Implementation of a Cloud-based Service-Oriented Architecture for Hardware Control Systems supported by Neural Network

O. Latka & J. Provost
Department of Mechanical Engineering
Technische Universität München, Germany

ABSTRACT: Embedded systems control and regulate applications in everyday life, thus changing the conventional way of work to a human-robot symbiosis. This paper describes a new approach of controlling mobile robots through a cloud-based service-oriented architecture supported by neural networks. The proposed architecture permits to operate with large and complex networks even on mobile robotic systems as illustrated on a case-study. Therefore the proposed architecture is based on dividing the management for the high-level on a computer or server and for the low-level directly on the mobile robot. Aiming at copying the cognitive capabilities of human beings, the system is reducing the amount of required sensors significantly. Successfully applied experimentally, the proposed architecture could also be replicated in various other industrial situations.

1 INTRODUCTION

Embedded systems control and regulate applications in everyday life, thus changing the conventional way of work to a human-robot symbiosis. Recently, several topics emphasize more and more on data management and decision making for pervasive automation systems. Topics such as Industry 4.0 (Industrie 4.0 Working Group 2013), Industrial Internet of Things (Atzori, Iera, & Morabito 2010, Evans & Annunziata 2012), and cloud robotics (Kehoe, Patil, Abbeel, & Goldberg 2015) are gaining momentum in the field of industrial and robotic automation.

Humans started to outsource service tasks to robots, but the requirements for this services became more complicated every year. Classic robotic system has to carry adequate physical processing power, various sensors and actors for fulfilling their jobs like visual navigation, path planning, recognition and scene analysis (Wang, Liu, & Meng 2012). However, developing a universal robot covering all possible services is infeasible due to the limitation of costs, power consumption, reliability, sensory, payload or kinematic constrains (Wang, Liu, & Meng 2015). Cloud enabled robots are currently an active developing and research field. The last years, robotic systems have been built around the paradigm of a cloud computing and service-oriented architecture, for extending the functionality of robots. Network technologies enables rapid improvements in term of performance and accessibility by outsourcing services to web, enabling new application such as providing robots access to web facilities and reducing software and hardware costs to create cheaper, lighter and more advanced intelligent robots. Enabling functionality like object recognition and voice services on demand, the sense of touch and creating a shared knowledge base in the cloud accessible for several robots (Doriya, Chakraborty, & Nandi 2012) gets possible to be implemented.

In this paper, a novel approach of controlling mobile robots using cloud-based service-oriented architecture supported by neural networks is described. This approach permits to operate with large and complex networks even on mobile robotic systems. As a proof of concept, a case-study combining software and hardware components is used as an illustration. Therefore the service-oriented architecture is composed to be distributed on several platforms. All software modules communicate over TCP socket using the programming language independent messaging protocol JSON (MacKenzie, Laskey, McCabe, Brown, Metz, & Hamilton 2006).

The proposed architecture is based on dividing the management for the high-level tasks on a computer or server and for the low-level tasks directly on the mobile robot. A state machine is taking care of the critical decisions, a neural network, however, is controlling the cognitive capabilities. Output data of the neural network can also be used for decision making and controlling of the process carried out by the state machine. The neural network is able to handle available and reliable measurements from sensors as inputs and to produce predicted values (Girish, Lam, & Jayaram 2003). The division of tasks on high- and low-levels controllers permits to benefit from huge computational power for complex tasks while using controllers with limited computational power on the embedded system. Also, in the event of a disruption of the connection between the high- and low-levels controllers, the system is still able to complete low-level tasks independently (if information is required from a high-level controller, the low-level controller would then halt waiting for the connection to be reestablished).

Trying to copy the cognitive capabilities of persons, the case-study robot is using its ultrasonic and compass sensors like human senses, just getting the distance to the next object and its orientation to evaluate the environment. Thus, it is able to build up a map for localization. All needed information are stored in a database, accessible for the mobile system as well as for the human user. A recursive MySQL database structure is used to retrieve historic neural network test, sensor and map data. The robot is moving based on the sensor and map information combined and calculated in the neural network and localization section.

The software module responsible for controlling the hardware is divided in two parts. One part is handling the TCP socket communications, logging and processing the data, the other part, used as a library, connects the hardware platform and is responsible for the data transmission to several microcontrollers. Thus, recompiling and testing the hardware is not necessary when changing or adding redundancy hardware. In this approach several microcontroller boards on the mobile robot are taking over the low-level actions to read in sensors, control actors and providing this information to other services. The experiment's platform is composed out of two parts.

The case study is illustrated in details in section 6 is based on a mobile self-constructed robot using ARM-Cortex-M0, ARM-Cortex-M4 and Arduino Boards with ultrasonic and/or compass sensors and a stationary computer processing the state machine, neural network, database, websocket server and hosting the HTML interface. For controlling the robot a mobile phone is used to browse the webpage. The results support the ideas and concept of the initial model. The successfully implementation of the proposed architecture on a case study could also be replicated in various other industrial situations.

2 MOTIVATION AND CHALLENGES

2.1 *Motivation*

One of the most interesting tasks that the human brain is capable of doing is to recognize the environment. Our brain distinguishes known surroundings almost momentarily despite of changes. People are able to combine the information from few sensors to a picture of the whole surrounding, without having as much computational power as modern computers. Over thousands of years the humans developed a minimum, but redundant set of sins, as the sight, the taste, the smell, the touch and the hearing, which are essential for recognizing the environment.

Therefore the question arises, why do modern Robots have to carry power consuming computers along? Is it possible to reduce the calculating and the large amount of sensors on a mobile robot by transforming the highly optimized and efficient bionic technologies to modern technology? Comparing with the humans behavior, obstacle recognition, localization, movement, sense of touch and learning are essential skills for the robot to recognize the environment.

The motivation of the paper is to build up a software architecture for robots, enabling to use all listed methods above operating reliable in unknown and for humans unsafe environment.

2.2 *Challenges*

Developing reusable robotic software is difficult, primarily due to the variability in robotic platforms (Nesnas, Simmons, Gaines, Kunz, Diaz-Calderon, Estlin, Madison, Guineau, McHenry, Shu, et al. 2006). The list of challenges below is not complete, but it is focused on the key challenges for developing a distributed, highly adjustable, neural network using robot. For operating in an unknown environment, which is hostile to life and therefore impossible to gain a broken machine, the robot shall meet the following challenges:

Control heterogeneous robots

Currently, there are no standard robotic platforms or framework flexible enough to address the variations in robots. The system has to handle differences in the sensor configuration, physical capabilities and hardware control. For example, an autonomous car can be composed out of different number of wheels, but it is also possible to use legs or create a hybrid using both, however, it is solving the same tasks in the end. This physical variability leads to different capabilities. The software, which will be deployed across several various platforms, has to provide a generic interface that handles these constraints.

Integrate new capabilities

The fast technological progress will produce more effective and powerful actors, sensors and computational platforms, providing more abilities. The software architecture has to consider modular self-reconfigurable structures for updating and upgrading the software modules automatically or remotely making significant technological advances to the field of robotics in general (Nesnas, Simmons, Gaines, Kunz, Diaz-Calderon, Estlin, Madison, Guineau, McHenry, Shu, et al. 2006).

Computational power limits
One key challenge for robotic is the power consumption. Processing sensor data and controlling actors are high resources taking tasks. Thus mobile robot suffers from a limited operating time and physical processing power. There are two ways of solving this problem. At first reducing the amount and type of sensors and actors. Secondly distributing the calculation to stationary computer. Therefore, the mechanical and electric hardware has to be changed accordingly and algorithms for distributed robotic software architectures has to be implemented. The CPU heavy tasks, such as image recognition, voice recognition, path planning can be performed in the cloud which reduces the energy consumption up to 40 (Jordan, Haidegger, Kovács, Felde, & Rudas 2013).

Artificial robots
Artificial intelligence improves robots with the capability of making their own decisions. Their sensors mostly have low accuracy because of technological restrictions and absence of built-in correction means (Sachenko, Kochan, Turchenko, Tymchyshyn, & Vasylkiv 1999). Dealing with uncertainties will open a wild field of tasks, which robots cannot fulfill currently. Neural networks provide the robot the ability to get trained and make their own decisions.

Accessibility of the robots
Cloud robotics can be divided into two areas, accessibly to the robot and increasing the limited computational capabilities. Transforming the potential of robotics will enable poor equipped robots to fulfill complex tasks by reducing the hardware limitation, while large amount of available resources and parallel computing capability are accessible in the cloud (Wang, Liu, & Meng 2012). The second area concentrates on monitoring and controlling the robot. The goal is to achieve telepresence for getting fully capable virtual representation, as well as to perform complex tasks. Supporting the robot with services about the environment, like Google maps for navigation, could exceed the limit of its on-board capacity (Jordan, Haidegger, Kovács, Felde, & Rudas 2013).

Implement a generic framework
A software framework is providing generic software functionalities, as part of a larger software platform, to reduce the development time. Frameworks include support for libraries, tool sets, application programming interfaces and further more to implement software projects. In computer systems, a framework is often a layered structure indicating what kind of programs can or should be built and how the different software parts would interrelate. Following requirements are set:

- Adaptable
- Updatable
- Upgradable
- Adjustable without recompiling
- Monitoring
- Cloud ready
- Multiple clients
- Supporting real and simulated platforms
- Addressing distributed computation

3 SOFTWARE BACKGROUND AND STATE-OF-THE-ART

3.1 *Robotic Framework*

Frameworks are providing generic functionality for programming software. The DAvinCi framework is used for generating 3D models of environments allowing robots to perform simultaneous localization and mapping (SLAM) (Arumugam, Enti, Bingbing, Xiaojun, Baskaran, Kong, Kumar, Meng, & Kit 2010). For addressing the current limitations of computational power a robot cloud center is designed, called RoboEarth, enabling robots to autonomously share information concerning description of the environment, object models as well as the assigned tasks. This framework allows the user to run their software with minimal configuration in the cloud (Hunziker, Gajamohan, Waibel, & D'Andrea 2013). The Robotic Operating System (ROS) is also providing a framework and will be discussed further more:

3.2 *Robotic Operating System*

The Robot Operating System is a software framework for developing robotic software and providing functionality like an operating system. Developed in 2007 by the Stanford Artificial Intelligence Laboratory in support of the Stanford AI Robot project, the development was continuing primarily at Willow Garage in 2008 The ROS is providing operating system services such as:

- Hardware abstraction
- Low-level device control
- Implementation of commonly used functionality
- Message passing between processes
- Package management (Adiprawita & Ibrahim 2012)

Service tasks are requested and respond between nodes. A collection of libraries and tools provide several services, easy to include in the developer's robotic system are available for Unix based operating systems. In the past years ROS has become by far the most wide-spread robot library (Jordan, Haidegger, Kovács, Felde, & Rudas 2013). This paper gets also proved by the group of supporter, such as Google and NASA. The *rosjava* package was developed by Willow Garage and Google, aiming to become a leader in the field of Cloud Robotics. Google is also developing an Android cloud robot in collaboration with NASA (Jordan, Haidegger, Kovács, Felde, & Rudas 2013).

Although the researches has improved, there are still challenges to be further addressed. Respect to the benefits mentioned previously, providing costless and seamless services is one of the most meaningful field to implement. For simplification the most of the developer assume unlimited resources in the cloud. But, for instance network bandwidth, CPU occupancy for parallel computation and the available number of hosts are limited. Designing a module which maximize the utility of all available resources is a challenging problem. Furthermore, ROS does not support queue management, real time robotic tasks, microcontroller support and is not sufficient for multi robot systems (Wang, Liu, & Meng 2012). This paper was motivated by creating a cloud-based distributed service-oriented architecture supported by neural networks which solves the above mentioned problems and is providing the following aspects:

- Publish / subscription service
- Redundancy and adjustable hardware
- Low computational power
- Executable on every platform
- High- and Low-level support
- Artificial Intelligence with online learning
- Web based monitoring and controlling interface
- service-oriented architecture
- Language independent Messaging
- Microcontroller support
- Master–Slave messaging
- Modular software and hardware design
- Tool based design
- Thin software modules
- Distributed computation in the Cloud
- Low power consumption
- Built-in logging
- High reliability of the controlling software

4 SOFTWARE AND HARDWARE ARCHITECTURE

The selection of suitable software framework and hardware for the robotic system was a crucial step for the development of the modular, distributed, redundant software oriented robot. The commonly used framework ROS which was discussed in chapter two, nevertheless, is not able to fulfill the desired tasks.

4.1 *The proposal for the software and hardware architecture*

This automated, robotic systems are highly dependent on the robustness and quality of their control software while coworking next to human employees (Knight 2015). This paper describes a new approach of controlling mobile robots while using cloud-based distributed service-oriented architecture supported by neural networks. This proposed architecture permits to operate with large and complex networks and a set of parameters even on mobile robotic systems and in unsafe environment. Therefore, the service-oriented architecture is composed to be distributed on several platforms. The proposal is based on dividing the management for the high-level on a stationary computer or server and for the low-level direct on the mobile robot. A state machine, named *Statemachine* in the remainder of the paper, is taking care of the critical decisions, the neural network however, is controlling the cognitive capabilities. Output data of the neural network can also be used for decision making and controlling of the process carried out by the Statemachine. The neural network is able to handle available and reliable measurements from sensors as inputs and to produce predicted values (Girish, Lam, & Jayaram 2003).

4.2 *Motivation*

One of the most interesting tasks that the human brain is capable of doing is to recognize the environment. Our brain distinguishes known surroundings almost momentarily despite of changes. People are able to combine the information from few sensors to a picture of the whole surrounding, without having as much computational power as modern computers. Over thousands of years the humans developed a minimum, but redundant set of sins, which are essential for recognizing the environment. Aiming to copy the cognitive capabilities of persons, the robot is using its ultrasonic and compass sensors like human senses, just getting the distance to the next object and itselfs orientation to evaluate the environment. Thus it is able to build up a map for localization. All needed information are stored in a database, accessible for the mobile system as well as for the human user. Build on a recursive MySQL database structure to retrieve historic neural network test, sensor and map data. The robot is moving based on the sensor and map information combined and calculated in the neural network and localization section. The software module responsible for controlling the hardware is divided in two parts. One part is handling the TCP socket communicating, logging and processing the data, the other part as a library connects the hardware platform and is responsible for the data transmission to several microcontrollers. Thus recompiling and testing the hardware is not necessary when changing or adding redundancy hardware. In this approach several microcontroller boards on the mobile robot taking over the low-level actions to read in sensors, control actors and providing this information to other services. The experiment's platform is composed out of two parts. A mobile, self-constructed robot using ARM-Cortex-M0, ARM-Cortex-M4 and Arduino Boards with ultrasonic and/or compass sensors and a stationary computer processing the state-machine, neural network, database, websocket server and hosting the HTML interface. For controlling the robot, a mobile phone is used to browse the webpage. The results support the ideas and concept of the initial model. Verified by the experience, this work allows to replicate the concept in other industrial situations and to benefit of the results.

4.3 *Hardware*

The proposal is based on dividing the management for the high-level on a stationary computer or server and for the low-level direct on the mobile robot. The Statemachine is taking care of the critical decisions, the neural network however, is controlling the cognitive capabilities. Output data of the neural network can also be used for decision making and controlling of the process carried out by the state machine. The neural network is able to handle available and reliable measurements from the ultrasonic and compass sensors as inputs and to produce predicted values.

4.3.1 *Robot Hardware*

In this approach several microcontroller boards on the mobile robot taking over the low-level actions to read in sensors, control actors and providing this information to other services. The experiment's platform is composed out of two parts. A mobile, self-constructed robot using ARM-Cortex-M0, ARM-Cortex-M4 and Arduino Boards with ultrasonic and/or compass sensors and a stationary computer processing the state machine, neural network, database, websocket server and hosting the HTML interface. The connection to the stationary computer is established using a Wi-Fi router. All sensors, actors and microcontrollers are attached to a robot car chassis. The ideas and concept for the software architecture is verified by the experience on the build robot.

4.3.2 *Software module for the mobile hardware*

The software module responsible for controlling the hardware is divided in two parts. One part is handling the TCP socket communication, logging and processing the data, the other part, designed and implemented as a library, connects the hardware platform and is responsible for the data transmission to several microcontrollers. Thus, recompiling and testing the hardware is not necessary when changing or adding redundancy hardware.

5 SOFTWARE IMPLEMENTATION

This section presents the software implementation for the proposed cloud-based service-oriented architecture supported by neural networks. The software is divided into different modules:

- Statemachine and Orchestra
- Database connection module
- Hardware and Hardware Wrapper module
- Neural Network module
- Localization module
- Websocket server and Websocket client

In order for a structured program the architecture has to be set and should be adhered by every software module. The service-oriented architecture is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. Therefore, distinct units, which combines separate functions, requires to be loosely coupled.

The several parts communicate with their corresponding partner by passing data in a defined, shared format. Nevertheless, there is the need for a router to queue, in case of a lost connection, and route all messages to the right module. Thus, in the event of a lost connection, all messages will be forwarded as soon as the connection is reestablished. Meanwhile, the low-level controllers would continue performing their tasks as long as they can do so independently; if a connection to a cloud-service is required on a periodic way (and programmed adequately), the low-level controllers would then run a safe-stop routine. A Statemachine takes care about the critical decision making, outsourcing the subtasks to the correspondingly software modules. The Statemachine transmits tasks to several software modules, as described in the following.

Database connection
for performing the CRUD (create, retrieve, update, delete) actions.

Hardware and hardware wrapper module
are creating the connection between stationary computer and mobile microcontroller. The hardware wrapper is designed as a library which is included in the main hardware module. This way changing or/and adding boards or new functionality do not require a new compilation of the main hardware module.

Neural Network module
is supporting the Statemachine in the decision making process. Cognitive capabilities, sensor input and database input are combined and the result is sent back to the Statemachine.

The Localization module
recognizes structures for setting the orientation and position of the robot and planning the path to a target point. For autonomous robots this function is essential. For instance, if a command for a robot to turn right is sent by a high-level controller, the low-level localization module will allow this operation to be performed only if the collected data from the surrounding environment permit to do so (e.g. absence of obstacle).

Websockt server and Websocket client
are providing the graphical user interface. The websocket server is transferring messages with the Statemachine, the websocket client is receiving commands from the websocket server. A HTML webpage with included Java Script as the websocket client is creating the access point on the client side for the user, exchanging content with the websocket server.

The described software architecture is presented in figure 1.

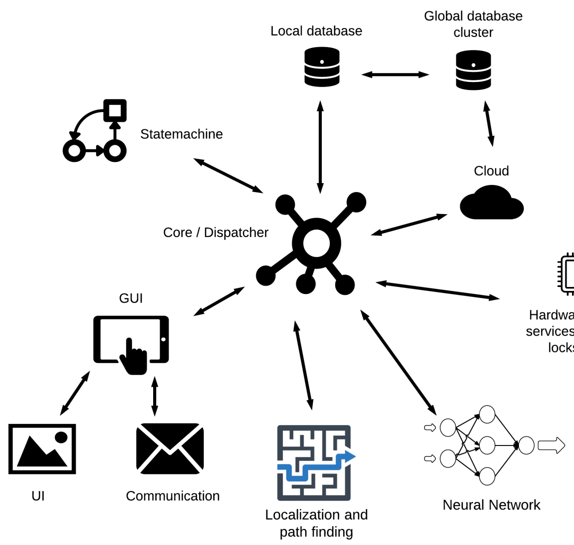

Figure 1: A digramm of the proposed software architecture.

All software modules, except the Websocket server and client, are based on the same structure. The programming language for the software modules is pure C++. The common parts are separated as a library. This way updating the common software parts have effect on all subscribed software modules, assure sharing the same basics, except for the GUI parts. Proving the ability of distributed software, the websocket client is written in HTML and Java Script talking to the websocket server, written using the C++ framework Qt. In this approach the database module is performing the CRUD actions on a MySQL database.

6 EXPERIMENTS AND RESULTS

The experiments are performed in an unknown, indoor environment providing sufficient Wi-Fi reception on a six wheel robot chassis (see fig. 2).

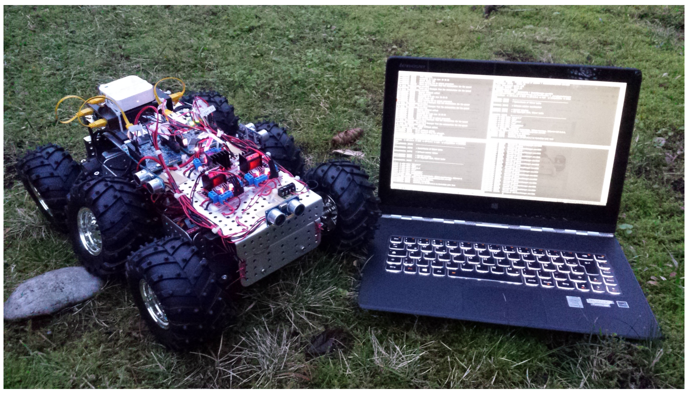

Figure 2: The experimental robot.

The software modules on the stationary computer are started via a controlling script. All information about the outcome of this experiment are temporary saved in the log file. For more accurate results, and especially for checking the duration between the exchanged messages, the time resolution is increased to microseconds. The processes list is adjusted to the experimental setup for starting the Core, the Hardware Control, the Websocket server, HTML GUI, Database connection and Neural Network. Within one second all listed modules created the TCP socket connection and are ready for communication.

- Core with Statemachine and orchestra
- Neural Network
- Database connection
- Websocket server
- HTML GUI
- Hardware control with wrapper

As shown by the experiment and verified by the log, the Statemachine noticed the connection of all modules. Meanwhile the Hardware Module checked the IP addresses of all available microcontrollers. After the initialize process the Statemachine retrieves the training set by the Database connection. Similar to the other software modules, the connector to the database is handling all MySQL procedures, including connecting to the database, translating commands to MySQL queries, refurbish the retrieved data and responding to the Statemachine. In case of the database module the procedure of creating, retrieving, updating or deleting will not change, suited to outsource the queries of the source code for avoiding merging MySQL and C++. The Statemachine transmits the received training set to the Neural Network implying the teaching.

Meanwhile the Statemachine is waiting to receive commands the user provided by the HTML interface remotly. The robot is performing until completion or an updated task is provided. The task gets started by displaying all sensor values to the graphical user interface and requesting commands for the cognitive capabilities by the neural network. Almost instantly the response arrived and is ready to be conducted. The Neural Network module's internal structure is not based on if clauses or switch case statements, where the incoming values have to be compared to a predefined target within a certain time. The decision making is performed by simple additions and calculating thresholds between conjunct neurons and proving results based on this procedure. As the experiment proves, this extends also to uncertainty situations, where the neural network decision interstates between known circumstances. Within the limits of the task for avoiding obstacles, the software system overcome all difficulties and validated the proposed software architecture and hardware.

Cleaning robots, as a related industrial use, demonstrate the suitability for daily use of technical inferior systems and the possibility for improvements in this field.

The results of this experiment support the ideas and concept of the initial model. Verified by the experience, this work allows to replicate the concept of the proposed architecture in other industrial situations and to benefit of the current results.

Future work considers the evaluation safety implications associated with each component of the proposed architecture and the implementation of robust state-of-the-art solutions when possible (e.g. for the message routing functionality).

REFERENCES

Adiprawita, W. & A. R. Ibrahim (2012). Service oriented architecture in robotic as a platform for cloud robotic (case study: human gesture based teleoperation for upper part of humanoid robot). In *2012 International Conference on Cloud Computing and Social Networking (ICCCSN)*.

Arumugam, R., V. R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F. F. Kong, A. S. Kumar, K. D. Meng, & G. W. Kit (2010). Davinci: A cloud computing framework for service robots. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 3084–3089. IEEE.

Atzori, L., A. Iera, & G. Morabito (2010). The internet of things: A survey. *Computer networks 54*(15), 2787–2805.

Doriya, R., P. Chakraborty, & G. C. Nandi (2012). Robotic services in cloud computing paradigm. In *Cloud and Services Computing (ISCOS), 2012 International Symposium on*, pp. 80–83. IEEE.

Evans, P. C. & M. Annunziata (2012). Industrial internet: Pushing the boundaries of minds and machines. *General Electric*, 21.

Girish, T., S. Lam, & J. Jayaram (2003). Reliability prediction using degradation data–a preliminary study using neural network-based approach. In *Proc. European Safety and Reliability Conference (ESREL 2003)*, pp. 15–18. Citeseer.

Hunziker, D., M. Gajamohan, M. Waibel, & R. D'Andrea (2013). Rapyuta: The roboearth cloud engine. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 438–444. IEEE.

Industrie 4.0 Working Group (2013). Recommendations for implementing the strategic initiative industrie 4.0. acatech.

Jordan, S., T. Haidegger, L. Kovács, I. Felde, & I. Rudas (2013). The rising prospects of cloud robotic applications. In *Computational Cybernetics (ICCC), 2013 IEEE 9th International Conference on*, pp. 327–332. IEEE.

Kehoe, B., S. Patil, P. Abbeel, & K. Goldberg (2015, April). A survey of research on cloud robotics and automation. *IEEE Transactions on Automation Science and Engineering 12*(2), 398–409.

Knight, W. (2015). Inside amazon. *MIT Journal*.

MacKenzie, C. M., K. Laskey, F. McCabe, P. F. Brown, R. Metz, & B. A. Hamilton (2006). Reference model for service oriented architecture 1.0. *OASIS standard 12*.

Nesnas, I. A., R. Simmons, D. Gaines, C. Kunz, A. Diaz-Calderon, T. Estlin, R. Madison, J. Guineau, M. McHenry, I.-H. Shu, et al. (2006). Claraty: Challenges and steps toward reusable robotic software. *International Journal of Advanced Robotic Systems 3*(1), 023–030.

Sachenko, A., V. Kochan, V. Turchenko, V. Tymchyshyn, & N. Vasylkiv (1999). Intelligent nodes for distributed sensor network. In *IEEE INSTRUMENTATION AND MEASUREMENT TECHNOLOGY CONFERENCE PROCEEDINGS*, Volume 3, pp. 1479–1484. INSTITUTE OF ELECTICAL ENGINEERS INC (IEEE).

Wang, L., M. Liu, & M. Q.-H. Meng (2012). Towards cloud robotic system: A case study of online co-localization for fair resource competence. In *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*, pp. 2132–2137. IEEE.

Wang, L., M. Liu, & M. Q.-H. Meng (2015). Real-time multisensor data retrieval for cloud robotic systems. *Automation Science and Engineering, IEEE Transactions on 12*(2), 507–518.