a test rack, place the group of circuit breakers in adjacent test slots, move to a different position on a test rack, pick up circuit breakers that have completed their testing from the adjacent test slots, move to the Out Pallet, and place the circuit breakers in adjacent slots in the Out Pallet.

When the robot holon receives a request from a task holon to move a circuit breaker, the information is added to the MoveInfo table. This information

includes the pick position, place position, unique message ID and parity (used to indicate group membership). The robot holon waits for the requests of all the circuit breakers that are grouped together (as described in Section 4.5), and then adds the relevant information to one of two buffers: if the request is to move a circuit breaker from the In Pallet, the move command is added to the InPalletFIFO buffer, and correspondingly moves to the Out Pallet are added to the OutPalletFIFO buffer. When a circuit breaker has failed its test, it is also entered into the OutPalletFIFO and a scrap flag is set high. The circuit breaker will then be picked up as normal, but the robot will dump the scrap circuit breaker in a scrap shoot on the way to the Out Pallet. If the scrapped circuit breaker was part of a group, the robot will place the remaining good circuit breakers in the Out Pallet and leave the scrapped circuit breaker's position empty. The ETS control system will not try to fill this slot, since the robot would have to move with only one circuit breaker in the gripper to fill the slot and this would be detrimental to the cycle time.

The unique message IDs of the task holons' request are saved in the MoveInfo table and used when sending messages back to the task holons indicating that the move commands have been completed.

4.6.2. Gripper Functionality

The control of the gripper was integrated into the robot holon because any reconfiguration of the gripper would require that changes be made to the robot holon too. All positioning actions of the robot use the tool centre point as a frame of reference. This is done to keep the robot from crashing into its workspaces and general safety concerns related to robots. By hardcoding the tool centre point into the robot and robot holon, the robot would avoid collisions when invalid move

commands have been issued or a software limit has been reached. These safety features are integrated into the robot's controller.

The decision making part related to moving circuit breakers is the robot holon's responsibility, while there is no decision making role for a gripper holon. The gripper holon would only contain data entries that indicate the locations of the gripper's jaws, how many circuit breakers the gripper can pick up at once, as well a few inputs and outputs. Therefore the gripper holon can be easily integrated into the robot holon. By integrating the gripper holon into the robot holon, changes made to the gripper would automatically be made to the robot holon.

The only other holon that communicates with the gripper is the supervisor holon. The supervisor holon only needs to know how many circuit breakers the gripper can pick up at a time and where the positions of the jaws of the grippers are. This information is used by the supervisor holon to group the circuit breakers. By integrating the gripper's functionality into the robot, any possible delays caused by inter-holon communication between the robot and gripper are eliminated. Even though this delay may seem small when compared to the time it takes the robot to move, these small delays can quickly add up since the gripper is actuated six times to move two circuit breakers through the ETS.

An alternative to the abovementioned approach is that the robot holon has an internal holarchy, which can include several grippers, managed by the robot holon. If a gripper changer was introduced to the ETS to allow the robot to change grippers autonomously, separate gripper holons would have several advantages over the approach discussed above. To allow grippers to be dynamically added or changed, the tool centre point would be set to the point where the grippers are

connected to the robot. The robot would then rely on the gripper holon to keep the gripper from crashing into objects.

The cost of human effort (Hoffman et al. 2013) of a skilled operator to change parameters should be traded off against the development cost of sophisticated control software required to enable autonomous changing of the grippers. The optimal approach depends on the level and frequency of reconfigurations. In the ETS implementation, preference was given to lower initial development costs and the use of skilled operators.

4.7. Product Holon

The ETS has its own product holon that stores all product information need to test the circuit breakers. The ETS's product holon regularly communicates with the cell controller and stores a local copy of all information relevant to the ETS. The information is stored in a .csv file on external or internal storage of the embedded PC. This file can easily be edited using Microsoft Excel to allow the ETS to continue work even when the cell controller fails or a manual override of the system is activated.

An alternative approach that was considered, but not selected, was to remove the product holons from the ETS. The ETS's task holons would then communicate with the cell controller's product holons. This would ensure that the task holons receive the most recently updated information at the cost of higher network traffic to and from the cell controller's product holons. This approach would, however, reduce the robustness of the ETS, because the ETS will be unable to continue work when the cell controller fails or a manual override of the system is activated.

4.8. Tester Holon

Each tester holon manages a test rack, with all the test slots contained in the test rack, keeping track of all the relevant information using a data table. In the ETS, the tester holon sends the test settings to the test slots and receives the test results from the test slots. The tester holon therefore enables holonic communication between the task holons and test slots.

5. Assessments

A number of assessments were performed to evaluate the ETS control system with respect to the requirements of the ETS and the characteristics of an RMS given in Section 3. Some assessments were done by performing the associated operations and are called "experiments" in this section. Table 1 shows which RMS characteristics were tested in each assessment. The first assessment did not require any physical changes to the ETS, in contrast to assessment 2. Assessment 3 considered the failure of a part of the ETS, while assessment 4 considered aspects more specific to the case study.

Since the test slots were still under development at the time of the assessments, the test racks' functionality was simulated by the tester holons for all the experiments. The physical interfaces of the test slots were represented by a mock-up test rack with four test slots. A gripper with 2 positions was used for the testing. The cell controller of the RQA cell was also under development at the time of testing and therefore a simulated cell controller was used in the assessment, i.e. a PC that sent and received XML formatted strings via TCP/IP, as the intended cell controller would have. The simulated cell controller did not, however, implement product holons and therefore the ETS used its own product holons for all product information. The ETS control system's manual override

capabilities could not be assessed, because the test racks are still under development and manual override would require a human-machine-interface (HMI) connected to the each test rack's controller.

Figure 4 shows the laboratory setup of the ETS that was used in the assessment.

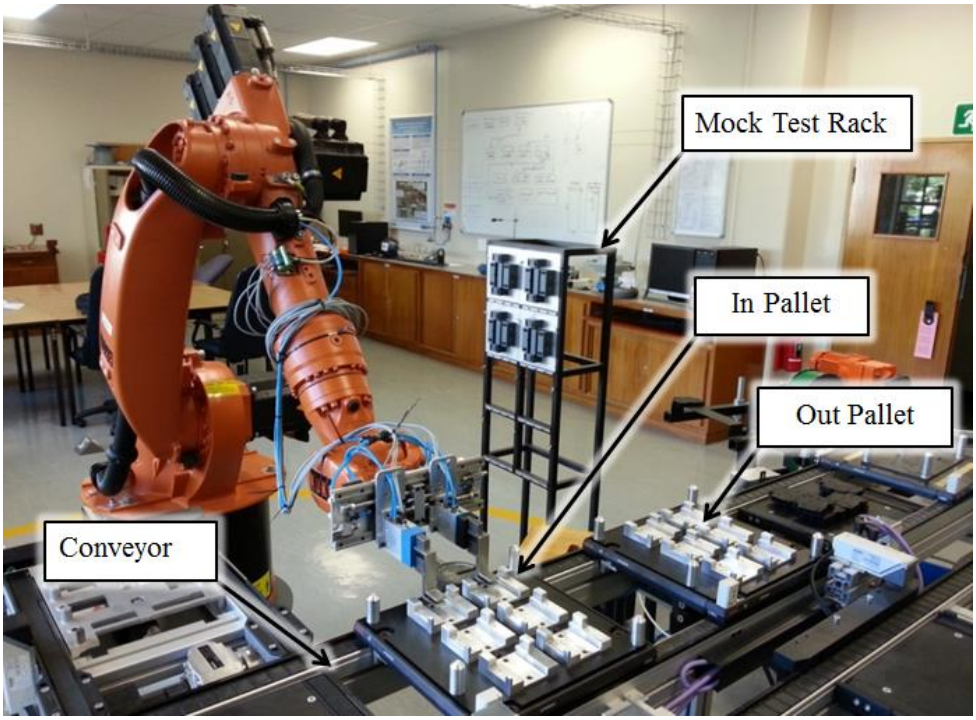

Figure 4. Laboratory set-up of ETS.

5.1. Change within a Product Family

This experiment tested the customizability of the ETS, and in particular the control system's ability to accommodate new products that are within the current product family. The product family was taken to be circuit breakers with a similar enough geometry so that all its members can be handled in a given physical configuration of the ETS. The experiment entailed adding the test settings, product code and command string of the new product to the ETS's product holon. Normally these parameters for the new product would be added to the cell controller's product holon and the information would then be synchronised with the ETS's product holon, but as pointed out above, the cell controller was still

under development and therefore the ETS's product holon was changed directly. None of the ETS's other holons were affected.

The changes to the product holon were done in a few minutes and required changing the csv data file of the product holon. The changes therefore required specific knowledge of the ETS controller and, if suitable instructions are available, can be completed by a trained operator that need only have basic computer literacy. The new settings had to be tested and a ramp-up phase was required. Since there were no changes to the physical configuration of the ETS, the ramp-up phase only required testing a control sample of the new circuit breaker (e.g. 100) and this was completed in 10 minutes.

5.2. Change in Physical Configuration

Physical changes to the ETS may be related to convertibility (such as introducing a new product family or a new technology) and scalability (such as relieving throughput bottlenecks). These changes can be completed efficiently if the ETS has good modularity and integrability. This section considers different aspects that may be encountered during physical reconfiguration. The combination of aspects required in a particular case will be determined by the extent and nature of the reconfiguration.

5.2.1. Change in Origin

This experiment entailed changing the position of a test rack and the In Pallet. The origins of the test racks and pallets are stored in the robot's controller. During reconfiguration only the robot's workspace information need be updated by operators, since the supervisor holon updates its internal workspaces table by polling the devices with workspace information, as described in Section 4.4, without the need for operator intervention.

To change the origin of a workspace, or create a new workspace, an operator has to use a built-in calibration procedure of the robot. This procedure requires an operator to move the tool centre point of the gripper to 3 points on the new workspace, i.e. the origin, a point on the X axis of the workspace and a point

on the XY plane of the workspace. From these 3 points the robot controller calculates the origin and local coordinate system of the workspace. The calibration is done by using the pendant of the robot's controller.

In the experiment, the origins were changed successfully in 30 minutes and required an operator trained to calibrate the robot, typically a person with a technical qualification. After the origins were changed, a ramp-up phase was required to ensure that the robot could pick and place circuit breakers successfully in the changed workspaces. During the ramp-up phase the ETS tests circuit breakers as it normally would, with the exception that the robot moves at a greatly reduced speed while a skilled operator monitors the robot's movement. The pendant of the robot is used by the operator to control the speed of the robot and stop the robot if a possible crash is detected by the operator. This ramp-up phase takes approximately 10 minutes and should be repeated whenever an origin is changed or a new workspace is created.

The experiment proved the ability of the ETS to accommodate changes in the origin of workspaces, as well as the introduction of new workspaces.

5.2.2. Change in Test Slots or Test Racks

If a new product is introduced with dimensions incompatible with the test racks installed in the ETS, the slots would have to be changed or a new test rack would have to be introduced.

An experiment was conducted to test the ability (convertibility) of the ETS's control system to accommodate a change in the test slots: two test slots on the mock-up test rack where moved 10 mm in the negative Y-direction of the test rack's local coordinate system. The test slots were moved the same distance to

avoid requiring a change in the gripper (which is considered in Section 5.2.5). After the test slots' new positions were measured accurately, the workspace information was saved in the tester holon using the test rack's HMI. This reconfiguration was completed in approximately 30 minutes, but the time would depend on the number of slots that have moved. A ramp-up phase similar to the one described in Section 5.2.1 was required after the changes were made to ensure that the robot can pick and place circuit breakers in the test slots without crashing.

The above experiment was concluded successfully with the ETS testing 50 circuit breakers. The control system of the ETS adapted to the changes and used the changed test slots successfully.

Another experiment was conducted by adding a new operational holon in the form of a new test rack. To accommodate the new test rack, the device list of all the holons had to be updated with the address and port of the new test rack. Since all the holons on the ETS's embedded PC use the same device list data file, the new test rack was added to the device list in a few minutes. If suitable instructions are available, the update of the device list can be completed by a trained operator that need only have basic computer literacy. Additionally, the robot's controller also has to be updated with the new workspace and origin of the new test rack, as described in Section 5.2.1. The time and skill required to physically install the test rack was not taken into account as it depends on the design and weight of the test rack. Since only one mock-up test rack was built, the second test rack was added virtually in the experiment and the ETS was tested in a dry run mode. The dry run means that virtual circuit breakers were used and the

sensors in the gripper were modified to always detect a circuit breaker when the grippers are in the closed position.

The assessment was completed successfully with the ETS using two test racks and operating in a dry run mode. A total of 100 circuit breakers were tested successfully during this assessment.

5.2.3. Adding a New Subsystem

This assessment considered adding a completely new operational holon, in the form of a weighing station. The addition of a weighing station would test the integrability, modularity and convertibility of the control system. The weighing station would require changes in several holons: the supervisor holon, the product holon, the robot holon (new FIFO buffer and possibly new robot commands) and the task holon (new commands associated with the new process step). In addition to the changes in the holons, the robot's controller would also have to be updated with the origin of the weighing station. These changes can be expected to take several hours to complete and require a technician familiar with the architecture of the controller. The ramp-up phase will also be required after the changes have been made to ensure the ETS is functioning correctly.

5.2.4. Change in Pallet

An experiment was conducted to examine the effects that changes of the pallet would have on the ETS. The In Pallet was moved to a new position on the conveyor and the pallet was rotated 90 degrees clockwise. Due to rotating the pallet, the robot has to approach the pallet from a new direction to pick the circuit breakers up successfully, thereby simulating the introduction of a new pallet type with the same distance between circuit breakers as in the previous pallet. The workspace and origin of the In Pallet was changed in the robot controller as described in Section 5.2.1. After the ramp-up phase was completed, the ETS tested 50 circuit breakers to confirm it was working as intended. The experiment was conducted successfully and the ETS tested the 50 circuit breakers using the changed In Pallet workspace.

An additional assessment was done to examine what changes are necessary in the ETS when the distance between circuit breakers on the pallet changes. To accommodate a change in pitch, the mechanical end stops that control the pitch of the grippers (Hoffman 2014), have to be changed by a trained

operator. The skill level is determined by the need for precise adjustments. The new pitch of the gripper has to be saved in the supervisor holon, since the pitch of the gripper is used to pair circuit breakers. The changes to the supervisor holon can be done in 15 minutes and, with suitable instructions, can be performed by a trained operator on an HMI.

Further, a ramp-up phase is required to ensure the robot can pick and place the circuit breakers correctly. This ramp-up phase is similar to the one described in Section 5.2.1.

5.2.5. Change in Gripper

An assessment was performed of the effects on the ETS of a change in the number of circuit breakers simultaneously picked up by the gripper. This assessed an aspect of the scalability of the system. Only the supervisor holon and the robot controller have to be updated to accommodate the change in the number of gripping positions: the robot controller has to be updated with the new tool centre point, while the gripper pitch and number of gripping positions variables have to be updated in the supervisor holon. If the new gripper has been used before or is very similar to a previously used one, the changes in the hardware and software can be done in approximately 15 minutes by a trained operator. However, if the gripper has not been used before or has changed significantly, the new gripper would require thorough testing during the ramp-up phase which can take more than an hour. The ramp-up phase of this configuration will probably require a technician.

5.2.6. Summary of Reconfiguration Times

Table 2 summarises the controller reconfiguration times and required skill levels. The time required to reconfigure the hardware is not included.

5.3. Robustness

To test the disturbance handling capabilities of the ETS, the following experiments were conducted: An experiment was conducted to test how the control system reacts to a circuit breaker failing the electrical test and another experiment tested how the control system reacts to a test slot failing. For the first

experiment, one of the testers was programmed to fail all the circuit breakers that it tests. This was done to test whether the ETS scraps the correct circuit breaker. In the second experiment, the tester holon responsible for one of the test racks simulated the failing of a test slot: during the periodic update of the supervisor holon's workspaces table (Section 4.4), the tester holon would report the status of the failed test rack as "unavailable".

Both the experiments were successfully completed and both experiments tested 100 circuit breakers. The first experiment scrapped the correct circuit breaker every time the robot picked up a circuit breaker that had to be scrapped. During the second experiment the control system did not make use of the faulty tester and placed the circuit breakers correctly. These experiments prove the ETS's ability to successfully handle some disturbances.

5.4. Cycle Time

To test the effect of the robot on the cycle time of the ETS, 16 test slots in two test racks were required so that the availability of test slots would not limit the cycle time. The mock-up test rack and a virtual test rack was therefore used in this experiment, with the virtual test rack added to the left (Figure 4) of the mock-up test rack. No circuit breakers were used during the test, because of the use of the virtual test rack. The robot was run at 100% speed and the time the robot takes to place circuit breakers on the Out Pallet was recorded. Video footage was used to determine the time taken by the robot to complete commands that were sent to it. These results were averaged and used to determine the cycle time. Two experiments were conducted, each testing 200 circuit breakers.

For the first experiment, the robot holon sent one command at a time to the robot controller. This means, e.g., that to pick a circuit breaker up from the In Pallet, the robot holon would first send a move command to the robot controller to move the robot to the In Pallet. Once the robot has finished the command, the robot holon would send a command to pick the circuit breaker up. The robot stops after each command which results in a small pause between commands. This

affects cycle time adversely. The individual commands can be seen in Table 3 with the time each command takes to complete.

The first experiment was conducted successfully with the ETS testing 200 circuit breakers. The average cycle time was 9.6 seconds with 2 circuit breakers leaving the ETS in each cycle. This results in a cycle time of 4.8 seconds per circuit breaker, or 12.5 circuit breakers per minute.

For the second experiment, the robot holon sent combined commands to the robot controller. This was done to make use of the advance run feature of the robot controller, which allows the robot to move through several points without stopping. A move-and-place command and a move-and-pick command were created in the robot holon and robot controller to make use of the advance run feature. If the robot has to pick a circuit breaker up from the In Pallet, the robot holon only sends the move-and-pick command. The robot then moves to the In Pallet and picks up the circuit breaker without stopping in-between. The combined commands can be seen in Table 4 with the time each command takes to complete.

The second experiment was conducted successfully with the ETS testing 200 circuit breakers. The average cycle time was 7.5 seconds per cycle for the 200 circuit breakers tested, with two circuit breakers leaving the cell in each cycle. This results in a cycle time of 3.75 seconds per circuit breaker or 16 circuit breakers per minute.

The results show that the combined commands significantly reduced the cycle time of the ETS when the robot is a determining factor. The results also show that the movement between the test racks and the pallets account for a large

portion of the cycle time. This time can be reduced by using a faster robot and by moving the test racks closer to the conveyor.

6. Overall Control Evaluation

The preceding section evaluates the ETS control system's abilities in terms of RMS characteristics and the station's throughput. This section considers the advantages and disadvantages of the ETS control system, as presented in this paper, in the context of a station controller and in comparison with agent-based control (ABC), which is the conventional approach for RMS controllers.

The ETS control system runs on a Beckhoff embedded PC, which is widely accepted in industrial environments. ABC can also use industrial hardware such as industrial PCs, but industrial PCs cost significantly more than embedded PCs. ABC also does not offer the same level of robust interfacing with IO and provides poor control of latency in the control loop.

The reconfiguration times and effort given in the previous section are mostly determined by the holonic control architecture. Therefore, the times and effort required should be similar for ABC and the approach presented here. However, the skill levels required are quite different, as discussed in the following paragraph.

The ETS control system was created using IEC 61131-3 programming languages and the TwinCAT programming software provided by Beckhoff. The IEC 61131-3 languages are the industrial control standard and are used in most PLCs, with the result that most automation maintenance crews are familiar with the languages. The maintenance crews would only need training on how the control system works to be able to make changes to the software. In contrast, ABC relies on JAVA or other object-orientated programming languages and uses

non-OEM development environments to develop the code. The level of programming expertise required for ABC exceeds the abilities of most maintenance crews, thus placing ABC at a disadvantage compared to Beckhoff TwinCAT. However, some proprietary features of TwinCAT were used that might not be available from other OEMs and may not be known to maintenance crews that have not used these features. These features include the ability to run multiple virtual PLCs on one controller and the linking of variables between different virtual PLCs. Further research should investigate how the developed control system can be applied to industrial controllers of other OEM's.

In the context of station control, it is important that the control system has the ability to simply and robustly interface with hardware (such as actuators and sensors), in "real-time ", i.e. with short and precisely known latencies. This ability is provided to the ETS control system by the industrial controller (embedded PC), the IEC 61131-3 programming language implemented in Beckhoff TwinCAT and the IO cards connected to the industrial controller. Access to industrial communication bus standards, e.g. Profibus, is readily and robustly available in the industrial controller. In contrast, ABC relies on additional custom software layers to access IO and is not capable of real-time interfacing with hardware. These additional software layers also add complexity to an ABC system.

Due to limitations of IEC 61131-3, holons cannot be instantiated dynamically. This forces the ETS control system to instantiate all holons when the program starts. Although the lack of dynamic instances can be viewed as a disadvantage in comparison to ABC where holons can be instantiated

dynamically, it can be argued that the requirements and design of the ETS removes the advantages of dynamic instances, as described in Section 4.5.

The ADACOR control approach was used in the ETS control system because it uses a supervisor holon that optimises the sequencing of the task holons, as described in Sections 4.4 and 4.5. The developed control system relies on the supervisor holon to group and optimise the task holons, and therefore the control system becomes dependant on the supervisor holon. All communication with the cell controller also passes through the supervisor holon. This means that if the supervisor holon encounters an error and fails, the control system would also fail. In such a situation the ETS could be switched over to manual override and operators could continue testing the circuit breakers. Although the control system can be changed to run without the supervisor holon, the cycle time would be significantly worse than the cycle time during a manual override.

7. Conclusions

Due to industry's reluctance to accept the RMS concept, this research was aimed at developing a control system that the industry will be willing to accept, in the context of station controllers. Special care was therefore taken to use control hardware and software platforms that have already gained the industry's acceptance.

The Mechatronics Automation and Design Research Group of Stellenbosch University is developing a reconfigurable quality assurance (RQA) cell for a circuit breaker producer. This cell is also being used for various research projects in the research group. The case study for this paper is the electronic test station (ETS) of the RQA cell.

The focus of this paper is to evaluate the suitability of a Beckhoff embedded controller and Beckhoff's TwinCAT software for a controller of a station in an RMS, i.e. the ETS of the RQA cell. To maximise the possibility of the industry accepting the developed control system, only IEC 61131-3 programmes were used for the controller.

The ETS was designed to be reconfigurable internally, incorporating the characteristics of an RMS. A holonic control approach, based on the ADACOR architecture, was therefore used and each holon, with the exception of the task holons, runs in its own thread on the embedded controller. The requirements of the ETS created particular challenges for the control system since task holons had to be grouped and their circuit breakers moved together. The control system uses a supervisor holon to determine which task holons to group, to determine the best positions to move the circuit breakers to and to optimise the order in which the circuit breakers are moved. A task holon manager manages all the task holons in the system, ensuring that the task holons move together once grouped. The functionality of all the task holons were incorporated into the task holon manager using a data table. The task holon manager can successfully provide the functionality of all of the task holons because only one task holon can be active at any given time to maintain the optimal sequence determined by the supervisor holon. This allows the control system to function without the ability to instantiate holons dynamically.

When comparing the IEC 61131-3 approach presented in this paper to the agent-based control (ABC) approach, the following was observed: The IEC

61131-3 approach benefits from the availability of reliable and widely accepted controllers, with robust and simple to integrate IO. Maintenance crews are likely to be familiar with much of the IEC 61131-3 implementation, while experience with ABC is rare, even amongst engineers. However, the lack of dynamic instantiation in IEC 61131-3 languages inhibits the application of classical holonic control architectures. This indicates, therefore, that the IEC 61131-3 based implementations will be better suited to lower control levels, where hardware interfacing is important and control reconfigurations are likely to be directly related to hardware reconfiguration. On higher control levels, where the implementation of complex logic, which is more difficult in IEC 61131-3 languages, is more important and where reconfiguration is more likely to be done by engineers, ABC's advantages will be more important and its disadvantages less important.

This paper shows that a control system for an RMS can be created, in the context of station control, using IEC 61131-3 and industry accepted technologies. The most significant extension to IEC 61131-3 required for the approach described in this paper is that the controller's hardware platform must allow multiple virtual PLCs to be run in individual threads. It was also useful in the case presented in this paper, that the virtual PLCs could exchange data through shared IOs. The control approach presented here can be used to create station based control systems that offer optimised cycle times and the benefits of an RMS in combination with benefits of industry accepted technology.

Further research should be done to investigate how the developed control system can be implemented on industrial controllers of other OEMs. Companies often have a preferred choice of OEM for the industrial controllers they use in their factories. If the control system can be successfully implemented on other OEM's industrial controllers, the industry would be even more likely to adopt the control approach.

Acknowledgements

The contributions of CBI Electric: Low Voltage, who provided case study information, and the funding support of the Department of Science and Technology's ICT initiatives (DST/CON 0089/2014), are gratefully acknowledged.

References

Almeida, F. L., B. M. Terra, P. A. Dias, and G. M. Gonçalves. 2010. "Adoption Issues of Multi-Agent Systems in Manufacturing Industry." *Proceedings of the Fifth International Multi-conference on Computing in the Global Information Technology*, Valencia.

Bell, R., S Rahimifard, K Toh, "Holonic Systems", in *Handbook of Life Cycle Engineering Concepts, Models and Technologies*, edited by A. Molina, J. M. Sánchez, and A. Kusiak, 115-148. 1999. Springer

Brückner, S., J. Wyns, P. Peeters, and M. Kollingbaum. 1998. "Designing Agents for Manufacturing Control." *Proceedings of the 2nd AI and Manufacturing Research Planning Workshop* 40-46.

Chirn, J.-L. and D. C. McFarlane. 2000. "A Holonic Component-based Approach to Reconfigurable Manufacturing Control Architecture." *IEEE Proceedings of the 11th International Workshop on Database and Expert Systems Applications* 219-223.

Heikkilä, T., M. Jarviluoma, and T. Juntunen. 1997. "Holonic Control for Manufacturing Systems: Design of a Manufacturing Robot Cell." *Integrated Computer Aided Engineering* 4(3): 202–218.

Hoffman, A. J. 2014. "IEC 61131-3-Based Control of a Reconfigurable

Manufacturing System." MEng (Mechatronic) Thesis, Stellenbosch University.

Hoffman, A. J., A. H. Basson, and A. le Roux. 2013. "Towards Alternatives for Agent Based Control in Reconfigurable Manufacturing Systems." *Proceedings 5th International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV 2013)* 237-242.

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

Koren, Y., and M. Shpitalni. 2010. "Design of Reconfigurable Manufacturing Systems." *Journal of Manufacturing Systems* 29: 130 - 141.

Kruger, K., and A. H. Basson. 2013. "Multi-Agent Systems vs IEC 61499 for Holonic Resource Control in Reconfigurable Systems." *Forty Sixth CIRP Conference on Manufacturing Systems, Procedia CIRP* 7: 503–508.

Leitão, P. 2009. "Agent-based Distributed Manufacturing Control: A State of the Art Survey." *Engineering Applications of Artificial Intelligence* 22(7): 979-991.

Leitão, P., and F. Restivo. 2006. "ADACOR: A Holonic Architecture for Agile and Adaptive Manufacturing Control." *Computers in Industry* 57: 121-130.

Liu, S., W. A. Gruver, D. Kotak, and S. Bardi. 2000." Holonic Manufacturing System for Distributed Control of Automated Guided Vehicles." *Proceedings of IEEE International Conference on Systems, Man and Cybernetics* 3: 1727–1732.

Marik, V. and D. C. McFarlane. 2005. "Industrial Adoption of Agent-Based Technologies." *Intelligent Systems*, IEEE 20(1): 27-35.

Monch, L., M. Stehli, and J. Zimmermann. 2003. "FABMAS: An Agent-Based System for Production Control of Semiconductor Manufacturing Processes." In *Holonic and Multi-Agent Systems for Manufacturing. Lecture Notes in Artificial Intelligence*, edited by In V. Marik, D. McFarlane, and P. Valckenaers, 258-267. Springer-Verlag Berlin.

Mulibika, C., and A. H. Basson. 2013 "Comparison of IEC 61499 and Agent Based Control for a Reconfigurable Manufacturing Subsystem." *COMA'13, International Conference on Competitive Manufacturing* 283-288.

Tönshoff, H. K., and M. Winkler. 1996. "Shop Control for Holonic Manufacturing Systems." *CIRP Proceedings Manufacturing Systems* 20(3): 277-281.

Valentea, A., M. Mazzolinib, and E. Carpanzanoa. 2015. "An approach to design and develop reconfigurable control software for highly automated production systems." *International Journal of Computer Integrated Manufacturing* 28(3): 321–336.

Van Brussel, H., J. Wyns, P. Valckenaers, L. Bongaerts, and P. Peeters. 1998. "Reference Architecture for Holonic Manufacturing Systems: PROSA." *Computers in Industry* 37: 255 - 274.