Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 28 (2015) 119 – 124

www.elsevier.com/locate/procedia

3rd CIRP Global Web Conference

Autonomous production systems using open architectures and mobile robotic structures

G. Michalos[a,*], S. Makris[a], P. Aivaliotis[a], S. Matthaiakis[a], A. Sardelis[a], G. Chryssolouris[a]

[a]*Laboratory for Manufacturing Systems and Automation, Department of Mechanical Engineering and Aeronautics, University of Patras, Patras, Greece*

* Corresponding author. Tel.: +30-261-099-7262; fax: +30-261-099-70744. *E-mail address:* michalos@lms.mech.upatras.gr

Abstract

This paper investigates the flexibility aspects of production systems that use highly interactive and autonomous mobile robotic units. Open communication architectures and ontology technologies enable the accurate representation of robot capabilities. Mobile robots can relocate themselves and support the production process, thus providing a higher reconfiguration potential. Services are used for real time transactions between stationary and mobile robots towards the implementation of a process plan. The units can cooperate and determine their course of actions. This approach was applied to an experimental cell, where the system managed to implement production plans for the packaging of small sized products without human intervention.

Keywords: System architecture; Reconfiguration; Robot; Planning;Assembly

1. Introduction

Typical manufacturing systems comprise rigid flow line structures by employing model-dedicated handling and transportation equipment of raw materials and components [1]. They have fixed control logic and the signals-based tasks sequencing requires significant effort for the implementation of changes in the production plan. In Figure 1, the hierarchical representation of an assembly line with multiple stations and resources (R1, R2 etc.) is shown along with the tasks' breakdown into operations for each resource. The current practices involve the use of Programmable Logical Controller (PLC) signals to denote the start/stop of the operations, requiring a hard-coded approach that signifies high complexity and downtime in case of changes.

These systems cannot follow the market needs, for fast introduction of new products or frequent improvement of the existing ones. New production systems need to exhibit attributes such as flexibility, reusability, scalability and reconfigurability [1-3].

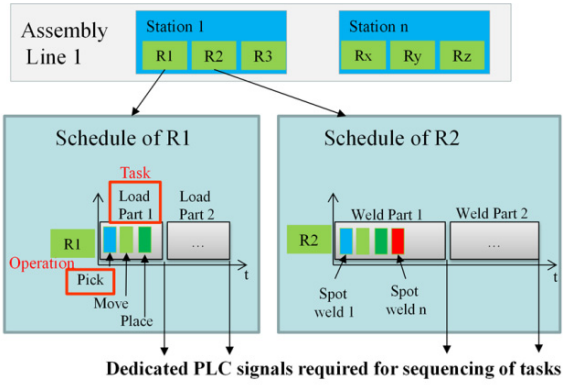

Fig. 1. Hierarchical model of production line and operations

In order for these goals to be achieved, alterations of the production and logistics processes are required to enable the system's fast reconfiguration with minimal human intervention [4]. The current problems addressed by this approach may be summarized into the following:

doi:10.1016/j.procir.2015.04.020

- Reduction of hard wired control logic that allows limited or no reconfiguration capabilities and requires great effort in terms of human intervention. Activities such as those of scheduling, planning and programming of resources are now partially or individually automated. As a result, a significant reduction in the overall system reconfiguration time is expected.
- Reinforcement of random production flows through the use of mobile robots, eliminating the existing fixture based - static production paradigms that do not allow for changes in the production system structure.
- Autonomous behavior – planning of activities at multiple levels. Currently, autonomy is constrained by rules that are imposed by the strictly specified task execution routines for each resource. Robots however, can execute the same task, in a multitude of ways, but are now limited by the human dictated programming and planning. A significant reduction in programming efforts will be achieved.

This study considers the case of automated production systems, where mobile robotic units are used for the provision of the desired reconfiguration capabilities. In this paradigm, the mobile robots are capable of navigating into assembly stations and undertaking/supporting new assembly tasks automatically. The evolution of the production systems is conceptualized in Figure 2.

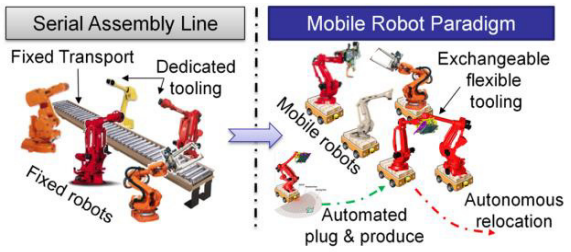

Fig. 2. Evolution of the production system

Different attempts have been made so far to introduce mobile manipulators to industrial environments and exploit their flexibility potential [5][5]. The latest examples involve the introduction of a mobile manipulator for assembly applications [6], the creation of an autonomous multi-purpose industrial robot [7] as well as the development of a high payload mobile manipulator for automotive Body in White (BiW) applications [8][8]. One of the most important problems of deploying mobile robots is that the environment around them is not static. Therefore, the mobile units should be capable of changing their path in case of any alterations in their surroundings [9-11].

The exploitation of the flexibility potential in systems that utilize mobile robots, signifies the definition and solution of a complex planning and scheduling problem. The first part of the problem deals with the production planning level (identification of tasks) and secondly, the scheduling part that deals with the assignment of these tasks to the resources [12]. Agent based systems have been the main research direction that has been followed towards addressing these problems [13,14].

In most of the aforementioned cases, the mobile robots act as individual units that execute the tasks foreseen in the production schedule. The main difference of this approach with past attempts lies in the flexible nature of robots:

- Robots are capable of undertaking a variety of tasks (processing and handling) and therefore, infinite alternatives can be realized when multiple aspects in the decision making: robot type selection, sequencing, motion planning etc, are being considered. This for example, contradicts the application of agents in Computer Numerical Controller (CNC) machines that are usually part of Flexible Manufacturing Systems (FMS). In this case, machines have several programs stored and the agents decide which one to execute on the basis of the pending operations.
- The dynamic nature of the tasks (pick and place from unknown positions, navigation in the shop floor etc.) discussed in this paradigm, requires a much more complex coordination between the resources (horizontal integration) themselves as well as the higher level coordination mechanisms/services (vertical integration) which has not been investigated into for such types of resources.
- Agent based approaches, although are flexible in pursuing a smooth operation, they are not generic enough to support a dynamic operation by multiple, however dissimilar resources. The wealth of the robotic equipment available and the respective capabilities offered, calls for technologies such as: Standardized interfaces for integration and configuration of different hardware and software components, through hardware and software abstraction capabilities and decoupling of parameters, request/storage/ acquisition with the use of open frameworks such as ROS [15].

The following sections present an approach, where the tasks are automatically allocated to the stationary and mobile robots, enabling a more dynamic approach to the system's reconfiguration. In this context, the underlying models and required technologies are presented in Section 2. Section 3, provides the details of such a system's implementation, while section 4, discusses its application in a case study. Finally, section 5, draws the conclusions of the approach and provides the outlook and challenges that future research should focus on.

2. Approach

The approach creates an architecture that would allow stationary and mobile robotic units to communicate with each other and negotiate the production plan that they need to implement. The main characteristics that justify the development of the architecture involve:

- **Openness of the architecture**: robustness and assures autonomous behavior in case of a failure.
- **Flexibility:** not unique to a particular robot or task.
- **Dynamic operation:** Unlike other resources, robots and especially mobile units require continuous data/ status update during their execution.

In this context, a robot can be used in more than one production processes and in case of failure, another robot can undertake its task. The mobility enables the creation of random production flows since the production processes can be transferred to the shop floor. The building blocks and their implementation in such a system are presented in the following sections.

2.1. Data model

A data model should include all the information needed to enable the resources and services to perform the decision making on their own i.e. autonomously [16]. This information should depict the complete shop floor status, both in terms of physical elements and operations. The Unified Modeling Language (UML) schematic of the data model is shown in Figure 3, where all classes are represented. Associations or composition between classes is also shown. For example, the association "hasDockingStations" denotes whether a station is equipped with an area where the mobile robot can dock and connect to external power [17]. The representation of these characteristics allows them to be automatically included in the decision making process.

2.2. Integration and communication architecture

The architecture is considered open when its specifications are public. This includes officially approved standards as well as privately designed architectures, whose specifications are made public. Such architectures enable the plugging of new modules and the entire system can be developed through the evolvement of the separate modules. The use of an open architecture has the advantages of: a) Reduced cost, b) Faster development, c) Greater innovation potential and d) Easier integration with existing systems.

Several attempts have been made recently for the development of standard robot software platforms with software development kits (SDKs) by robotics suppliers in order to simplify integration and robot development. The Robot Operating System (ROS) is an open framework for robot software development that aims to simplify the task of creating a complex and robust robot behavior across a wide variety of robotic platforms. Based on ROS, an architecture has been developed in this study for the integration and communication of mobile robotic units. The concept is shown in Figure 4.

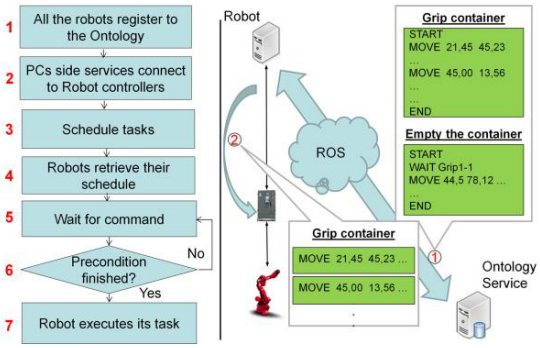

Fig. 4. Architecture concept

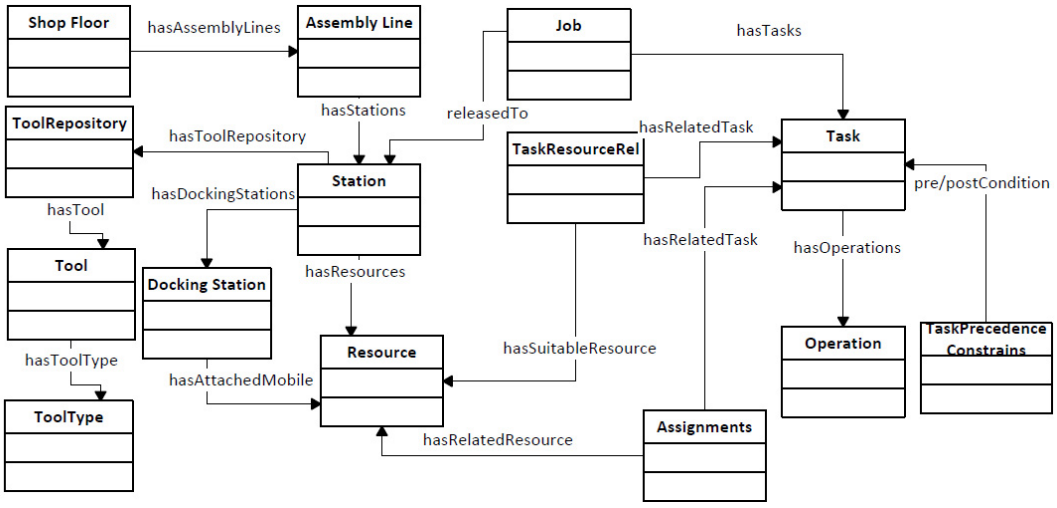

Fig. 3. Data model for the reconfiguration logic

The mobile unit itself has an interface and is able to communicate with the other resources by using the server/ client or the publisher/ subscriber model. The first allows the synchronous communication between the resources. In other words, a resource will have the role of the client, who can request something from the resource, playing the role of the server, and then waiting for the response. The second model (publisher/subscriber) is used for asynchronous topic-based communication. All the resources subscribe to a topic and everyone can publish messages. The messages are read by the subscribers and afterwards are parsed and translated into something useful by the resources interested in them. For example, when a robot breaks down, a message is published to the public topic and all the others can read it and do whatever is necessary in order for the broken down resource to be substituted by another mobile robot.

Building on top of the ROS, the Ontology Service integration and communication software was developed. The Ontology Service is used for data management and storing services to the resources and services. A semantic repository, referred to as the "Ontology Repository" software module, is also utilized for these functionalities. In Figure 5, the basic structure of the software architecture is shown.

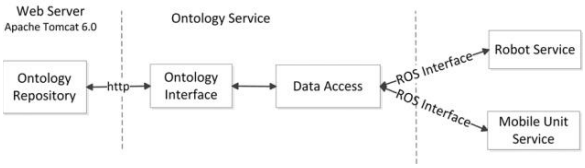

Fig. 5. Software architecture structure

The 'Ontology Repository' is implemented as a web application and in this case, is hosted on an apache Tomcat server. To enable the proper semantic functionalities, the Jena framework is utilized for the implementation of the 'Ontology Repository'. The Ontology Service comprises two software modules, the 'Data Access' and the 'Ontology Interface'. The 'Data Access' module provides the communication among the Ontology Service and the other resources or services. The 'Ontology Interface' module enables the communication of the Ontology Service with the 'Ontology Repository'. In this way, when a ROS message, requesting information from the 'Ontology Repository', arrives at the 'Data Access' module, the latter triggers the 'Ontology Interface' module and retrieves the information from the 'Ontology Repository'. The communication between the "Ontology Interface' and the 'Ontology Repository' application is implemented using HTTP request, while communication between the 'Data Access' and the 'Ontology Interface' is implemented via function pointers.

Every resource in the architecture has a 'Data Access' module, whilst a ROS Interface exposed in this module runs as a separate thread. The 'ROS Interface' depicts the implementation of the message exchanging mechanisms utilizing the ROS framework, whereas the 'Data Access' is responsible for parsing the incoming messages to meaningful for the resource information or creating outgoing messages. The 'ROS Interface' implements the service calls and the service descriptions to be advertised to the rest of the resources or services by each module. Its development again utilizes the ROS framework and follows the service oriented paradigm principles. Any information received by the 'ROS Interface' (data requests or instructions) will be forwarded to the 'Data Access' module.

The 'Data Access' module is responsible for analyzing the messages received by the 'ROS Interface' and for triggering the relative software modules within the resource. If for example, a message for a path information request is received by the 'ROS Interface' of the Mobile Unit, the 'ROS Interface' will pass the message data to the 'Data Access' module and the latter will trigger the appropriate software module within the Mobile Unit internal control systems. After the pieces of path information have been calculated, the 'Data Access' module sends them to the relevant resource.

2.3. Mobile robot control services

The Mobile Unit Software modules are developed in such a way so as to perform the entire control and navigation of the Mobile Unit. All modules included are shown in Figure 6. In order to be integrated into the platform, the Mobile Interface was developed following the client-server ROS model.

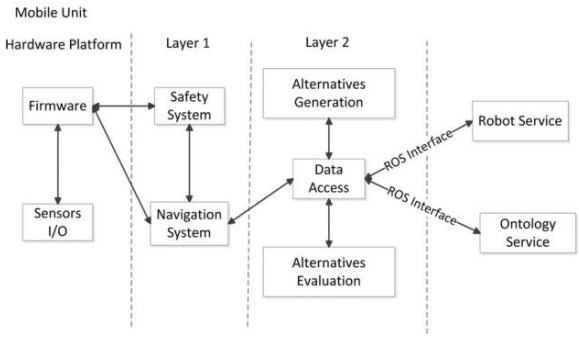

Fig. 6. Mobile unit local architecture

The most important functionalities to be handled through the communication architecture involve:

- **Retrieve Mobile Unit Position:** The Mobile Unit interface provides information about its current position so that it can be used for task allocation

- **Navigate to position:** the Mobile Unit interface is used to passing a command to the Mobile Unit Navigation system in order for it to move to a new position
- **Retrieve Information about a Path:** the Mobile Unit should provide information about the selected paths between two positions. Path distance and estimated arrival are used by the planning algorithm.
- **Leave / Arrive at a docking station: t**he Mobile Unit is able to push/update information about its docking or undocking to the Ontology service.

3. Case study

A case study was setup with the use of a mobile robot in order for the aforementioned functionalities in a manufacturing system to be demonstrated. The scenario involves two Comau® Smart5 Six fixed robots and one Robotino® mobile robot, as shown in Figure 7. The stationary robots are used to sorting small parts, in this case, plastic shaver handles, while the mobile robot transfers new parts into a small container that the robot on the right (R2) picks and places on the conveyor in front of it.

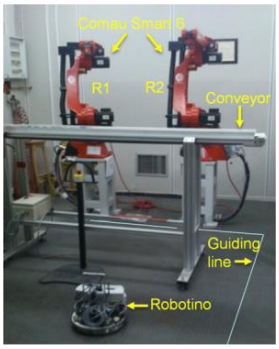

Fig. 7. Case study physical layout

The testing workload included only one job that comprised several tasks to be undertaken by the mobile and the stationary robots respectively. The hierarchical breakdown is shown in Figure 8.

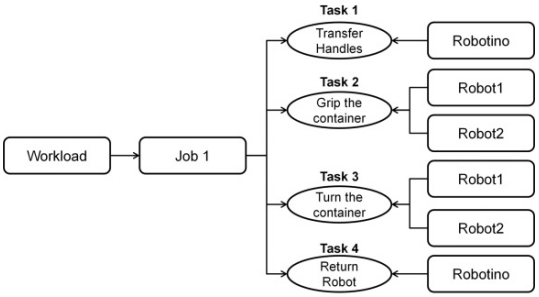

Fig. 8. Workload breakdown

Tasks 2 and 3 can be assigned only to R1 or to R2 or be performed by the cooperation of the two fixed robots. The scenario foresees a breakdown of robot R2, which is communicated to the ontology. As a result, a negotiation between the services of all the resources takes place and the task of unloading the container from the robot is assigned to the other robot (R1). The mobile robot is also automatically instructed to move in front of R1 so as for the unloading to take place.

As described in section 2, ROS was used in order to implement the communication among the resources. At the system's startup, all the PCs services that were directly connected to the robot controllers, registered to the Ontology. The breakdown signal was transmitted from the service that was connected to the stationary robot (right side of Figure 7). The execution of the scheduling algorithms was carried out by one of the robots (after negotiation between them) and the tasks of the broken down robot were re-assigned to the other one with the same suitability. The outcome of the scheduling process is graphically shown in Figure 9.

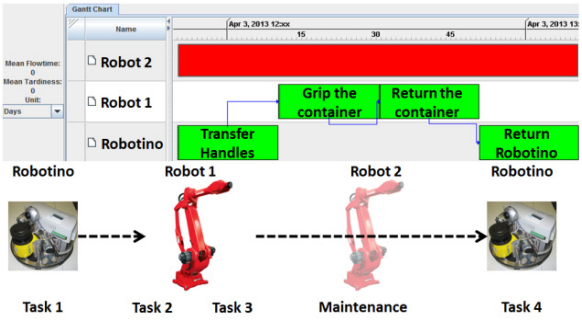

Fig. 9. Scheduling algorithm result for the case study

All robots, mobile and stationary, have successfully retrieved their schedule, whilst the commands entered a waiting status for the execution of their operations.

The RobotinoView® software suite was used for the navigation and interfacing of the mobile robot through the WiFi network. This software enables the intuitive creation of programs through the use of function blocks that allow logic, mathematics and arithmetic operations. A camera and a line detector function block were used for the navigation program. (guiding line in Figure 7). Its final position is dependent on the fixed robot that will execute the gripping task.

At the beginning of the execution process, a signal was sent from ROS to Robotino to define the new (updated) final position and pose of the mobile robot. Firstly, from the starting position following its path, Robotino transferred the container, which included the plastic handles. Upon resuming this position, Robotino via the RobotinoView, signaled to ROS the task's completion. The next task being the gripping of the

container by the robot was then initiated. After the robot had performed the third task and emptied the container above the conveyor, it also signaled the end of the task. Finally, ROS sent a signal to Robotino and the last task (return of mobile robot to starting position) was executed by the mobile unit. The small scale testing proved that all technologies could be integrated and work efficiently. The verification in large scale assembly environments (such as the automotive) and diverse applications (such as consumer goods industry) is an ongoing study in the AUTORECON project.

4. Conclusions

This paper discussed the integration and communication architecture for the efficient utilization of an autonomous mobile robot, in cooperation with other fixed robots inside the production system. The architecture is open and enables the integration of multiple resources through the use of ontology and service technologies. The hardware and software architectures that the robotic resources need to comply with for the exploitation of this method, have also been presented. The implemented systems are able to:

- generate task assignments for the robots (mobile and stationary) within the production environment
- sequence these tasks automatically without the use of hard coded PLC approaches
- coordinate the operation of the resources for the automatic execution of these tasks.

With this method's application to a case study, a feasible reconfiguration plan, allowing the replacement of a malfunctioning robot by an adjacent one, without any human intervention, was generated. The benefits of adopting such technologies over the traditional control techniques mainly lie in the shortened reconfiguration time as well as in the reduction of the efforts, required for the commissioning of new resources.

Future research should focus on the standardization of hardware (electrical and mechanical) and software interfaces in order for a seamless 'Plug & Produce' behavior of the resources to be achieved. Moreover, the current control architectures on monolithic PLC – control, should evolve towards a more open, service oriented architecture that would enable easier and less complex networking among the different devices. Finally, the method's expansion to account for processes that humans and robots can cooperate in the same workspace, is also an open challenge [19].

Acknowledgements

This work has been funded under the EU FP7 project "Autonomous co-operative machines for highly reconfigurable assembly operations of the future - AUTORECON" (www.autorecon.eu). The authors thank all project partners for their valuable feedback.

References

[1]. Michalos, G., Makris, S., Papakostas, N., Mourtzis, D., Chryssolouris, G., 2010, Automotive assembly technologies review: challenges and outlook for a flexible and adaptive approach. CIRP Journal of Manufacturing Science and Technology, 2, p. 81-91.

[2]. Chryssolouris, G., 2006, Manufacturing Systems: Theory and Practice, 2nd ed. Springer-Verlag, New York.

[3]. Paralikas, J., Fysikopoulos, A., Pandremenos, J., Chryssolouris, G., 2011, Product modularity and assembly systems: An automotive case study, CIRP Annals - Manufacturing Technology, 60, p. 165–168.

[4]. Reinhart, G., Tekouo, W., 2009, Automatic programming of robot-mounted 3D optical scanning devices to easily measure parts in high-variant assembly, CIRP Annals - Manufacturing Technology, 58, p. 25-28.

[5]. Angerer, S., Strassmair, C., Staehr, M., Roettenbacher, M., Robertson, N.M., 2012, "Give me a hand — The potential of mobile assistive robots in automotive logistics and assembly applications", Technologies for Practical Robot Applications (TePRA), 2012 IEEE International Conference on, p.111-116

[6]. Hamner, B., Koterba, S., Shi, J., Simmons, R., Singh, S., 2010, An autonomous mobile manipulator for assembly tasks, Autonomous Robots, 28, p. 131-149.

[7]. Hvilshoj, M., Bogh, S., 2011, "Little Helper" - An Autonomous Industrial Mobile Manipulator Concept. International Journal of Advanced Robotic Systems, 8, p. 80-90.

[8]. URL AUTORECON, 2013, [Online] Available: http://www.autorecon.eu

[9]. Yu-Qing CHEN, Yan ZHUANG, Wei WANG, 2007 ,"A Dynamic Regulation and Scheduling Scheme for Formation Control", Acta Automatica Sinica, 33, p. 628-634

[10]. Ch.K. Volos, I.M. Kyprianidis, I.N. Stouboulos, 2013, "Experimental investigation on coverage performance of a chaotic autonomous mobile robot", Robotics and Autonomous Systems, 61, p. 1314-1322

[11]. J.L. Posadas, P. P!erez, J.E. Sim, G. Benet, F. Blanes, 2002, "Communications structure for sensory data in mobile robots", Engineering Applications of Artificial Intelligence,15, p. 341–350

[12]. Stefano Giordani, Marin Lujak, Francesco Martinelli, 2013, "A distributed multi-agent production planning and scheduling framework for mobile robots", Computers & Industrial Engineering, 64, p. 19-30

[13]. J.L. Posadas_, J.L. Poza, J.E. Simo´ , G. Benet, F. Blanes, 2008,"Agent-based distributed architecture for mobile robot control", Engineering Applications of Artificial Intelligence, 21, p. 805-823

[14]. Weiming Shena, Qi Hao, Shuying Wanga, Yinsheng Lia, Hamada Ghenniwac, 2007 ,"An agent-based service-oriented integration architecture for collaborative intelligent manufacturing", Robotics and Computer-Integrated Manufacturing, 23, p.315-325

[15]. Morgan Quigley, Brian Gerkeyy, Ken Conleyy, Josh Fausty, Tully Footey, Jeremy Leibsz, Eric Bergery, Rob Wheelery, Andrew Ng, 2012, "ROS: an open-source Robot Operating System", ICRA workshop on open source software

[16]. Yang, Q. Z., Miao, C.Y., Zhang, Y., Gay, R., 2006, "Ontology Modelling and Engineering for Product Development Process Description and Integration," Industrial Informatics, 2006 IEEE International Conference on, p. 85-90

[17]. Secchi, C.; Bonfe, M.; Fantuzzi, C., 2007, "On the Use of UML for Modeling Mechatronic Systems", Automation Science and Engineering, IEEE Transactions, 4, p.105-113

[18]. URL ROS, 2013, [Online] Available: http://www.ros.org/

[19]. Morioka, M., Sakakibara, S., 2010, A new cell production assembly system with human–robot cooperation, CIRP Annals - Manufacturing Technology, 59, p. 9–12.