Reconfigurable Knowledge-based Control Solutions for Responsive Manufacturing Systems

Alessandro Brusaferri, Andrea Ballarino, Emanuele Carpanzano

Institute of Industrial Technologies and Automation, National Research Council,
via Bassini 15, Milan 20133, Italy,
{alessandro.brusaferri, andrea.ballarino, emanuele.carpanzano}@itia.cnr.it

Abstract: Nowadays, a new generation of responsive factories is needed to face continuous changes in product demand and variety, and to manage complex and variant production processes. To such an aim, innovative self-adaptive automation solutions are required, capable to adapt their control strategies in real-time to cope with planned as well as unforeseen product and process variations. In such a context, the present paper describes an automation solution based on a modular distributed approach for agile factories integration and reconfiguration, integrating a knowledge based cooperation policy providing self-adaptation to endogenous as well as exogenous events. The proposed approach is discussed through its application to a plant for customized shoes manufacturing.

Keywords: Distributed Control, Reconfigurable Manufacturing Systems, IEC 61499, Multi-agent, Semantic Web.

1. Introduction

To face new consumer centered manufacturing paradigms, like mass customization and personalization, factories must be capable to adapt themselves in real time to continuously changing market demand. Thus, the whole production cycle for small or even single batches has to be executed in very short times, i.e. a few days or even hours. In order to properly approach such complex and strict requirements adaptive knowledge based production systems have to be developed. In particular, the conception and development of a new generation of automation solutions, that integrate all factory levels from machines controls up to shop-floor supervision and production planning in a unique real time framework, is mandatory. Future factory automation systems have to be modular, open, agile and knowledge based in order to promptly self-adapt themselves to changing exogenous conditions, like consumers expectations, market dynamics, design innovation, new materials and components integration. To such an aim, a new generation of intelligent, highly-interoperable and self-reconfigurable control systems is a fundamental enabling technology.

To tackle such a challenge, agile manufacturing paradigms - particularly flexible manufacturing systems (FMS) - have been adopted, often proving to be expensive and difficult to manage due to overall complexity. Furthermore, the integration of flexibility capability is not feasible for any kind of application. Therefore, to overcome such barriers and to provide cost effective flexible solutions, Reconfigurable Manufacturing Systems (RMS) have been introduced, characterized by strongly modular architectures and easy reconfiguration capabilities. Therefore, modularity, integrability, diagnosability, customization and convertibility are identified as key features of a RMS [1]. Among these, system modularity can surely be considered the most important property, as outlined in [2] where implications and relationships between the architecture of a logic control system, its modularity and the overall system reconfigurability, are discussed. The problem of agile systems reconfiguration has been faced mainly from the mechanical point of view with the development of easily pluggable mechatronic solutions. Nevertheless, proper solutions addressing a fully modular and reconfigurable control system have still to be identified. As a matter of fact, present automation approaches and architectures - adopted in current industrial practice - are still based on rigid, loosely-coupled solutions, difficult to manage and to adapt, while current methods and tools for control system programming do not effectively support control system reconfigurability [3]. Thus, the integration of a new device within the overall production system, or the replacement of a faulty device, very often requires a critical stop of the system, to perform physical connections and allocations of the new device, as well as partial/total reprogramming of some parts of the control

system, and modifications in production plans, which need time consuming testing and commissioning operations to be executed afterwards [4].

The present paper proposes a self-adaptive control solution in order to support the RMS agility. Particularly, Section 2 briefly analyzes the current state of the art related to the development of self-adaptive control solutions for RMS and summarizes the main features to be guaranteed. Section 3 introduces the IEC 61499 standard exploited for the control solution design. In Section 4 the considered industrial application case is presented. In Section 5 the proposed control solution strategy, architecture and implementation are illustrated, highlighting the exploited paradigms and tools. Section 6 describes how the control solution has been verified by means of simulation based methods. Finally, Section 7 addresses conclusions and next developments.

2. Requirements and Available Technologies

RMS architecture and requirements

Typically, Reconfigurable Manufacturing Systems architectures may be structured into three major hierarchical layers as shown in Figure 1: unit level, cell level and system level. In particular, the overall system is composed by the aggregation of different cell modules, according to the system layout, while each cell module is constituted by the aggregation of more units. Such units can be either operating machines or modules dedicated to parts handling and transportation, e.g. conveyors, rotating tables and manipulators.

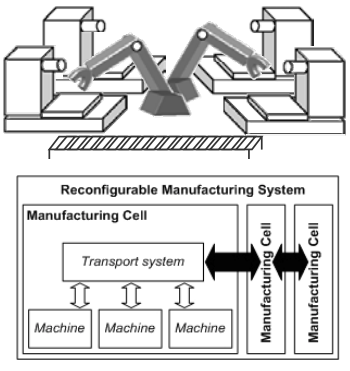

Figure 1. RMS overall architecture.

Starting from last sampled status of underlying controlled objects, each module of the RMS control system - being unit control module, cell control module or system control module – decides and performs its control actions according to the fixed decision policy defined and hard-coded during the control system development phase. The interactions among modules are established *ex ante* and implemented according to fixed bindings among different modules interfaces. Such a strictly coupled architecture is very difficult to modify during RMS reconfiguration phases, as time and costs required for such operations grow rapidly with the increase in system complexity. To such an aim, a new generation of loosely–coupled control architectures, based on a distributed easy-reconfigurable architecture, integrating a flexible knowledge based decision policy, has to be introduced.

Available enabling technologies

Today, the required levels of modularity and distribution of control solutions are not properly adopted in industrial practice due to the lack of well defined and accepted reference models. Major consequences of this lack are twofold. First of all, implemented control and supervision strategies are today typically based on rigid centralized approaches organized into strictly coupled sequences of operations. Difficulties in reconfiguration and in real-time adaptation to production needs are most relevant resulting problems. As a second major consequence, suitable readability, portability and integrability of overall control and automation solutions are not supported. Therefore, the capitalization and reuse of company specific know-how on process and control is very difficult. Flexibility, optimization and failure management features are not properly tackled as well, thus critically impacting the overall production process efficiency and fault tolerance.

Great research efforts have been spent in recent years to conceive a common and well accepted reference model. In such a direction, the multi-agent system (MAS) paradigm shall be mentioned as one the major efforts for development of robust distributed control systems. Such a paradigm is based on

autonomous modules, which integrate knowledge-base for decision making inference, high-level communication protocols and languages to support loosely coupled architectural organizations [5]. Despite its potential capabilities, major limits of such an approach for complex industrial test-cases reside in difficulties of guaranteeing strict execution time requirements.

Recently different research projects, as SIRENA[6], RIMACS[7], SOCRADES[8], have considered an alternative architectural solution for developing systems composed of autonomous and interoperable units: the Service Oriented Architecture - SOA. Such paradigm is characterized by coarse-grained service interfaces, loose coupling between service providers and service consumers, and message-based, asynchronous communication systems [9]. Leveraging the SOA paradigm allows services to be re-used across processes and systems, and systems to be "built for change". Reliability is improved as applications and systems can be made up of tested and proven components. SOA offers the potential to provide the necessary system-wide visibility and interoperability in complex systems subject to frequent changes and operating in a multi-vendor environment. Nonetheless, Service Oriented solutions are still unable to reach the hard-real time constraints in particular for controlling complex manufacturing processes with huge amounts of data and high numbers of units. Furthermore, the decisional logic is not directly supported by SOA, thus, intelligence has to be integrated onto the SOA level. In particular, self-adaptivity needs self-interoperability of information: the knowledge has to be structured in order to be understood by autonomous intelligent agents able to interpret the boundary conditions and to take the proper decisions. For such reason, a major research effort is ongoing widespread to exploit the adoption of Semantic Web approaches into the factory automation domain. Such paradigm is oriented to the adoption of machine interpretable information supporting the implementation of intelligent control solutions based on formal knowledge models. Particularly, the formal definition of classes' properties and instances allows inferring new knowledge from the one already structured into a model.

To cope with real time distributed control, a formal model has been proposed within the IEC 61499 standard of the International Electro technical Committee, also promoted by the international O3NEIDA network [10]. The normative states the common interfaces and structure of the embedded solutions from simple basic function blocks, to composite functional integrations, up to overall control systems applications. It also provides guidelines for the application distribution within multi-vendor control execution devices. Nonetheless, it does not provide structured indications related to self-adaptive control systems design.

Several research actions have been also oriented to the integration of the low level, hard real-time, control layer and the high level, low real-time, control/supervision layer. In particular, [11] propose an interface for the integration of a heterogeneous low level control based on IEC 61499 standard and a Multi-Agent System for the manufacturing domain. [12] proposes the integration of Service Oriented Architecture and a Multi-Agent System (MAS) in order to build a control architecture suitable for automated reconfigurability. [13] introduces a holonic manufacturing control architecture integrated with the logic control layer, designed to improve the agility and reconfigurability of production systems. Despite the performed research efforts and the emerged benefits, such paradigms are currently not implemented within industrial solutions. In fact, the real world applicability needs to be demonstrated through complex industrial test cases highlighting the concrete advantages and providing guidelines for industrial applications.

Such open problems will be approached within the following sections while describing main architectural and functional aspects regarding the proposed self-adaptive control solution. In particular, starting from the process specification, a modular and distributed control architecture has been defined, integrating a real-time IEC 61499 distributed control layer and a multi-agent semantic enriched control and supervision layer. Furthermore, a real industrial plant is

considered as a test-case in order to properly support the description and provide application details. The proposed solution can be integrated within any manufacturing automation system. Therefore, a proper configuration of specific control rules and knowledge base classes is required.

3. IEC 61499 Based Control System Design

In the present approach, in order to achieve the desired agility objectives, object-oriented concepts have been exploited within control system development. To such an aim, the IEC 61499 standard has been adopted as design paradigm due to its orientation to the deployment of modular and distributed control solutions [14]. As briefly introduced within the previous section, the IEC 61499 standard application is based on a fundamental module, the Function Block (FB), which represents a functional unit of software, associated to a hardware resource of the control system, as shown in Figure 2.

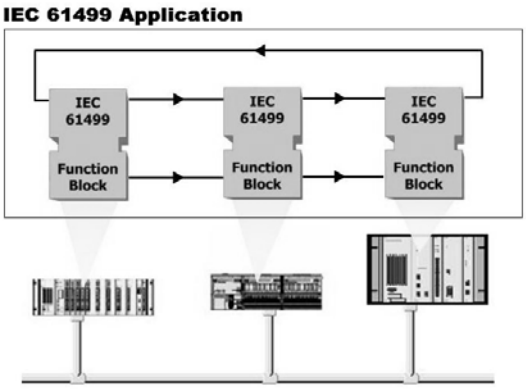

Figure 2. The IEC 61499 distributed model.

A FB instance is characterized by: its type name and instance name, sets of event inputs/outputs and data inputs/outputs, internal data, an Execution Control Chart (ECC) and a set of algorithms, associated with the ECC states. The ECC has an architecture similar to a Petri Net [15], consisting of states, transitions and actions, which invokes the execution of algorithms in response to event inputs. The external interface of a function block is represented in Figure 3.

The execution of algorithms is invoked by the ECC (which is basically a Moore automaton)

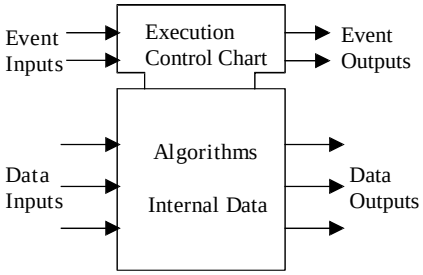

Figure 3. Function block model.

of a FB instance, in response to event inputs. As an example, Figure 4 shows a simple Execution Control Chart. When the execution of an algorithm is scheduled, the needed inputs and internal data values are read and new values for outputs and internal data may be computed. Furthermore, upon completion of execution of an algorithm, the execution control part generates zero or more event outputs as appropriate.

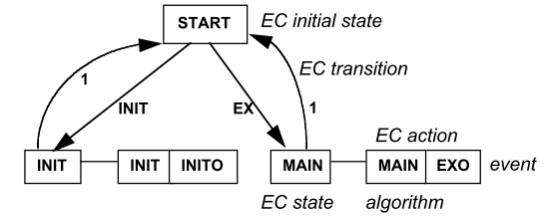

Figure 4. Execution Control Chart.

By properly connecting more FBs an application is defined. Furthermore, a hierarchical approach can be adopted by connecting and encapsulating basic function blocks into composite function blocks. The architecture of composite function blocks will be detailed within the next sections. Regarding configuration aspects, an application can be distributed among several control system devices. A device uses the causal relationship specified by the application to determine the appropriate responses to events. Furthermore, in the IEC 61499 standard a resource is considered to be a logical subdivision within the software (and possibly hardware) structure of a device, which has independent control of its operations. Each FB instance is associated to one single resource. With given definitions, the architecture of a manufacturing automation system can be modeled as a collection of devices, divided in resources, interconnected and communicating with one another by means of one or more communication networks, while the functions

performed by such a system are modeled as applications.

The adoption of the IEC 61499 standard as formal reference model enhances the definition of reusable models, since the principles of modularity, encapsulation and standardization of interfaces are strongly exploited. In fact, the encapsulation of the control functionalities into a network of interconnected function blocks provides an effective high-level view of the application, supporting agile integration and (re)use of the developed control solutions. Besides, modularization implies the possibility of developing software code in different control sub-programs, communicating by means of suitable software interfaces, so having smaller and more manageable modules and related software programs. The modules can be structured on different hierarchic levels, according to a top-down functional decomposition approach. Moreover, based on encapsulation principle, every module hides its internal algorithms and variables to the other components of the control system, so making both the software development and its maintenance easier. Furthermore, the possibility of deriving specialized components allows re-using existing software functionalities by simply extending them with minor functional and software modifications. The IEC 61499 supports also the adoption of an event-driven design approach, so enhancing control solutions manageability and reconfigurability.

4. Industrial Application Case

Before starting the description of the deployed control architecture, the considered real industrial application is presented, i.e. an innovative shoe manufacturing plant managed by ITIA-CNR, see Figure 5. For the sake of brevity, a simplified version of a part of the manufacturing system is here considered.

The focused manufacturing system integrates an innovative transport line for moving the semi-finished shoes from a machining station to another one according to operations to be performed. The transport line innovative molecular structure enhances the modularity, scalability, integrability and reconfigurability properties of the production system, increasing the overall flexibility of the plant.

Figure 5. Shoe manufacturing plant.

The basic element of the molecular structure is the “Tern”, which is constituted by two rotating tables, called “Table” and “Island”, and by a rotating three arms manipulator. The Table moves the semi-finished shoes either to the next Tern or to the Island of the same Tern. Moreover, it moves backward the lasts flowing back towards the warehouse (the last is the object around which the semi-finished shoe is built upon). The Island directs the semi-finished shoes towards the different machining stations, laid around the Island itself. The manipulator carries out the transport of the semi-finished shoes and lasts between Tables and Islands.

Furthermore, each rotating element integrates one equipment necessary to perform the rotation, plus a certain number of pushing devices, namely two on the Table, three on the Manipulator, and an amount equal to the number on connected machines on the Island, for moving the semi-finished shoes from a rotating element to another one. Pushing equipments are managed by dedicated electro - pneumatic valves connected to the Tern controller by means of I/O signals. Furthermore, sensors aimed at monitoring the position between the In and Out state have been integrated. Similarly, each rotating device is actuated by an electrical motor and monitored by position (rotation) sensors connected to the Tern controller I/O channels. Thus, the overall molecular transport system integrates about 300 input and output signals to be properly managed by the control system.

5. Control Solution Development

Control strategy

The present section details the Tern control logic. As introduced within the previous section, the molecular transport system is aimed at moving the semifinished shoes between the production line working stations. In particular, arrows in Figure 6 depict the possible movements that involve a generic semi-finished shoe or last in a generic single Tern.

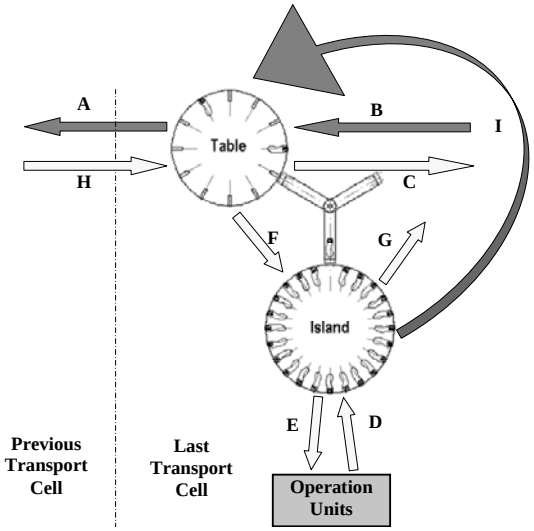

Figure 6. Work pieces flows.

A semifinished shoe that arrives on the Table of a Tern (arrow H) and which has not to be machined by the Tern itself is directly moved to the next Tern (arrow C); otherwise, it is moved to the Island (arrow F), where it is properly machined (arrows D and E). For the sake of simplicity only one machining station has been here represented. As soon as the machining is over, the shoe is moved towards the next Tern (arrow G).

Whenever a shoe is finished, it is removed from its last, and the last itself goes back to the warehouse by flowing back through the whole transport system. So, for a generic Tern, the last that must be stored in the warehouse comes from the adjacent Tern (arrow B) and goes back, through the Table, to the previous Tern (arrow A). Specifically, the lasts start their way back to the warehouse in the last Tern, once the shoe working process is over (arrow I). The proposed control strategy for a generic Tern is based on the following three basic assumptions:

- avoid deadlocks, i.e. avoid situations into which none semifinished shoe or last can be moved;
- favor the backward lasts flow respect to the forward one of the semifinished shoes;
- favor the unloading of the Tern resources, i.e. of the Table, Island and Operation Stations.

Such heuristic rules have been defined to guarantee a correct and efficient use of the system resources. To implement the first assertion it has been decided to have always one slot free on a Table for the last backward movement toward the warehouse. Moreover, to implement the second and third assertions proper priorities have been assigned to the different possible operations illustrated with reference to Figure 6. Such priorities are represented in Figure 7. In the columns of the table the different operations have been listed. The presence of the character "x" in a cross between a column and a row means that the operation associated to the column has minor priority than the one associated to the row.

	AH	A	BFG	BC	BF	BG	B	C
AH	-	X	X	X	X	X	X	X
A	-	-	X	X	X	X	X	X
BFG	-	-	-	X	X	X	X	X
BC	-	-	-	-	X	X	X	X
BF	-	-	-	-	-	X	X	X
BG	-	-	-	-	-	-	X	X
B	-	-	-	-	-	-	-	N.O.
C	-	-	-	-	-	-	N.O.	-
D	-	-	-	-	-	-	-	-
E	-	-	-	-	-	-	-	-
FG	-	-	-	-	-	-	-	-
F	-	-	-	-	-	-	N.O.	-
G	-	-	-	-	-	-	N.O.	-
H	-	N.O.	-	-	-	-	-	-

Figure 7. Tern priority logic.

Notice that the possibility to execute concurrent operations, e.g. operations A and H, is also considered in the table, e.g. operation AH. As a consequence, it is not significant to define a priority between operations A and H, since the concurrent operation AH can be carried out.

Control solution architecture

Within the present section, the implemented control solution architecture is described. In particular, the molecular line has been considered as a set of interacting Terns, each one with its own independent control system. Each Tern control module communicates

with the related Table, Island and Manipulator control modules, and is connected to the adjacent Terns control system modules, to coordinate the exchanges of semi-finished shoes and lasts, as shown in Figure 8.

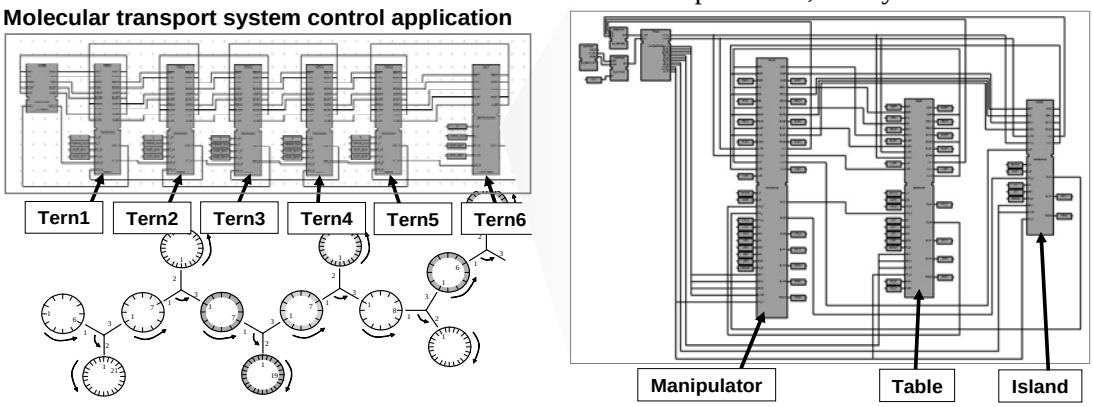

Figure 8. Control architecture of the Molecular transport line.

Furthermore, each basic function block encapsulates the control logic by means of a state machine responsible for the activation of dedicated IEC 61131 based control algorithms depending on run-time events and conditions. As an example, in Figure 9, the Pusher Function Block is shown.

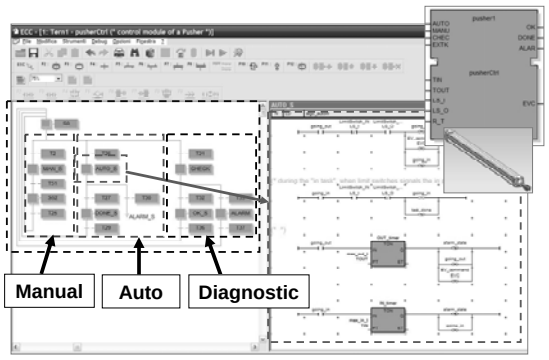

Figure 9. Pusher function block

In particular, the execution control is structured into initialization state, manual and automatic execution, failure and diagnostic state, so as to decouple different running modes. The elementary control modules have been integrated so as to obtain the overall control solution. Therefore, a bottom-up approach has been adopted. The intrinsic modularity of the process to be controlled has been maintained within the developed control application by composing Pusher and Rotation devices Function Blocks and the coordination function block within Table, Island and Manipulator composite FBs in the same way. Figure 10 shows the Table control Function Block.

Besides, the semiworked shoes flow policies have been embedded within the Table, Manipulator and Island coordination function blocks. In particular, an asynchronous event-driven interaction approach has been adopted. For each shoe to be moved from a transport system module to another, the former sends an event to the later. Then, the decision is taken by evaluating the shoes priority and the feasibility of the operation (i.e. target slot free).

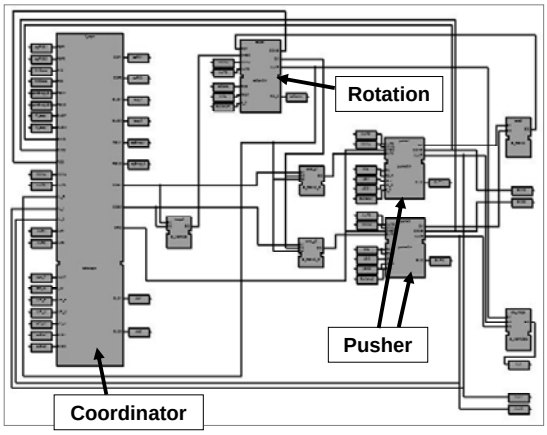

Figure 10. Table composite function block

Moreover, the Tern Control modules represent the low level control layer, responsible for real time control tasks. Additionally, tasks dedicated to manage nominal and failure operating conditions have been integrated. Such control modules have been implemented within the ISaGRAF 5 Workbench [16]. By supporting IEC 61499, such environment acts as backbone for the overall application development, from design to implementation and validation. In ISaGRAF 5 the distributed hardware architecture can be defined by properly

assigning the Resources (Virtual PLCs) executed within each device, and each device integrates I/O interfaces connected to the factory field. The application can so be designed as a unique control program. Furthermore, the developed control application can be generated into target-independent C code, executable by a firmware, which emulates a virtual machine on a hardware target, being it an Industrial PC, an embedded board or a tiny controller. Therefore, portability and scalability of the developed application is guaranteed. In order to provide self-adaptive capabilities to the deployed automation solution while reconfiguring manufacturing resources (i.e. adding, removing operational units or integrating manufacturing functions) a multi-agent control and supervision layer has been integrated onto the object-oriented real-time control layer. In particular, each Tern control module has been connected to a Tern agent, responsible for the interaction within the high level control layer. The interface between the IEC 61499 modules and the multi-agent layer, deployed in Java language, has been developed by means of the Java Native Interface provided by Sun Microsystems Inc. [17].

production orders as well as provide a run-time updated factory image to the management layers. To such an aim, dedicated interaction mechanisms have been implemented.

Furthermore, the self-adaptation mechanism has also to react when operational capabilities become unavailable due for example to failures or maintenance operations. Therefore, a dedicated management agent has been integrated, implementing a knowledge based reasoner connected to a complementary knowledge model based on a common formal domain description:

– *Product knowledge based model*: aimed at the structuring of information related to products to be processed by the manufacturing system, as required operations and priorities with reference to the overall production plan;

– *Resource knowledge based model*: aimed at the formal description of the intelligent units run-time integrated within the production facility, i.e. the operations provided and execution states.

Such models are dynamically updated by means of a dedicated Interface agent, in order to be aligned to the process evolution, supporting

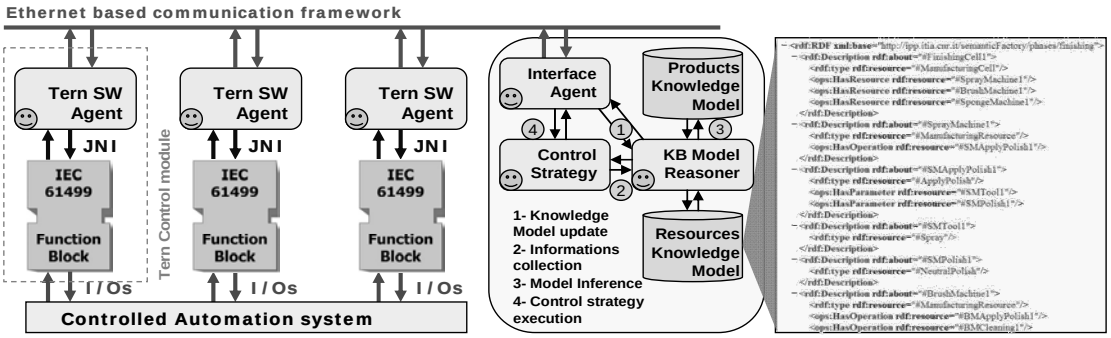

Figure 11. Overall multi-agent control architecture.

The higher level control layer is responsible for the coordination of the low level real-time control layer in response of predicted as well as unforeseen events. The overall control architecture is outlined in Figure 11. In fact, the decisional logic has to be dynamically adapted to follow changing products features/ requirements contextually to machine operational capabilities reconfiguration, tools integrations, and new machines implementation into production cells. The higher level control layer is also aimed at interacting within MES and ERP systems so to receive incoming

the optimal real-time decision strategy. Two dedicated model update mechanisms have been implemented: the first has in charge the management of the products entering/leaving the system and their operational state changes due to processed manufacturing tasks performed within the RMS; the second supervises the resources integrated within the system, updating their available operations depending on their evolving execution states and/or reconfigurations.

Thanks to the adoption of such an asynchronous policy, the model adaptation task is called only when required, reducing the overall execution time. Furthermore, semantically-rich based descriptions have been adopted to implement the knowledge based solution. In particular RDF and OWL-DL W3C standards [18] coming from the Semantic Web area have been considered.

Machine reasoning has been used to perform automatic matchmaking of required and offered services using logical inference, rather than performing hard-coded one-to-one mappings. Such type of matchmaking enables the use of services that did not exist or were not known when the requestor side was programmed, enabling automation system reconfiguration without control logic reprogramming.

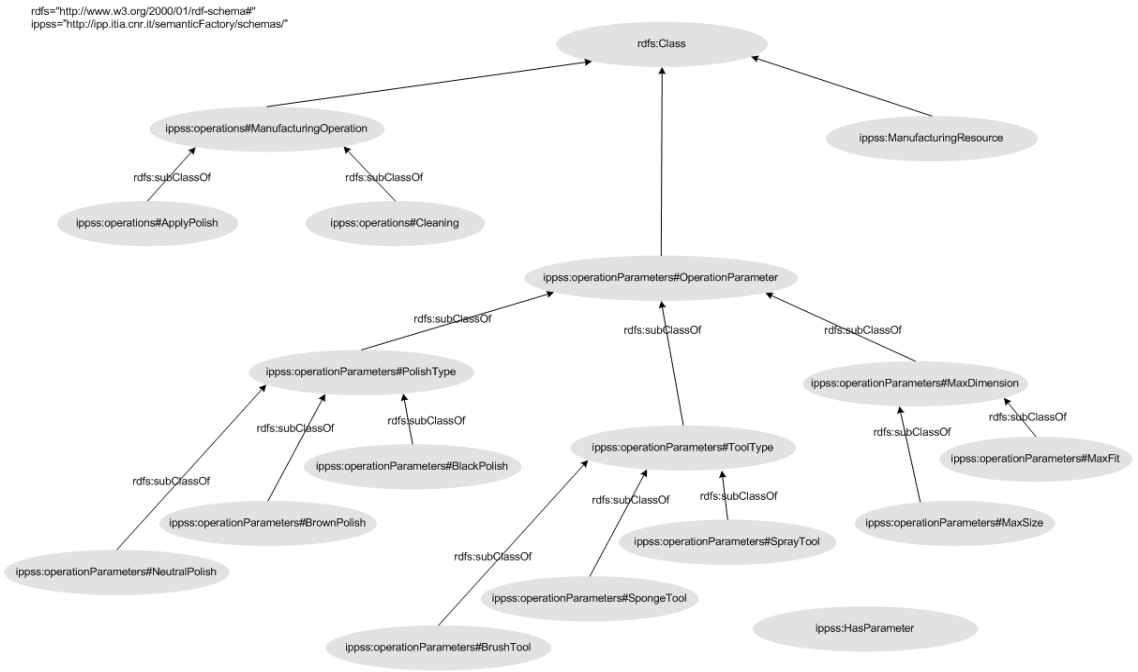

Figure 12. RDF based description of the finishing Tern.

As an example, Figure 12 shows the RDF based representation of the manufacturing operations provided by the machines integrated within a Tern of the line. In particular, the finishing Tern, aimed at cleaning the shoes before being putted into the boxes is described. Figure 13 shows the XML file implementing the finishing Tern RDF description. Such format is processed by the reasoner integrated within the supervisor.

```
<?xml version="1.0" ?>
<!DOCTYPE rdf:RDF (View Source for full doctype...)>
- <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:ops="http://ipp.itia.cnr.it/semanticFactory/phases/finishing/operations#
    xml:base="http://ipp.itia.cnr.it/semanticFactory/phases/finishing">
  - <rdf:Description rdf:about="#FinishingCell1">
      <rdf:type rdf:resource="#ManufacturingCell" />
      <ops:HasResource rdf:resource="#SprayMachine1" />
      <ops:HasResource rdf:resource="#BrushMachine1" />
      <ops:HasResource rdf:resource="#SpongeMachine1" />
    </rdf:Description>
  - <rdf:Description rdf:about="#SprayMachine1">
      <rdf:type rdf:resource="#ManufacturingResource" />
      <ops:HasOperation rdf:resource="#SMApplyPolish1" />
    </rdf:Description>
  - <rdf:Description rdf:about="#SMApplyPolish1">
      <rdf:type rdf:resource="#ApplyPolish" />
      <ops:HasParameter rdf:resource="#SMTool1" />
      <ops:HasParameter rdf:resource="#SMPolish1" />
    </rdf:Description>
  - <rdf:Description rdf:about="#SMTool1">
      <rdf:type rdf:resource="#Spray" />
    </rdf:Description>
  - <rdf:Description rdf:about="#SMPolish1">
      <rdf:type rdf:resource="#NeutralPolish" />
```

Figure 13. RDF XML format

Once a new operational unit is plugged into the production system its agent asks for the registration service to the manager passing is description as argument, formally reporting the operations to be provided and the required resources (i.e. tools) to be used during operation execution. The operations to be provided and the resources to be used are described with reference to a common ontology [19].

The manager interprets the ontological description of the intelligent unit and integrates it into a dedicated knowledge base within the manager architecture. To such an aim, software utilities provided by the Jena Semantic Web framework [20] have been adopted. Jena is an open source Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine.

A searching algorithm, implemented in Java, is then executed in order to find the run-time available manufacturing resources capable of providing the requested operations.

6. Control Solution Verification

The validation of the control software is the final task of the whole automation system design life cycle before the system deliver to customers. Within such phase, the correspondence of the automation system behaviour with the requirements described in the system specifications is verified. Normally, the control software development is not carried out directly on the hardware devices, but is performed "offline", into dedicated development environments. Once completed, deployed control software is downloaded on the hardware devices and tested onsite, by controlling the real process. Major drawbacks are the increase in commissioning times and costs, huge underperforming conditions and critical damages on the plant devices, caused by undetected errors in the control system. Furthermore, software modifications are more difficult to be implemented in such conditions, due to complexity of overall system.

For bigger plants, the first validation of the software is often carried out by means of software/hardware test panels, offering to the control system the same electrical interface, manually operated to change field values so emulating the process variables variation. This method is carried out without any structured criteria and the dynamic control system response cannot be tested deeply, so leading to the identification only of some rough implementation errors.

A different way to validate the control software is possible by using process simulators, which are not widely adopted in industrial practice because their usage requires relevant process competencies - not always available - and additional costs for programming activities.

In the present work, a closed loop simulation approach has been adopted for the validation of overall developed control application also considering the bottom-up methodology presented in [21]. In particular, a simulation model of the process to be controlled, running on a dedicated PC, has been developed and connected to the control system by mapping I/O boards.

Figure 14 illustrates the architecture of the verification framework. In this application the Simulink, State Flow and Virtual Reality toolboxes of the Matlab platform have been respectively utilized for the development of the logical and graphical model of the molecular transport system.

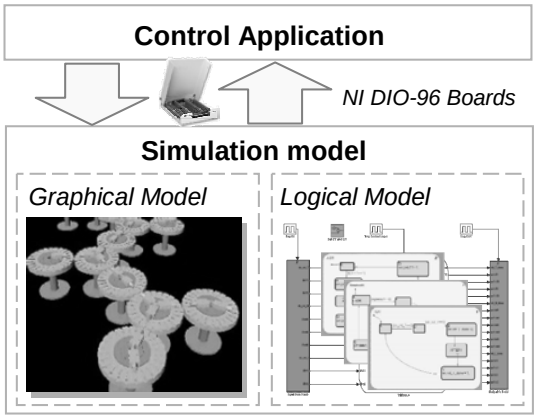

Figure 14. Simulation framework.

Furthermore, the software solutions running on the real control hardware, before being installed into the real manufacturing plant, have been validated within a pilot plant which reproduces in scale 1:10 the molecular transport line, as shown in Figure 15. This technologic demonstrator is part of the ITIA Automatic Control Laboratory and it has been realized with the same number of I/O signals exchanged between the real manufacturing plant and its automation system, for study and testing purposes.

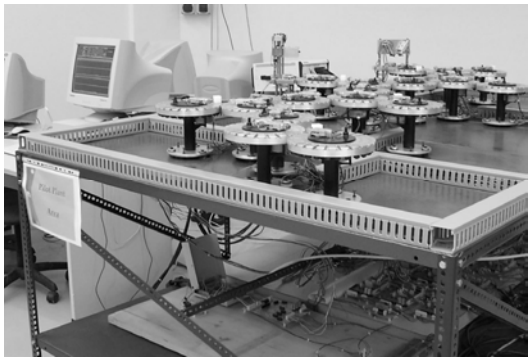

Figure 15. Technological demonstrator

Thus, an automation system tested with the demonstrator can be directly connected to the real manufacturing plant, reducing ramp-up efforts and costs.

7. Conclusions

The present paper describes a self-adaptive control architecture deployed for a real industrial plant. To such an aim, a modular and distributed approach has been adopted, integrating an IEC 61499 based control solution for real-time control purposes and a semantically enriched multi-agent control layer for dynamic supervision strategy. Main focus of the paper is kept on the description of the overall solution structure, highlighting the capabilities of run-time updating of the knowledge model used for control decisions, while factory products and resources changes occur.

Furthermore, the designed control strategy, implemented within a structured and reconfigurable IEC 61499 based application has been detailed.

Moreover, a closed loop simulation based approach has been adopted to validate the implemented control solution before the integration into the real manufacturing plant, thus reducing commissioning time and costs.

Next efforts will mainly regard the integration of dedicated supervision and on-line dispatching/scheduling facilities onto the deployed control architecture.

Acknowledgements

The work presented in this paper has been partially supported by Regione Lombardia within the project “Accordo di Programma Quadro CNR - Regione Lombardia: Progetto 3 - Processi high tech e prodotti orientati al consumatore per la competitività del manifatturiero lombardo”.

REFERENCES

1. KOREN, Y., U. HEISEL, F. JOVANE, T. MORIWAKI, G. PRITSCHOW, G. ULSOY, H. V. BRUSEEL, **Reconfigurable Manufacturing Systems**, in Ann. CIRP 1999, vol. 48(2), pp. 527-540.
2. ALMEIDA, E., J. LUNTZ, D. TILBURY, **Event - Condition - Action Systems for Reconfigurable Logic Control,** IEEE Transaction on Automation Science and Engineering, Vol. 4(2), April 2007, pp. 167-181.
3. BRUSAFERRI, A., A. BALLARINO, E. CARPANZANO, **Enabling Agile Manufacturing through Reconfigurable Control Solutions**, Proc. at 14th IEEE International Conference on Enabling Technologies and Factory Automation (ETFA09), September 2009, Mallorca, Spain.
4. CARPANZANO, E., F. JOVANE, **Advanced Automation Solutions for Future Adaptive Factories**, Annals of the CIRP, 2007, pp. 435-438.
5. PECHOUCEK, M., V. MARIK, **Industrial Deployment of Multi-agent Technologies: Review and Selected Case Studies**, Autonomous Agents and Multi-Agent Systems Journal, Publisher: Springer Netherlands, Published online: 14 May 2008, pp. 397-431.
6. SIRENA Project www.sirena-itea.org.
7. RIMACS Project - Radically Innovative Mechatronics and Advanced Control Systems, www.rimacs.org.
8. SOCRADES Project www.socrades.eu.
9. LASTRA, J., I. DELAMER, **Web Services in Factory Automation: Fundamental Insights and Research Roadmap**, IEEE Transactions on Industrial Informatics, Vol. 2(1), February 2006, pp. 1-11.
10. O3NEIDA Network of Networks to Advance Distributed Industrial Automation, www.oooneida.org
11. LEPUSCHITZ, W., M. VALLEE, M. MERDAN, P. VRBA, J. RESCH, **Integration of a Heterogeneous Low Level Control in a Multi-Agent System for the Manufacturing Domain**, Proc. at 14th IEEE International Conference on Enabling Technologies and Factory Automation (ETFA09), September 2009, Mallorca, Spain.
12. HERRERA, V., A. BEPPERLING, A. LOBOV, H. SMIT, A. W. COLOMBO, J. L. LASTRA, **Integration of Multi-Agent Systems and Service-Oriented Architecture for Industrial Automation**, Proc. at IEEE International Conference on

Industrial Informatics (INDIN2008), Daejeon, Korea, July 13-16.

13. LEITAO, P., F. RESTIVO, **Implementation of a Holonic Control System in a Flexible Manufacturing System**, IEEE Transactions on Systems on, Man, and Cybernetics - Part C: Applications and Reviews, Vol. 38, No. 5, September 2008, pp. 699-709.
14. International Electro - technical Commission, (IEC), **International Standard IEC61499**, Function Blocks, part 1-4, IEC Jan. 2005 Edition 1.0., http://www.iec.ch/
15. MEDJOUDJ, M., **ESA_PetriNet: a Tool for Extracting Scenarios in Computer Controlled Systems**, Studies in Informatics and Control, vol. 17, No. 1/2008, Published by the National Institute for Research and Development in Informatics, pp. 71-84.
16. ISaGRAF Workbench-www.isagraf.com.
17. Java Native Interface -http://java.sun.com/j2se/1.4.2/docs/guide/jni/.
18. OWL Web Ontology Language - http://www.w3.org/TR/owl-guide/.
19. OPREA, M., **Ontology Mapping in Open Multi-Agent Systems**, Studies in Informatics and Control, Vol. 16, No. 2/2007, Published by the National Institute for Research and Development in Informatics.
20. Jena Semantic Web framework - http://jena.sourceforge.net/
21. CARPANZANO, E., A. BALLARINO, **A Structured Approach to the Design and Simulation-based Testing of Factory Automation Systems**, IEEE ISIE'2002 International Symposium on Industrial, Electronics, L'Aquila, July 8-11, 2002.