Effective engineering of multi-robot software applications

2018 ACM/IEEE 40th International Conference on Software Engineering: Companion Proceedings

Effective Engineering of Multi-Robot Software Applications

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

ABSTRACT

The number and complexity of robotic applications that are being developed both in industry and academia are increasing continuously. However, those applications are not engineered through well-defined system engineering processes, and this leads to time-consuming issues. Besides, robot applications are increasingly based on teams of autonomous robots that work collaboratively with other robots and/or humans to accomplish complex missions. This further increases the complexity of the controlling application. In this Ph.D. project, we aim to bring software engineering best practices to the robotic domain in order to produce processes, architectural models, and methods to be used by developers in order to tackle common challenges such as reusability, variability, and modularity. The goal is to reduce development time and effort, thereby reducing the time-to-market of robotic applications. To validate our results we make use of different models of real robots in real-world scenarios.

ACM Reference format:
Sergio García. 2018. Effective Engineering of Multi-Robot Software Applications. In *Proceedings of 40th International Conference on Software Engineering , Gothenburg, Sweden, May 27-June 3, 2018 (ICSE '18 Companion),* 2 pages.
DOI: 10.1145/3183440.3183445

INTRODUCTION

Robots are increasingly involved in human lives. They are more and more used in environments such as houses, airports, hospitals, and offices for performing different tasks. The World Robotic Survey [5] estimated 35 million indoor service robots to be sold by 2018, accumulating a sales value of $12 billion since 2015. This increase is accompanied by a huge progress in robot technology. 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 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). Furthermore, since robot applications are nowadays involved in human lives, they have to adapt to human behaviors and to collaborate with humans. These demands drive the complexity of robot control software relying on appropriate software architectures. To tackle this complexity, we need to rethink design processes [7] by properly managing system integration and raising their abstraction level, while addressing qualities such as evolvability, configurability, scalability, and dependability.

ICSE '18 Companion, Gothenburg, Sweden
 978-1-4503-5663-3/18/05…$15.00
DOI: 10.1145/3183440.3183445

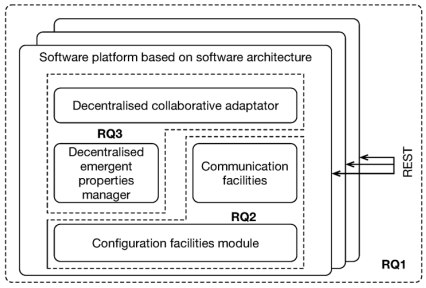

Figure 1: Architectural overview of the final system.

The research conducted during the present project will be split into two parts. The first one defines the best practices for engineering software robotic applications. During this period we study the current software engineering practices for both single and multi-robotic systems. According to the Multi-Annual Roadmap For Robotic in Europe [3], there are no mature system's development processes for robotic applications, meaning that robotic applications are nowadays created from scratch without following systematic-engineering methods. In this light, the aim of this project is to establish systematic engineering process to facilitate the development of the software of robotic systems through the creation of reusable robot building blocks with well-defined interfaces and properties. Robots should collaborate with a team to accomplish complex missions because often actions performed by a single robot are not sufficient to accomplish a specified mission. In this context, choreography means the way of representing and controlling the interactions between multiple services of a system in a decentralized way. Our goal is to perform the choreography of a team of potentially heterogeneous robots in dynamic environments with the presence of human beings. In order to do so, issues as *Emergent properties* [2] and selecting the most suitable *Collaborative adaptation* [9] techniques must be addressed.

Research Questions. We state the following research questions:
RQ1. What are the current software engineering practices for engineering robotic applications and what are their limitations?
RQ2. What software engineering practices can improve the process of engineering robotic applications?
RQ3. What are the applicable strategies to manage an application of multiple and heterogeneous robots with only partial knowledge of a dynamic environment?

Contributions. Our contributions are: (1) a thorough study of the current state of the art concerning engineering practices applied to the robotics domain (RQ1); (2) a software architecture able to structure a robotic team [4] (RQ2); (3) an extensible software platform where all the developed algorithms and tools can be plugged in (RQ2); (4) configuration mechanisms to enable *start-up configuration* and *run-time configuration* (RQ2); (5) an approach to support

the communication among robots (RQ2 and RQ3); and (6) the algorithms and rules in charge of managing the choreography of the robotic team (RQ3). (1) and (3) are being continuously developed (we already performed a study map of software architectures for robot applications [4]). (2) is already developed [4]. (4) will be addressed in the near future and (5) in the second part of the project.

Validation. Our work is performed in the context of the Co4Robots[1] project, what allows us to validate our research questions with real robots in real-world scenarios, since the industrial partners[2] provide three different robots and their facilities for experimentation, both in real-life and simulation. Such scenarios are dynamic and changing with the presence of humans. In order to validate the developed code and artifacts, we defined three different approaches. The first approach consists in getting feedback from stakeholders and practitioners; this has been done, for instance, through presentations of our work during meetings and consortiums. The second one relies on simulation tools that allow us to validate our artifacts before implementing them in real robots. The third one consists of validation with real robots in real-world scenarios.

ROBOTIC APPLICATIONS

RQ1. After a through study of the current literature we conducted study map [4] that is based on and extends the work of Ahmad et al. [1]. In this work we identify software architectures applied to robotic systems and point out their limitations.

RQ2. To answer RQ2 we divide the software to be deployed within each robot in different items.

Software architecture. We inspected architectures identified by a mapping study of Ahmad et al. [1], which investigated software architectures for robotics systems. According to this study, currently there are no architectures that permit the achievement of a mission by teams of robots that collaborate among them and with humans in a decentralized manner. For this reason, we developed the Self-adaptive dEcentralised Robotic Architecture (SERA) [4]. It supports a real-time decentralized robot coordination and adaptation to accomplish missions with teams of robots collaborating with humans. SERA is inspired by and extends concepts of existing proposals for robot software architectures from the literature. Specifically, it is a three layers, component-based architecture that is strongly influenced by the well-known work of Kramer et al [6]. It was developed to support a wide range of applications, i.e. different robot models, environments, missions, etc. SERA was already tested in a real-world scenario, supporting the operation of a robot—i.e. performing collaborative transportation and autonomous navigation.

Software platform. The software platform will integrate the software architecture, all the generated tools and software and the configuration facilities. The platform is also a collection of instantiations created by developers of each component of the architecture. In the current instantiation of SERA, the components of our architecture are represented as ROS [8] nodes and packages (although it was developed to support different middleware). All these 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 enforces the modularity of our system. To achieve the abstraction of the interfaces, we defined an abstract class for each component where all the communication code is stored. Thus, every time a developer wants to create a new instantiation of a component he/she must create a new class that inherits the properties of the abstract one. Finally, in order to enable the communication between robots we implemented an approach based on REST over ROS that uses a component that works as an interface and enables the communication among robots using services (for developing this component we followed the guidelines of already available tools[3]).

Configuration facilities. Suitable configuration facilities will allow our robotic applications: (1) to be customizable at design-time, so we can configure the application exchangeable components based on the requirements of our context (i.e. hardware installed in each robot, environment where they will be deployed, etc.); and (2) to self-adapt or self-configure at run time, so each robot can apply changes in its configuration based on events in the environment that cannot be predicted at design time or failures of their system.

RQ3. Techniques for supporting the correct choreography of multiple robots will be studied and applied [9]. Collaboration and interaction of robots among themselves, with humans, and with the environment could lead to emergent properties (properties that can be observed at a global level but that none of the interacting entities exhibit on its own), representing unexpected behaviors [2]. Emergent properties can be beneficial, neutral, or even harmful; for instance, when they hamper safety or the mission accomplishment.

Conclusion. Without well-defined engineering processes, most of the robotic projects are neither time nor cost efficient. Therefore, we strive to bring software engineering best practices since it can be the key technology for the improvement of applications developed for robotic systems. Furthermore, managing a collaborative team of robotic agents in dynamic environments with human presence is a required feature of nowadays' robotic applications. In this project we aim to solve these challenges while validating the premises and obtained results in real-world scenarios with real robots. The usage of robots in environments populated with humans has some ethical issues, but is something explicitly considered by Co4Robots.

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* (2016).
[2] Francesco Luca De Angelis and Giovanna Di Marzo Serugendo. 2016. *Logic Fragments: Coordinating Entities with Logic Programs.*
[3] 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).
[4] S. García, P. Pelliccione, C. Menghi, T. Berger, and R. Wohlrab. 2018. An Architecture for Decentralized, Collaborative, and Autonomous Robots. On submission.
[5] IFR. 2016. World Robotic Survey. https://ifr.org/ifr-press-releases/news/world-robotics-survey-service-robots-are-conquering-the-world-. (2016).
[6] J. Kramer and J. Magee. 2007. Self-Managed Systems: an Architectural Challenge. In *Future of Software Engineering.*
[7] E. A. Lee. 2008. Cyber Physical Systems: Design Challenges. In *ISORC.*
[8] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Mg. 2009. ROS: an open-source Robot Operating System. (2009).
[9] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. 2013. A survey and analysis of multi-robot coordination. *Journal of Advanced Robotic Systems* (2013).

[1] http://www.co4robots.eu/
[2] https://www.bosch.com/, https://pal-robotics.com/en/home/
[3] https://goo.gl/1RVW54