Decentralized Robot-Cloud Architecture for an Autonomous Transportation System in a Smart Factory

Jon Martín, Stefan May
Faculty of Electrical Engineering, Precision Engineering and Information Technology in the Technische Hochschule Nuernberg Georg Simon Ohm, Germany
Nuremberg 90489
jon.martingarechana@th-nuernberg.de

Sebastian Endres
Faculty of Mechanical Engineering and Building Services Engineering in the Technische Hochschule Nuernberg Georg Simon Ohm, Germany
Nuremberg 90489

Itziar Cabanes
Department of Automatic Control and System Engineering in the University of the Basque Country (UPVEHU), Bilbao, Spain
Bilbao 48013

ABSTRACT

The aim of this paper is to present a decentralized control architecture for an autonomous transportation system in the manufacturing facility of the future. Each component in the factory (machine, robot, operator...) is represented as an individual agent on the cloud and makes autonomous decisions based on the information exchanged with other agents. Production machines control their own material replenishment by contracting the services of Autonomous Transportation Vehicles (ATV) over the cloud. Moreover, two control panels for human synergy on strategical and operational layers for monitoring and interacting with the system have been provided. Based on new trends on the industry 4.0 revolution, this paper develops an intelligent communication architecture between machines, robots and humans. This novel architecture makes the system robust, flexible and scalable. The requirements for such an architecture have been first defined and then validated via a series of theoretical cases and two experiments where communication and hardware failures have been triggered.

CCS CONCEPTS

• **Computer systems organization** → **Distributed architectures**; **Robotics**; **Reliability**; • **Human-centered computing** → *Human computer interaction (HCI)*;

KEYWORDS

Decentralized Architecture, Autonomous transportation System, Industry 4.0, MRTA

1 INTRODUCTION

Modern factories are constantly changing and optimizing to adapt to the rapid fluctuations on the markets: Shorter product life cycles, mass customization and the rising speed of delivery. In the classical hierarchical and highly structured industrial environments changes are difficult to implement and involve time and financial costs. Therefore, new manufacturing facilities tend to be highly decentralized systems with self-organized modules that grant flexibility and improve the adaptability to changes.

New trends such as Industry 4.0, the Internet of Things (IoT) and Cloud Robotics lead the path to the future smart factories. Every component in the factory –such as production machines, robotic components (Autonomous Transportation Vehicles (ATVs), Industrial Robots, etc.) and operators (via PDAs or smart tablets) – will be represented as individual software agents in the cloud. All of them are interconnected and have the ability to make their own decisions. In this way, the current hierarchical control structures will be replaced by decentralized mesh-like control architectures. Moreover, the cloud concept offers additional services such as outsourcing computational power and data storage. The main benefits of the smart factories are:

Robustness: The decentralization of the control system avoids the single point of failure problem.

Modularity: The independence between modules increases the *flexibility* by making it easier to update or replace them (when obsolete or when a failure occurs) without affecting other modules. The factory does not have to stop for hardware or software updates.

Scalability: It is easier to adapt the factory by for example increasing the robot units during peak-workloads or as the market grows.

High information flow [1]: In most cases the material flow is still controlled by cyclical analog systems such as kanban cards. Thanks to the digitalization of the system, electronic kanban cards offer real time information. This huge data provided by every single component in the factory can then be used for forecasting, learning and optimizing the system.

Connects People to the factory: Through smart devices, for instance mobile phones, tablets or PCs, information in real time is accessible over the cloud for maintenance and diagnosis operation but also for the clients to inform about the current status of their product.

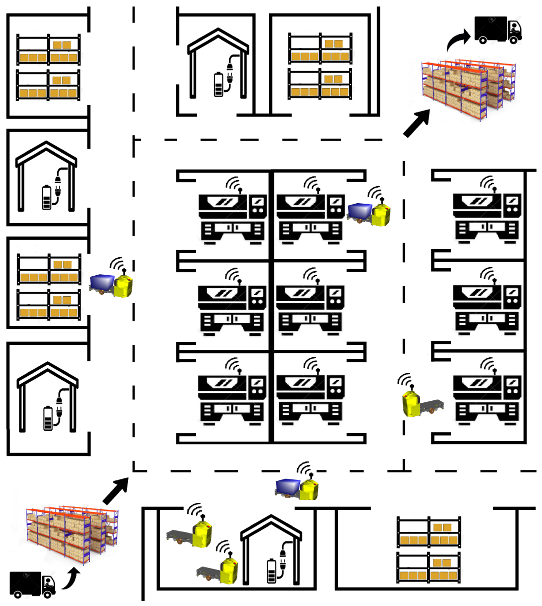


Figure 1: The factory layout contains the following components: source and delivery warehouses (bottom left/top right corners); interim storages and charging stations (on the sides); working stations with production machines (center and right side); ATVs spread all over the factory.

This paper presents a decentralized control architecture for a robust transportation task on a Smart Factory. The control is divided into a strategical layer, where global production goals are defined, and an operational layer, where individual intelligent modules (production machines, ATVs, Operators...) make decisions based on the information exchanged with each other. The ATVs offer a transportation service over the cloud that is used by the production machines to replenish their material reserves. At the same time the operators can interact over the cloud with both, machines and robots for a more secure and reliable functionality.

Figure 1 presents an intermediate layout between the current and the future factory layout. The material is first transported from the warehouse to local storage areas and from there to the machines by ATVs. In the future, hundreds of robots could transport the material directly from warehouses to machines avoiding the intermediate step, as well as freeing valuable space in the factory.

This paper is structured as follows: Section 2 presents related works on smart factories and autonomous transportation systems. In section 3, the decentralized cloud architecture is presented. Section 4 introduces the use cases and the experiments done for the requirements validation of such an architecture. Finally, the conclusions are summarized and the future work is discussed.

2 RELATED WORK

New research on smart factories have been focused on the effort to connect humans, machines and infrastructure over the network for sharing data and processing. Works on modularized systems, the IoT and cloud computing that pushes this new industrial 4.0 revolution are presented here. Furthermore, prior work on Multi Robot Task Allocation (MRTA) problem is introduced.

The cooperation of single modules within one logistic system has been summarized in [1] as Cyber Physical Logistic Systems (CPLS): a concept defined between embedded systems and the vision of the IoT and the Internet of Services [2], [3]. During the last years, the modularization on autonomous transport systems has been proven in several research and industrial projects such as flexible conveyor systems [4], [5] or Autonomous Transport Vehicles [6], [7], [2]. Also the RoboCup Industrial @Work league competition [8] pushes the idea of a mobile platform integrated with and industrial arm that autonomously schedules a series of picking up - delivery tasks received from a referee system.

The cloud architecture offers several advantages on robotic and automation systems and can be divided into two complementary tiers as in [9]: In the first machine-to-cloud (M2C) communication level, computing and storage resources can be moved to servers in the cloud, reducing costs in the robot but providing them with an almost infinite power. Some examples are DavinCi [10] or Amazon Web Services [11]. Moreover, the stored information can be shared with other robots for a common learning: Projects such as RoboEarth [12] have demonstrated that the robots can learn new abilities and share them with others over the cloud. However, these systems are based on central servers where the information and the computing power are available. A failure in the agent-cloud connection or directly in the server prevents the agent from making decisions and leads to inoperability. To ensure the necessary robustness in industrial applications, agents should be self-sufficient to perform primary tasks on their own and only outsource secondary, less urgent, tasks.

Within the second tier, the machine-to-machine (M2M) communication level, a group of robots communicate via wireless links to form a collaborative computing factory. The benefits of a cooperative factory are: First, the computing capability from individual robots can be pooled together to form a virtual ad hoc cloud infrastructure. Second, among the collaborative computing units, information can be exchanged for synergetic decision making. Moreover, operators can access the cloud to obtain information or to interact with robot agents by for example decreasing their velocity on security issues.

The MRTA problem consists of the efficient assignment of a set of tasks to a set of robots. Its taxonomy has been well defined in [13] and our system corresponds to a decentralized version of the "Single-Task Robots, Single-Robot Tasks, Instantaneous Assignment (STSRIA)" classification.

Centralized control approaches were used more likely in the past as they provide optimal solutions to the MRTA problem. However, as explained in [14] and due to robustness and scalability problems, the tendency is now turning to decentralized approaches. In this paper, Khamis presents a decentralized market-based approach, where robots bid for tasks based on their capabilities. For the bidding process the Contract Net Protocol (CNP) [15] has been used. This multi-agent task-sharing protocol is divided into 4 stages: Task announcement by an agent that takes the role of the coordinator; Bid submission by individual agents; evaluation and winner selection; and finally the contract stage of the winning agent.

Our task allocation approach is based on this protocol, whereat the machine fulfills the role of the coordinator and the ATVs participate as bidders. However, in order to increase the robustness, besides assigning a primary robot to perform the task, an additional secondary robot will be assigned. It will supervise the primary robot and assume its task in case of failure. Moreover, the auction process is run locally between machines and the ATVs nearby. This simplifies the approach, reducing the information exchanged and allowing the scalability.

3 DECENTRALIZED CLOUD ARCHITECTURE

3.1 Requirements

In this section the requirements for a novel automatic transportation logistic architecture that contain most of the properties of the previously mentioned Industry 4.0 are presented:

- Agents are independent from each other. A failure or complete removal of an agent must not affect any other entity (unless there is a direct co-working in this moment). The addition of new agents on the system does also not affect other agents.
- Every agent is able to connect to the cloud and exchange information with other agents. Based on this information exchange, agents make autonomous decisions allowing a decentralized control architecture.
- It is assumed that point to point connection can suffer delays or be temporarily interrupted. The system must be able to prevent and endure these situations.
- It is assumed that ATVs could have problems during the performance of their tasks (obstacles on the way, connection lost...). There must be a rapid response of the system to react and solve these problems.

3.2 Decentralized Control Architecture

In order to ensure the required autonomy and modularization, the current hierarchical control structure is replaced by a mesh-like decentralized structure. This limits the top management layer to a strategical goal definition and supervision of the factory, rather than the control of the complete system.

The production machines are integrated with embedded systems to endow intelligence and connexion to the cloud allowing an autonomous control of their material replenishment. Thereby, robots and machines interact over the cloud and self-organize the factory according to the global specifications. Moreover, operators have the possibility to supervise and interact with the system in case of failure or hazard via smart devices.

For the communication, the Robot Operating System (ROS) [16] middleware has been used. ROS provides comprehensive and well-structured libraries and drivers for several robots and sensor devices and a well defined structure for communication. For the communication between agents, it is possible to share information on the cloud for any other agent that is interested (Publisher/Subscriber) or arrange a point to point information exchange (Service/Client).

Figure 2 shows the two main layers that compose this decentralized architecture. Between them, the cloud assures the correct connectivity between modules and offers the possibility to outsource services such as computing power or Data-Base services.

Strategical layer

The strategical layer is responsible for long term operations and has three main functionalities:

- Setting strategical goals: Based on incoming sales orders, the strategical layer defines global goals (e.g. manufacturing of a certain product in predefined quantity) that will be transmitted to the machines. It will not control the organization of the agents in the operational layer, but supervise them and interact with them only if there is a change in the global plans.

- Monitoring: Supervises the current state of the factory and saves historical data for learning, forecasting and summing up, to help building a more intelligent factory in the future.

- Connecting clients and factory: Via a web application, the customer could directly track the progress of his order.

Operational layer

The operational layer is composed of cooperating machines, robots and operators that execute the middle-term operations of the factory. Based on strategical goals defined on the top layer, these individual agents cooperate with each other to autonomously organize the factory.

As focusing on the automations of the transportation system, the agents presented in this work are: production machines, that produce according to specifications from the strategical layers; ATVs, that bring the materials from the interim storage areas to the machines and final products to

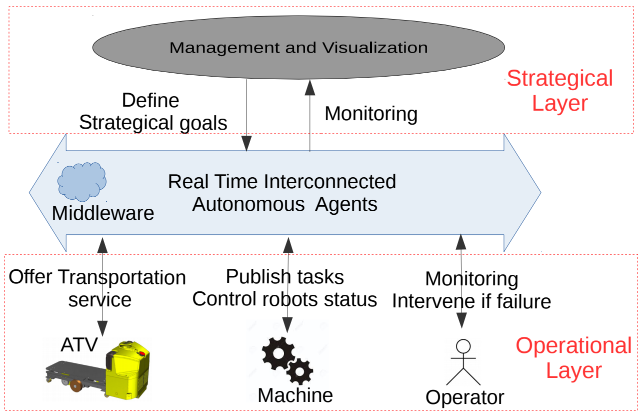

Management and Visualization
Define Strategical goals
Monitoring
Strategical Layer
Real Time Interconnected Autonomous Agents
Middleware
Offer Transportation service
Publish tasks Control robots status
Monitoring Intervene if failure
ATV
Machine
Operator
Operational Layer

Figure 2: Proposed Decentralized Control Architecture. Strategical Layer containing the visualization panel (top). Operational layer containing ATVs, Machines and Operators (bottom)

delivery areas; and operators, that interact with the system in case a hazard is detected.

- *Machine*: Represents the point of use provider or production line. Provided with an embedded system that endues its intelligence and the possibility to connect to the cloud, it autonomously manages its material replenishment. When the machine requires new material provisions, it sends a request to the cloud to contract the services of the most suitable ATV for the replenishment.

The machine updates on the cloud brief information about its internal status named Machine Heart Beat (MHB). This information contains its ID, status, IDs of robots contracted for the transportation, material type required and deadline for the task to be finished. The machines can be connected to the cloud via LAN or WLAN.

- *ATV*: Provides a transportation service on the cloud which is used by the Machines to replenish their material reserves. Each ATV is equipped with an own computational system running on Ubuntu and using the Robot Operating System (ROS) as middleware. Its intelligence depends on an own developed singleton pattern based state machine. Laser scanner data is used for the autonomous localization and navigation in the factory. Robots are necessarily connected to the cloud via WLAN.

As well as the machine, the ATV updates continually on the cloud a few internal information named Robot Heart Beat (RHB). This information contains its ID, status (IDLE, WORKING, ERROR...), localization and brief enlightenment in case a task has been assigned to them. This information can be used by the machines that are utilizing their services, by other robots in case they have to cooperate or by the management panel to supervise the system.

- *Operators*: Using a tablet like device, the operator interacts with ATVs in his surroundings. Though, he must not be able to see or manipulate information about the whole system. Therefore, the operator can only see information about the area he is working in.

Cloud communication

As explained in section 2, the cloud offers several advantages such as resources outsourcing, shared learning and a communication middleware between agents. This work focuses on this communication level. As in the future hundreds of robots, machines, operators, etc. will be connected to the cloud, it is important to create an intelligent communication network not to overload the cloud.

Some information can be directly sent to an specific entity while some other information will be shared with several entities. ROS provides a message passing interface with two information exchange possibilities for these cases: the service-client and the publisher-subscriber. On the one hand, the service-client semantic permits the direct communication between agents. The sender contacts a determined agent and receives back a response from it. The sender knows if the message has been received or not. Moreover, no other agent will notice this information exchange. A machine can thereby create a point to point communication with a desired ATV to receive a material supply. On the other hand, the publisher-subscriber semantic allows the diffusion (publication) of information on the cloud. Every agent interested in this message can subscribe to it. There is no response implicit in this type of communication and therefore the sender does not know who received the information. This communication option will be used for example by the machines and robots to publish their Heart Beat on the cloud. Even though this Heart Beat is limited to a small quantity of bytes, the publication rate will be limited to the smallest required in order not to saturate the cloud

In general a single, central *roscore* master manages all the communications between all the ROS nodes. To allow the decentralization, there are several multi-master packages, e.g. multimaster_fkie [17], that allow every agent to be its own master and exchange information between nodes without the need of a central server.

4 IMPLEMENTATION AND DEPLOYMENT

4.1 Scenario explanation and simulation

In order to test the software, the system has been recreated into the ROS environment and released in the github repository [18]. The factory layout (see figure 1) has been integrated in the management panel as shown in figure 4. Qt has been used for both the creation of the Management Panel and the Operators Panel as it is platform-independent. Furthermore, it is open source, easy to use and has a comprehensive documentation.

The strategical layer defines overarching objectives, based on market requirements or incoming sales orders, providing the respective machines with all data (e.g. bills of material (BoM), CAD-/CAM-Data, etc.) necessary to fulfill these goals. Considering the provided BoMs and CAD-/CAM-data, the machines calculate the necessary amount of material.

For the simulation, the material is supposed to be automatically distributed from the warehouse to the interim storage areas close to the machines. Therefore, this paper focuses on the autonomous transportation of material from these smaller storages to the machines. Fig 3 represents in a sequence diagram the course of events between machines and ATVs for the MRTA:

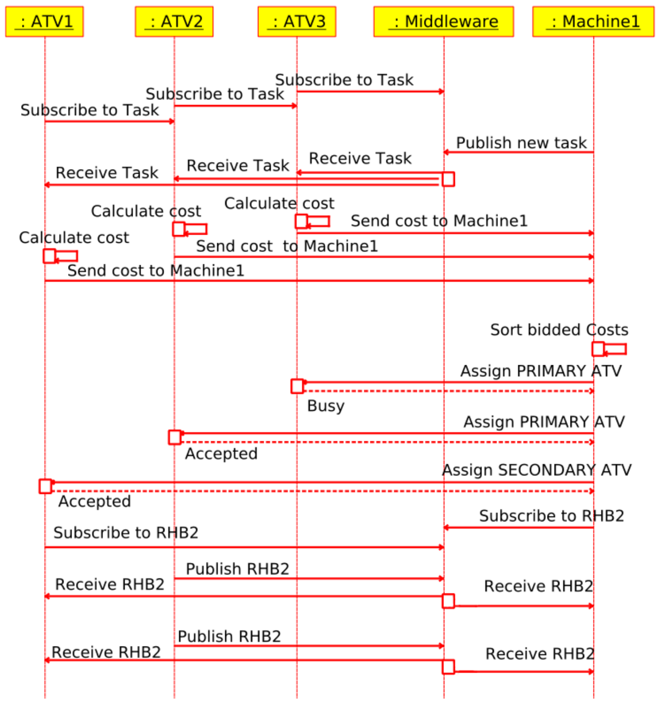


Figure 3: Task allocation sequence diagram for an autonomous machine-ATV material replenishment

The ATVs *subscribe* to the middleware and wait for new transportation tasks. When the machine reaches a specified minimum fill state of raw parts, it *publishes* a new replenishment task on the cloud. The available ATVs receive now the new task directly from the machine and calculate their own specific costs for fulfilling it. These costs consist of the path (ATV-material source-machine), the robot's battery status, and their suitability for the respective type of task (in case diverse ATVs with different abilities are available). Calculated the cost, the robots check if they can perform the task within the given task dead line and, if affirmative, they send the cost directly to the machine. After a certain bidding period, the machine shorts the biddings and tries to contract the services of the ATV with the lowest cost. It is important for the machine to know if the elected robot receives the request and if it is still available. As shown in figure 3, the ATV3 was the robot with the lowest cost but by the time the bidding time finished, it has already been elected by a different machine. As ATV2 has the second lowest cost and is still available, it becomes the PRIMARY robot for the task. To increase the robustness of the system, the machine will select a SECONDARY robot by following the same process (in this case ATV1). The function of the secondary robot, is to be the backup of the primary. In case the primary robot has a problem the secondary one will overtake its task. Once the task is assigned, the ATV navigates to the closest material storage area, performs an autonomous loading maneuver [19] (here simulated) and transports the material to the machine. After fulfilling the task, the robot starts subscribing again for new transportation tasks.

On the bottom of the diagram it can be appreciated how the elected primary robot ATV2 updates its unique RHB2 introduced in the subsection 3.2. The heartbeat allows the interested agents, in this case the machine and the secondary ATV3, to receive a status of the primary robot current situation.

4.2 Validation

Two different control panel prototypes have been developed for the validation of the system. Firstly, a management panel responsible for the strategical order generation and monitoring the entire systems. Secondly, a human-robot interaction panel that displays information of interest to the respective worker and allows him interaction with the robots/ATVs in case of hazard or failure.

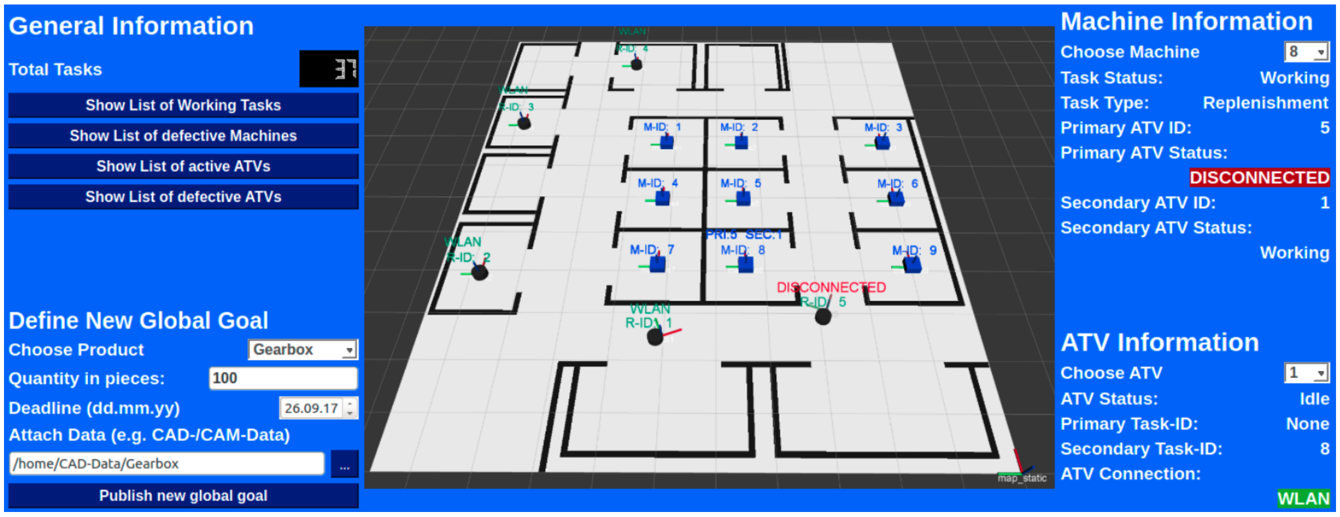
The Management Panel (fig. 4) is mainly used for surveillance purposes. It visualizes the fabrication plant (center) with the machines represented as rectangular blue shapes and the ATVs as circular black shapes. Furthermore, it displays general information about the factory (top left corner), a chosen machine (top right corner) and a chosen ATV (bottom right corner). In addition and for testing purposes, the user is given the opportunity to define and publish the product to be manufactured (bottom left corner). This option should be replaced in the future by an automatic system that reads the incoming orders and publishes the respective answers in the system.

The Human-Robot-Interaction Panel (fig. 5) runs on a tablet-like device and provides the operator with information about the working station he is at, and enables the interaction with the ATVs nearby. This interaction includes modification of ATVs behavior in case of failures such as in case 5 below.

In order to validate the architecture and ensure that it meets the requirements introduced in section 3.1, several cases and the reaction of the system to them are presented below.

CASE 1: FAILURE ON A DETERMINED MACHINE: In this case, an operator intervenes and solves the fault by for example updating the machines software or repairing a hardware component. If it is possible, the machines orders will be distributed between the other machines in the factory. If not, only its orders will be affected while the rest of the production system proceeds as usual.

CASE 2: PEAK WORKLOAD SEASON: In order to react to peak workload seasons it is sufficient to add more agents to the system. The agents that are already on the system will not be affected as every agent makes its own decisions.



General Information

Total Tasks 37

Show List of Working Tasks

Show List of defective Machines

Show List of active ATVs

Show List of defective ATVs

Define New Global Goal

Choose Product: Gearbox

Quantity in pieces: 100

Deadline (dd.mm.yy): 26.09.17

Attach Data (e.g. CAD-/CAM-Data): /home/CAD-Data/Gearbox

Publish new global goal

Machine Information

Choose Machine: 8

Task Status: Working

Task Type: Replenishment

Primary ATV ID: 5

Primary ATV Status: DISCONNECTED

Secondary ATV ID: 1

Secondary ATV Status: Working

ATV Information

Choose ATV: 1

ATV Status: Idle

Primary Task-ID: None

Secondary Task-ID: 8

ATV Connection: WLAN

Figure 4: Management Panel is used for surveillance purposes by visualizing the plant in ROS (*center*) and displaying several information about the plant (top left), the machines (top right) and the ATVs (bottom right). Additionally, it is used to define and publish global production targets (bottom left).

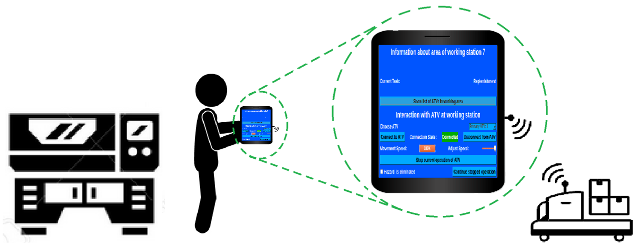


Figure 5: Operator at the machine connects to and interacts with an ATV via handheld device

Case 3: Network connection problems: A connection problem could on the one hand prevent the ATV from bidding for new tasks. To solve this issue, enduring bidding periods are permitted. This extends the time from the task publication to the contract of the ATV. However, it is enough to consider it in the replenishment time estimation and publish the order on ahead. On the other hand, a network problem can prevent the ATV from publishing its RHB. If this occurs during a transportation task, the system could react (assuming there is a failure on the primary ATV) and start a new duplicated transportation task. To avoid this problem, an additional ad-hoc network could be implemented on the ATVs. Thus, if an ATV has WLAN connection problem (e.g. too far from an access point, or huge latencies), it will still have the possibility to communicate its status to nearby ATVs (that have WLAN connection) that would transmit the information into the cloud.

Case 4: A busy ATV has issues that prevent it from fulfilling its current task: An unavoidable obstacle in the way or an important hardware issue makes it impossible for the ATV to reach its destination. In this case, the problem will be evaluated first. If there is the possibility to solve it within a reasonable time, an operator may be requested to intervene. Otherwise, the secondary ATV could assume the task, or the machine could publish another task request starting a new bidding process.

Case 5: Autonomous loading maneuver fails: The material box might get shifted during loading maneuvers, which could cause damage on the environment, the ATV itself, or in worst case the operators. Therefore, operators must be able to interact with robots by adjusting their speed or completely stopping their current activity. After solving the problem, the operator can confirm the elimination of the fault and allow the ATV to continue its operation.

Case 6: Software Updates: In the current factories updates require to stop a partial area or even the complete factory for long time. Moreover, it may be difficult to determine if the new software contains failures until it is implemented and tested. Modularized agents permit a scalar update of the factory. It is not necessary to stop the whole factory but single agents can be introduced and tested bit by bit before the whole system is updated.

4.3 Experiments

The system has been mainly tested into a simulated environment. However, in order to prove the cases 3 and 4 above, two experiments have been performed whereat the machine 8 is represented by a single-board computer and 3 ATVs are represented by Kobukis (see figure 7). Each Kobuki is equipped with an ODROID-XU4 single-board computer to endow intelligence, a RP-LIDAR laser scanner for localization and mapping, a TP-LINK router that provides WLAN connection and a NeoCortec NC2400 module [20] to generate the additional wireless mesh network communication introduced in case 3.

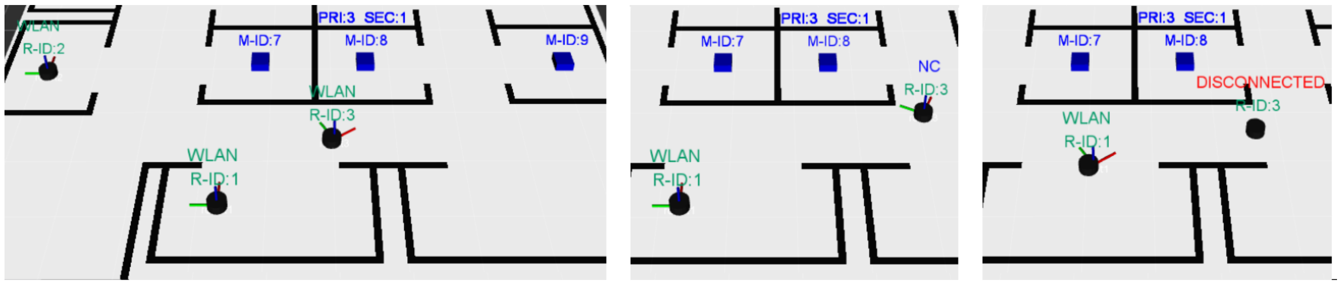


Figure 6: Experiments. Machine 8 publishes a new task and selects ATV3 as primary and ATV1 as secondary robots (left); WLAN connection failure in ATV3 that transmits its RHB via the NeoCortec module (center); Failure in ATV3 represented as disconnected (right).

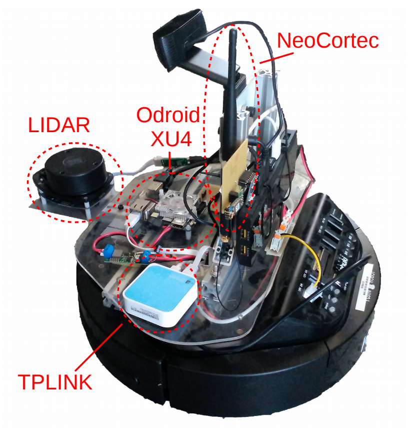


Figure 7: Kobuki platform that represents the ATV for validation tests

We settled a similar environment to fig. 1, created a map from it and let the three robots localize themselves on it. We conducted a use case in which the machine 8 publishes a new material replenishment task on the cloud. The three ATVs send their bidding costs to the machine and once the bidding period finishes, the machine selects the two ATVs with the lowest costs. In this case the ATV3 is selected as primary and the ATV1 as the backup secondary robot. Figure 6 (left) shows: the machine 8 with the selected primary (ATV3) and secondary (ATV1) robots; all three ATVs connected to the cloud via WLAN; and the ATV3 leaving the charging station and thus initializing the transportation task.

In order to test the WLAN connection failure in case 3, the router cable is unplugged from the ATV3 inhibiting the direct connection of the robot to the cloud. As shown in figure 6 (center), the ATV3 appears to be connected via NeoCortec (NC). Even though the TPLINK wireless connection is removed, the robot can yet broadcast its RHB3 through the NeoCortec module to other robots. These robots will now forward RHB3 to the cloud, informing the machine that the task is still being processed. The operators can in any case detect the connection failure and repair it once the delivery has been finished to allow the ATV3 participating in future bidding processes.

To prove the case 4, a hardware issue is simulated by switching off the primary robot (ATV3). In this case, the ATV1 notices that the primary robot is no longer in the system (neither WLAN, nor NeoCortec RHB). Therefore, ATV1 assumes the task and executes it itself. Figure 6 (right) shows the ATV3 as disconnected and the secondary ATV1 starting to perform its task. In this case, the operators notice the disconnection and remove the ATV3 from the factory in order to repair it.

5 CONCLUSIONS

This paper presents and validates a decentralized control architecture for an autonomous transportation system that includes the concepts of the IoT, modularized agents and cloud robotics that compose the future smart factories. We defined a set of requirements and challenges to ensure the robustness, flexibility and scalability that this decentralized architecture offers and validated them over a series of cases and experiments.

The proposed MRTA has been kept simple and local, allowing a massive scalability of the factory. When a machine offers a new task, the ATVs nearby are preferred to participate in the bidding process. This is limited by a distance parameter. In case the machine does not obtain any answer from the robots nearby during the bidding period, the parameter value is increased so more distant ATVs can participate in the bidding. Moreover, each ATV estimates its own cost and sends it to the machine, avoiding complex algorithms to run in any agent.

In the future we plan to move forward with the tests by extending the experiments in the replicated factory layout in our laboratory. The aim is to observe the influence of communication problems such as latencies, test the robustness, and in conclusion observe the reaction of a not simulated system to long working periods. We also plan to increase the number of machines, robots and simultaneously performed tasks in the simulation in order to prove the scalability of the system in complex situations.

REFERENCES

[1] C. Prasse, A. Nettstraeter, and M. T. Hompel, "How IoT will change the design and operation of logistics systems," *2014 Int. Conf. Internet Things, IOT 2014*, pp. 55–60, 2014.

[2] A. Kamagaew, J. Stenzel, A. Nettstrater, and M. Ten Hompel, "Concept of cellular transport systems in facility logistics," *ICARA 2011 - Proc. 5th Int. Conf. Autom. Robot. Appl.*, pp. 40–45, 2011.

[3] C. A. Biffi and A. Tuissi, *Stato dell'arte sulle tecniche di produzione additiva per metalli.* Springer Publishing Company, Incorporated, 2017, vol. 109, no. 1.

[4] K. Furmans, C. Nobbe, and M. Schwab, "Future of Material Handling - modular, flexible and efficient," *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, p. 6, 2011.

[5] K. Furmans, F. Schönung, and K. R. Gue, "Plug and work material handling systems," *Prog. Mater. Handl. Res.*, no. 1, pp. 132–142, 2010.

[6] I. Spectrum, "Three Engineers , Hundreds of Robots , One Warehouse," *Spectrum*, no. July, pp. 26–34, 2008.

[7] R. D. Andrea and P. Wurman, "Future Challenges of Coordinating Hundreds of Autonomous Vehicles in Distribution Facilities," pp. 80–83, 2008.

[8] "RoboCup Industrial @Work League." [Online]. Available: http://www.robocupatwork.org/

[9] G. Hu, W. P. Tay, and Y. Wen, "Cloud robotics: Architecture, challenges and applications," *IEEE Netw.*, vol. 26, no. 3, pp. 21–28, 2012.

[10] R. Arumugam, V. R. Enti, Liu Bingbing, Wu Xiaojun, K. Baskaran, Foong Foo Kong, A. S. Kumar, Kang Dee Meng, and Goh Wai Kit, "DAvinCi: A cloud computing framework for service robots," *2010 IEEE Int. Conf. Robot. Autom.*, pp. 3084–3089, 2010.

[11] Amazon, "Amazon Web Services," 2006. [Online]. Available: https://aws.amazon.com/

[12] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Gálvez-López, K. Häussermann, R. Janssen, J. Montiel, A. Perzylo, B. Schießle, M. Tenorth, O. Zweigle, and R. De Molengraft, "RoboEarth," *IEEE Robot. Autom. Mag.*, vol. 18, no. 2, pp. 69–82, 2011.

[13] B. P. Gerkey and M. J. Matarić, "A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems," *Int. J. Rob. Res.*, vol. 23, no. 9, pp. 939–954, 2004.

[14] A. Khamis, A. Hussein, and A. Elmogy, *Multi-robot task allocation: A review of the state-of-the-art.* Cham: Springer International Publishing, 2015, vol. 604, pp. 31–51.

[15] R. G. Smith, "Correction to The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver," *IEEE Trans. Comput.*, vol. C-30, no. 5, p. 372, 1981.

[16] "Robot Operating System (ROS)." [Online]. Available: http://wiki.ros.org/

[17] A. Tiderko, "Multimaster_Fkie," 2016. [Online]. Available: http://wiki.ros.org/multimaster{_}fkie/

[18] "Kobuki fleet repository." [Online]. Available: https://github.com/Jonmg/kobuki{_}fleet/

[19] J. Martin, T. Fink, S. May, C. Ochs, and I. Cabanes, "An Autonomous Transport Vehicle in an existing manufacturing facility with focus on the docking maneuver task - IEEE Xplore Document," in *3rd Int. Conf. Control. Autom. Robot. (ICCAR), 2017*, 2017.

[20] "NeoCortec." [Online]. Available: http://neocortec.com