Software engineering multi-robot applications

Sergio García
sergio.garcia@gu.se
University of Gothenburg, Gothenburg, Sweden

ABSTRACT

The number of robotic applications that are being developed is increasing exponentially both in industry and academia. However, those applications do not have well-defined system development process. This fact leads to different time consuming tasks as the necessity of starting projects from scratch for most of the robot developers, the inability of reusing already developed packages, exchanging the already implemented ones, etc. On the other hand, robot applications are increasingly based on *teams* of autonomous robots that work collaboratively to accomplish a mission more complex than one that could be achieved by a single robot.

In this PhD project, we aim to apply software engineering to robotic systems in order to produce processes, architectural models and methods to be used by developers in order to tackle common challenges as reusability, variability and modularity. The goal is to cut down the waste of time when developing robotic applications. Furthermore, for our multi-robot applications we study how to manage and exploit emergent behaviours and how to perform a collaborative adaptation based on them. In order to validate our results we make use of different models of service robots in different case studies.

ACM Reference format:
Sergio García. 2018. Software engineering multi-robot applications. In *Proceedings of 40th International Conference on Software Engineering, Gothenburg, Sweden, May 27–June 3, 2018 (ICSE 2018),* 5 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Service robots are increasingly being more and more involved in human lives. They are increasingly used in environments such as houses, airports, hospitals, and offices for performing navigation, transportation, and manipulation tasks. The World Robotic Survey [15] estimated 35 million indoor service robots to be sold by 2018, accumulating a sales value of $12 billion since 2015. The global sales of household and personal robots is expected to grow by 23.5% per year [27]. This increase is accompanied with huge progress in robot technology, especially in image processing, planning, control, and collaboration. Software engineering is key to sustaining this new technology.

A robot typically performs specialized tasks; however, some tasks are highly complex and require a team of robots, whose capabilities (e.g., perception, manipulation, and actuation) are coordinated and supervised. Such teams also need to adapt to changes, such as of the environment, of the desired tasks, or of the robot (e.g., hardware failures). These demands drive the complexity of robot control software relying on appropriate software architectures. To tackle this complexity, we need to rethink design processes [18] by properly managing system integration and raising the abstraction levels, addressing qualities like evolvability [23], configurability [13], scalability, power consumption, and dependability.

The PhD project presented in this paper is involved in the Co4Robots [1] European project. Being part of this project allow us not only to formulate but also to validate our research questions in real-world scenarios. According to [11]: "Usually there are no system development processes (highlighted by a lack of overall architectural models and methods). This results in the need for craftsmanship in building robotic systems instead of following established engineering processes". In fact, most of the robotic applications developed nowadays have to be started from scratch without following well-engineered methods to help in the process. In this context the aim of the Co4Robots project is to establish systematic engineering process to facilitate the development of the software for animating robotic systems through the creation of reusable robot building blocks with well-defined interfaces and properties.

The research conducted during present project will be split in two parts. The first one defines the best practices for engineering software robotic applications. During this period we studied the current software engineering practices for both single and multi-robotic systems. Furthermore, throughout this part of the research we will develop the platform that will be allocated within every robot of our applications. An instance of our *Software platform* will be deployed on each robot. It will integrate the *Software architecture* of the whole system and the *Configuration facilities*, which provide the required tools for configuring our architecture both at design and run-time.

On the other hand, the future work that will be tackled once the platform is completely addressed is group in the second part of the research, and aims to support the *choreography* of robotic applications. In this context, choreography means the way of representing and controlling the interactions between multiple services of a system in a decentralized way. The main expected outcome of this part is to perform the choreography of a deployed team of potentially heterogeneous robots in dynamic environments with the presence of human beings. In order to do so, issues as *Emergent properties* [8, 9] and selecting the most suitable *Collaborative adaptation* [29] techniques must be addressed.

Research Questions. Based on the division of the project, we state the following research questions:

ICSE 2018, Gothenburg, Sweden

DOI: 10.1145/nnnnnnn.nnnnnnn

[1] http://www.co4robots.eu/

RQ1 Which are the current software engineering practices for engineering robotic applications and their limitations?
RQ2 Which software engineering practices can be developed in order to improve the process of engineering robotic applications?
RQ3 Which are the applicable strategies to manage a heterogeneous robotic application with only partial knowledge of a dynamic environment?

Contributions. Our contributions are listed in the following:

(1) Definition of a software architecture able to structure a robotic team;
(2) Implementation of a software platform where all the algorithms and tools developed can be plugged in;
(3) Definition of configuration mechanisms to enable *start-up configuration* and *run-time configuration*;
(4) Integration of an approach based on ROS+REST for the internal communication between robots;
(5) Development of the algorithms in charge of managing the robotic team, based on:
    (a) Management of emergent properties;
    (b) Selection of collaborative adaptation techniques;

At this moment the software architecture and the communication mechanisms are already developed. The software platform is being continuously developed and we intend to start dealing with the configuration facilities in the near future. The development of the algorithms in charge of managing the robotic team are planned to be addressed during the second part of the project.

Organization. Section 2 describes the research approach of the current work and answers the first research question. In Section 3 we present the process that we follow for engineering robotic applications and answer RQ2. Then, in Section 4 we explain our plan for managing a robotic team while answering RQ3. In Section 5, we introduce different works with a similar scope and position our research. Section 6 explains our validation plan. It concludes with Section 7 with final remarks.

2 RESEARCH APPROACH

The plan within this project is to split the work in two parts: (1) the first part, focused on the study of current practices for engineering robotic applications (RQ1) and the development of the framework allocated within each robot (RQ2); and (2) the second part, focused on collaboratively managing a team of robots (RQ3).

This project is being developed closely with two companies[2][3] since they are partners of the Co4Robots European project. The main goal of this project is to deploy a robotic application in a "domestic" environment such as hospitals, hotels, airports, etc. These environments will be considered as dynamic —i.e. changing environments with presence of uncertainty— and with presence of human beings. We consider that robotic applications must be able to accomplish complex missions with a systematic, real-time and decentralized methodology. Furthermore, a robotic application could be composed by a team of potentially heterogeneous robots. For this reason, the robots must have integrated a set of perceptual capabilities that enable them to localize themselves and estimate the state of their highly dynamic environment in the presence of strong interactions and in a collaborative manner. That is, robots must not only interact between them, but also with human beings.

In order to learn which are the current practices for developing robotic applications we performed an extensive research in the field. We also plan to conduct empirical studies with different companies, starting with the industrial partners of Co4Robots. With this steps we expect to answer RQ1 and also to figure out which are the limitations of such practices.

A not negligible task within this project is to decouple the research made just for Co4Robots and the one intended for the whole PhD. While the platform development outcomes are common for both our own research and the Co4Robots' , our outcomes of the collaborative adaptation part are expected to reach far beyond results than the intended by Co4Robots.

[2]https://www.bosch.com/
[3]https://pal-robotics.com/en/home/

3 ENGINEERING ROBOTIC APPLICATIONS

After the research of the current state of the art we proceeded with RQ2, that will be addressed during the first part of the PhD as well. In order to answer it while explaining our proposed process we divided the development of the software to be deployed within each robot in different items.

Software architecture. We propose our software architecture, Self-adaptive dEcentralised Robotic Architecture (SERA) for being integrated within the software platform. In order to create and validate SERA we followed the steps detailed in Section 6. Specifically, we held eight internal meetings, test the work in three simulated scenarios, sent six emails with questions to each partner, held three consortium meetings, wrote one deliverable and held one integration meeting where SERA was tested.

Our architecture supports a real-time decentralized robot coordination to accomplish missions with teams of robots. Furthermore, it is self-adaptive, responding to external and internal events by computing new strategies to achieve the desired goals. SERA is inspired by and extends concepts of existing proposals for robot software architectures from the literature. Specifically, we inspected architectures identified by a mapping study of Ahmad et al. [1], which investigated software architectures for robotics systems to identify and analyze the relevant literature based on 56 peer-reviewed papers. Thus, our architecture is a three layers architecture that is strongly influenced by the well-known work of Kramer and Magee [17]. Furthermore, it is a component-based type of architecture, so functionalities of the system are encapsulated in modules called "components". It is important to remark that the aim of our project is to build a system that can be easily used by not technical users, so we had to define a way for them to command the missions to the robotic team. For this reason we added a central station that is just used during design-time in order to allocate a graphical interface to be used by a final user. We defined SERA by first conceiving an architecture for a single robot. Then, we extended and refined it in order to iteratively extend and refine the architecture towards enabling communication among robots and collective adaptations. Thus, all the robots have a instantiation of the reference architecture but are also able to communicate and share information with the rest of the team, making possible the collective adaptation. For further information regarding SERA we made available a deliverable where we presented it in detail [4].

SERA was already tested during an integration meeting of the project, where it demonstrate that can support the performance of a robot achieving different complex missions —i.e. collaborative transportation with an human being, autonomous driving in a dynamic environment.

Software platform. As explained before, the software platform will integrate the software architecture, all the tools and software created by developers and the configuration facilities. The platform is also a collection of the components that compose the architecture, a kind of library. In our project the components of our architecture represent ROS [24] nodes and packages, so the platform itself should be based on this middleware. All this components are developed abstracting the communication problems since we rely on the interfaces defined in the architecture. It not only significantly reduces the complexity of the code but also triggers the modularity of our system making possible exchanging the components based on the context. In order to achieve the abstraction of the interfaces we defined an abstract class for each component where all the communication code is stored. Then, every time a developer wants to create a new instantiation of a component they create a new class where all the logic is located and that inherits the properties of the abstract one.

Because we not only plan to control the performance of the system that is running in each robot but also its behaviour we implemented FlexBe [26]. It is a high-level behavior engine, flexibly applicable to numerous systems and scenarios. FlexBe not only provides a way of defining the behaviour of the robot in different scenarios (as of the study cases) by can also be used for defining the work flow. It also provides a graphical interface that simplifies enormously these tasks. FlexBe encapsulates functionalities of the robotic application, as our architecture does within components, and provides a way of orchestrate them so we keep the modularization of our system. As with ROS, an instantiation of FlexBe will be deployed in every robot. It improves the modularity, variability and reusability of our system facilitating the development of the software for animating robotic systems through the creation of reusable robot building blocks with well-defined interfaces and properties.

Finally, in order to communicate each robot with its teammates we implemented an approach based on ROS+REST. So, using a suitable component that works as an interface we are able to send messages in form of services between robots.

Configuration facilities. Since the components of our architecture are exchangeable our next short-term goal is to define configuration facilities that can be applied to our system depicted in the architecture. Then, our robotic applications will be able to:

(1) Being customizable at design-time, so we can configure its components based on the requirements of our context (i.e. hardware installed in each robot, environment where they will be deployed, etc.)
(2) To self-adapt or self-configure at run time, so each robot can apply changes in its configuration based on emergent events of the environment or failures of their system.

[4]https://goo.gl/zoekR2

In order to do so we will implement pluginlib [5], a package that uses the ROS build infrastructure and provides tools for writing and dynamically load plugins.

4 SUPPORTING THE CHOREOGRAPHY OF ROBOTIC APPLICATIONS

The future work will be focused on trying to find an answer to RQ3 during the second part of the research. In order to achieve it, a detailed study of the current state of multi-robot choreography regarding emergent properties or collaborative adaptation must be performed. Nevertheless, since our plan is to continuously integrate all the obtained knowledge into the project, the framework developed during the first division of the project will be used and tested through this part as well. An architectural overview of the whole research is depicted in Figure 1. In this figure the expected final system is represented in a schematic way. The aforementioned conceptual and temporal division of the research is expressed with dot-lined boxes. The contents of the box concerning RQ1 comprehend everything since it represents a broad study. Then, the research concerning RQ2 is more focused. The box that represents such research contains a robotic team represented by a group of folded boxes that in turn contain the framework intended for each robot. The communication methodology is also represented in the figure. Then, the techniques that we plan to study in order to perform the choreography are also depicted in the space within RQ3 box. Note that the contents regarding RQ2 are embedded into RQ3.

During the second part of this PhD project techniques for supporting the correct choreography of multiple robots will be studied and applied. Robots should collaborate in a team to accomplish complex missions because often adaptations performed by a single robot are not sufficient to accomplish a specified mission. Tasks may need to be reassigned to other robots, or a team of robots needs to be reconfigured. Collaboration and interaction of robots among themselves and with the environment could lead to emergent properties representing unexpected behaviors. Emergent properties can be beneficial, neutral, or even harmful —for instance, when they hamper safety or the mission accomplishment.

5 RELATED WORK

In this section we discuss related work with respect with the three proposed Research Questions.

Current software engineering practices for robotic applications and their limitations. In the last years some frameworks towards the application of software engineering methods to robotics have been proposed [25]. In fact, the necessity of this application was reflected with the formation of the *Journal of Software Engineering and Robotics* [4]. There are previous works that provide some of the functionalities that we want to implement in our approach, so they provided inspiration to our research. For example, in [28] the authors developed a MDE based tool for defining robotic applications which is used for describing their components, connectors and interfaces. Furthermore, from the existing models the user can generate automatically working code, what helps to avoid problems regarding communication definition or the instantiation of the components. Nevertheless, it is based on systems composed

[5]http://wiki.ros.org/pluginlib

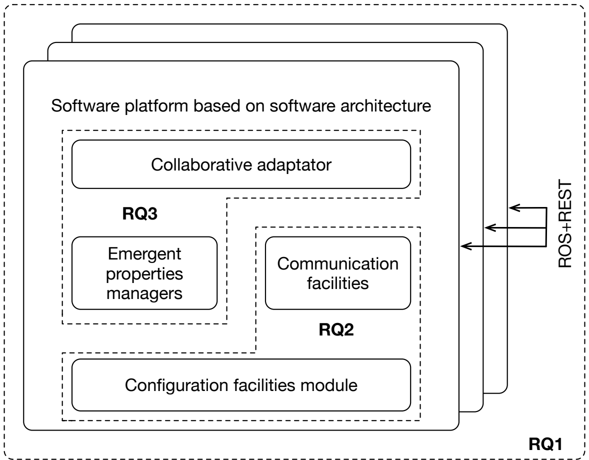

Figure 1: Architectural overview of the final system.

by just one robot. The work of Medvidovic et al. [20] proposed an architecture-based approach that supports heterogeneous robotic systems that are able to self-adapt in dynamic environments. However, this approach does not provide a platform that enables modularity, variability and reusability in the system neither work in a decentralized fashion. A project which aim was similar to the pursued by this work and specially to the pursued by Co4Robots was the BRICS project [2]. The researchers that worked during this project helped to the community providing a continuous increase of the knowledge in our field but most of their work is not maintained anymore. On the other hand, the HyperFlex toolchain, developed by Gherardi et al. [14], presents a way of defining robotic applications by reusing reference architectures instead of just reusing components. It is an extension of the research conducted during the BRICS project. This paper also provides a graphical editor that support the users during the development process of distributed component-based architectures that increase the modularity, reusability and composability of the resulting systems. Yet, they do not cover the multirobot collaboration providing a communication method or defining approaches for supporting the choreography of a team of robots.

Using Software Engineering to improve the process of engineering robotic applications. Several software architectures for robotic systems already exist. Kortenkamp et al. [16] provide an extensive discussion about this topic. The architectural styles followed by most of the authors for developing a software architecture in robotics are the component-based [3, 5, 7] and Service Oriented Architectures (SOA) [12]. Microservices [21] is a variant of the SOA architectural style. The microservices architecture structures an application as a collection of loosely coupled services, which should be fine-grained and the protocols should be lightweight. It helps to improve modularity, but the multiple and complex relationships between all this services (which could be not homogeneous) increase the complexity of deploying a system with these technologies. The Cloud Container Technologies [22] are nowadays experiencing a boom because they are being employed as an extension of the microservices. Those technologies aid the orchestration of applications in distributed topologies and provide enhanced distributed computing capabilities. Nevertheless, since we declined the usage of microservices in our system due to the unnecessary increase of complexity that it would lead to, we also declined the implementation of these technologies.

In fact, component-based software engineering has been broadly used, as explained in [6]. It allows a separation of concerns splitting the functionality of the whole software system into smaller, interchangeable, and configurable components. Between the previously cited works there are architectures that also allow self-adaptation, which is a pivotal feature within the field of robotics. There are also architectures that support decentralized systems following the component-based fashion [19]. A recent survey [29] shows that, even though several works support hierarchical and distributed architectures, most of these only support the same robot type. In this light, SERA strives to be used in a hierarchical and distributed way at run-time.

Managing heterogeneous robotic applications. A brief study of the current state of the art regarding the choreography of multiple robots has been performed in order to solve RQ1. However, we will conduct a deeper study during the second part of this PhD project due to the quick advance of technologies in this field. There are works that list systems that are able to perform self-adaptation in a collaborative way [10]. Recent surveys also analyse multi-robot coordination and strategies to be applied in order to achieve complex missions [29]. Also, we studied how to manage emergent properties from the work of De Angelis et al. [8, 9].

6 VALIDATION

In order to validate the code and artifacts developed for Co4Robots we defined three different approaches than can be followed depending on the artifact that we want to obtain. The first approach consists in presentations of our work during meetings and consortiums and get feedback from stakeholders and practitioners. For the second we make use of simulation tools that allow us to validate our artifacts before implementing them into real robots. Finally, the third one consists in validation with real robots in real-world scenarios. For example, the outline of the approach that we followed for validating SERA is:

(1) Personal work alternated with internal meetings.
(2) First validation by means of simulating scenarios.
(3) Presentation of achieved work to the Co4Robots committee and collection of feedback.
(4) Correction of work based on feedback.
(5) Validation during Milestones and Integration Meetings.
(6) Documentation for deliverables requested for every task within each Workpackage of the project.

In this case we combined the three stated validation approaches. However, there are other cases —such as the validation of the collaborative adaptation algorithms— where we plan to validate our artifacts starting from the second approach and then following with the third one since they do not need validation from the Co4Robots stakeholders.

7 CONCLUSIONS

Software engineering can be the key technology needed for the improvement of applications developed for robotic systems. Robots are nowadays a trend, but without well-defined engineering process for the researchers to follow most of the projects are started from scratch. It results in projects that are not time neither cost efficient. Furthermore, we plan to perform a detailed research in the state of the art in order to comprenhend the optimal way of managing a collaborative team of robotic agents in dynamic environments with human presence. In this project we aim to tackle those problems while validating the premises and obtained results with real-world scenarios where real robots work in a collaborative way.

ACKNOWLEDGMENTS

EU H2020 Research and Innovation Programme, grant 731869 (Co4Robots).

REFERENCES

[1] Aakash Ahmad and Muhammad Ali Babar. 2016. Software architectures for robotic systems: A systematic mapping study. *Journal of Systems and Software* 122 (2016).

[2] Rainer Bischoff, Tim Guhl, and Erwin Prassler. 2010. BRICS - Best practice in robotics. *Robotics (ISR)* (2010).

[3] Victor Braberman, Nicolas D'Ippolito, Jeff Kramer, and Daniel Sykes. 2015. MORPH: A Reference Architecture for Configuration and Behaviour Self-adaptation. In *CTSE*. ACM.

[4] Davide Brugali. 2010. From the Editor-in-Chief : A New Research Community , a New Journal. 1, January (2010).

[5] Davide Brugali, Luca Gherardi, A. Biziak, Andrea Luzzana, and Alexey Zakharov. 2012. A reuse-oriented development process for component-based robotic systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7628 LNAI (2012).

[6] D. Brugali and P. Scandurra. 2009. Component-based Robotic Engineering Part I: Reusable building blocks. *Robotics Automation Magazine* 16, 4 (2009).

[7] Herman Bruyninckx, Markus Klotzbücher, Nico Hochgeschwender, Gerhard Kraetzschmar, Luca Gherardi, and Davide Brugali. 2013. The BRICS component model: a model-based development paradigm for complex robotics software systems. *Proceedings of the 28th Annual ACM Symposium on Applied Computing* (2013).

[8] Francesco Luca De Angelis and Giovanna Di Marzo Serugendo. 2015. *Logic Fragments: A Coordination Model Based on Logic Inference.* Springer International Publishing.

[9] Francesco Luca De Angelis and Giovanna Di Marzo Serugendo. 2016. *Logic Fragments: Coordinating Entities with Logic Programs.* Springer International Publishing, Cham.

[10] Rog??rio De Lemos, Holger Giese, Hausi A. M??ller, Mary Shaw, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Gabriel Tamura, Norha M. Villegas, Thomas Vogel, Danny Weyns, Luciano Baresi, Basil Becker, Nelly Bencomo, Yuriy Brun, Bojan Cukic, Ron Desmarais, Schahram Dustdar, Gregor Engels, Kurt Geihs, Karl M. G??schka, Alessandra Gorla, Vincenzo Grassi, Paola Inverardi, Gabor Karsai, Jeff Kramer, Ant??nia Lopes, Jeff Magee, Sam Malek, Serge Mankovskii, Raffaela Mirandola, John Mylopoulos, Oscar Nierstrasz, Mauro Pezz??, Christian Prehofer, Wilhelm Sch??fer, Rick Schlichting, Dennis B. Smith, Jo??o Pedro Sousa, Ladan Tahvildari, Kenny Wong, and Jochen Wuttke. 2013. Software engineering for self-adaptive systems: A second research roadmap. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2013).

[11] EU. 2016. Robotics 2020 Multi-Annual Roadmap For Robotic in Europe. http://sparc-robotics.eu/wp-content/uploads/2014/05/H2020-Robotics-Multi-Annual-Roadmap-ICT-2016.pdf-. (2016).

[12] Lorenzo FlÃijckiger and Hans Utz. 2014. Service Oriented Robotic Architecture for Space Robotics: Design, Testing, and Lessons Learned. *Journal of Field Robotics* 31, 1 (2014).

[13] Nadia Gamez and Lidia Fuentes. 2013. Architectural evolution of FamiWare using cardinality-based feature models. *Information and Software Technology* 55, 3 (2013).

[14] L. Gherardi and D. Brugali. 2014. Modeling and reusing robotic software architectures: The HyperFlex toolchain. In *IEEE International Conference on Robotics and Automation (ICRA)*.

[15] IFR. 2016. World Robotic Survey. https://ifr.org/ifr-press-releases/news/world-robotics-survey-service-robots-are-conquering-the-world-. (2016).

[16] David Kortenkamp and Reid Simmons. 2008. *Robotic Systems Architectures and Programming.* Springer Berlin Heidelberg.

[17] J. Kramer and J. Magee. 2007. Self-Managed Systems: an Architectural Challenge. In *Future of Software Engineering*.

[18] E. A. Lee. 2008. Cyber Physical Systems: Design Challenges. In *ISORC*.

[19] C. Lesire, G. Infantes, T. Gateau, and M. Barbier. 2016. A distributed architecture for supervision of autonomous multi-robot missions. *Autonomous Robots* (2016).

[20] N. Medvidovic, H. Tajalli, J. Garcia, I. Krka, Y. Brun, and G. Edwards. 2011. Engineering Heterogeneous Robotics Systems: A Software Architecture-Based Approach. *IEEE Computer* 44, 5 (2011).

[21] Sam Newman. 2015. *Building Microservices* (1st ed.). O'Reilly Media, Inc.

[22] Claus Pahl, Antonio Brogi, Jacopo Soldani, and Pooyan Jamshidi. 2017. Cloud Container Technologies: a State-of-the-Art Review. *IEEE Transactions on Cloud Computing* May (2017).

[23] Jennifer Pérez, Nour Ali, Jose A. Carsí, Isidro Ramos, Bárbara Álvarez, Pedro Sánchez, and Juan A. Pastor. 2008. Integrating aspects in software architectures: PRISMA applied to robotic tele-operated systems. *Information and Software Technology* 50, 9 (2008).

[24] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Mg. 2009. ROS: an open-source Robot Operating System. *Icra* 3, Figure 1 (2009).

[25] Arunkumar Ramaswamy, Bruno Monsuez, and Adriana Tapus. 2014. SafeRobots : A Model Driven Framework for Developing Robotic Systems. *IEEE/RSJ International Conference on Intelligent Robots and Systems* Iros (2014).

[26] Philipp Schillinger, Stefan Kohlbrecher, and Oskar Von Stryk. 2016. Human-robot collaborative high-level control with application to rescue robotics. *Proceedings - IEEE International Conference on Robotics and Automation* 2016-June (2016).

[27] Colleen Sheng. 2016. Global Domestic Service Robots Market - Size and Trends to 2020. https://www.alliedmarketresearch.com/robotics-technology-market. (2016).

[28] M. Wenger, W. Eisenmenger, G. Neugschwandtner, B. Schneider, and A. Zoitl. 2016. A model based engineering tool for ROS component compositioning, configuration and generation of deployment information. In *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*.

[29] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. 2013. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems* 10 (2013).