Engineering the Software of Robotic Systems

Federico Ciccozzi*, Davide Di Ruscio†, Ivano Malavolta‡
, Patrizio Pelliccione§, and Jana Tumova¶
*Mälardalen University, Västerås - Sweden
†University of L'Aquila - Italy
‡Vrije Universiteit Amsterdam - The Netherlands
§Chalmers University of Technology | University of Gothenburg - Sweden
¶ KTH Royal Institute of Technology - Sweden

***Abstract*—The production of software for robotic systems is often case-specific, without fully following established engineering approaches. Systematic approaches, methods, models, and tools are pivotal for the creation of robotic systems for real-world applications and turn-key solutions. Well-defined (software) engineering approaches are considered the "make or break" factor in the development of complex robotic systems. The shift towards well-defined engineering approaches will stimulate component supply-chains and significantly reshape the robotics marketplace.**

The goal of this technical briefing is to provide an overview on the state of the art and practice concerning solutions and open challenges in the engineering of software required to develop and manage robotic systems. Model-Driven Engineering (MDE) is discussed as a promising technology to raise the level of abstraction, promote reuse, facilitate integration, boost automation and promote early analysis in such a complex domain.

I. Description of the topic

Robots have been part of our daily life for several decades, such as in manufacturing automation to cut and assemble parts, or in medical field to perform extremely delicate surgeries. Most recently, autonomous robots have shown a great promise for future, from self-driving vehicles in transportation, to robotic vacuum cleaners in households, to teams of mobile robots in autonomous warehouse solutions. However, in the production of software for robotic systems "*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.*" as stated by the H2020 Multi-Annual Robotics Roadmap ICT-2016 [7].

The use of ad-hoc development processes in general, and software engineering approaches in particular, hampers reuse and complicates the configurability of existing solutions. This justifies the need of systematic approaches, methods, and tools to (i) easily configure robots, or provide them with self-configuration capabilities, (ii) specify robotic tasks in an easy and user-friendly way, and (iii) instill intelligence into robots, i.e. to make them able to take decisions on their own to manage unpredictable situations. This shifts towards well-defined engineering approaches will stimulate component supply-chains and significantly impact the robotics marketplace.

In this technical briefing we focus on autonomous mobile multi-robot systems (MMRSs) that are represented by a set of robots operating as a team in a shared, potentially unknown, and unpredictable environments. These MMRSs are expected to be collaborative, effective, efficient, adaptive, and reactive. Unlike multi-agent systems in general, MMRSs are characterized by their physical properties, such as control and communication constraints. When it comes to the engineering of software specific for MMRSs, it is crucial to consider that this kind of software is typically embedded, concurrent, real-time, distributed, data-intensive and must guarantee system properties such as safety, reliability, and fault tolerance [3]. The raised interests around the topic of software and system engineering for MMRSs is testified by the existence of a significant corpus of research work dealing with specific aspects of MMRSs. For example, in [6] authors present an architecture-centric approach for building self-adapting and self-managing robotic systems. For the purpose of raising the level of abstraction in developing MMRSs, the adoption of component-based software engineering (CBSE) tailored to robotics has been advocated in [3]. In [9] authors present an approach for defining missions for swarms of drones via a domain-specific modeling language and for generating low-level instructions for each drone; the implementation of the approach is available as an open-source project [1]. Finally, the need for unification of software development for robots is demonstrated by the recent emergence of Robot Operating System (ROS) [8], that is nowadays considered to be the standard platform among the robotics community.

II. Software engineering for Robotic Systems

When bringing intelligence to robots, the unescapable need to provide specific capabilities, such as the human-like sense-process-(re)act chain, arose. At that point, software came into the picture and progressively gained importance eventually becoming one of the core aspects in robotics development and maintenance. This is the reason why an effective interplay of software engineering and conventional robotics is essential.

Several initiatives related to software engineering for robotics have been launched over the last years, such as: the IEEE Technical Committee on Software Engineering for Robotics and Automation, and the Journal of Software Engineering for Robotics. In the specific case of MMRSs, innovative software engineering approaches and methodologies able to support the definition, the development, and the realization of collective missions are needed. Several attempts have been documented in this direction [2]. Unluckily, the excessive

focus on performance issues led robotic engineering to neglect crucial quality attributes of software-intensive systems, such as reusability, flexibility, and interoperability.

This technical briefing shows why and how adopting and adapting mainstream software engineering methodologies and technologies towards a systematic, disciplined, and quantifiable approach throughout the system life-cycle can help in solving a number of challenges for MMRSs. In particular, this technical briefing focusses on the following challenges: (i) *Platform neutrality* for tractable cross-platform development; (ii) *Systematic reusability* that avoids isolated solutions which cannot be easily reused nor combined; (iii) *Orchestrating concurrency* managing simultaneous execution of multiple tasks performed by independent entities; (iv) *Context awareness* for meeting evolving environmental characteristics and constraints during a mission execution; (v) *Coping with uncertainty* that maintains desired degree of security, performance, and dependability during and after adaptation; (vi) *Dynamic discoverability of available resources* for automatic adjustment of mission plans; (vii) *Subsystems interoperability* for coordinating heterogeneous subsystems of an MMRS;(viii) *Human-robot synergy* for having proper means of interaction between humans and robots.

Dealing with these challenges is a very daunting task. In fact it demands a set of assorted abilities for: managing abstractions in MMRSs specifications, providing various degrees of automation in MMRSs software development, and offering means to enable analysis related to different concerns of MMRSs missions at design time as well as at runtime.

Abstraction, automation, and analysis are the idiosyncratic characteristics of MDE, which advocates the systematic use of models as first class actors in the software life-cycle. Models are meant to represent abstractions of real systems that are analysed and engineered through a coherent set of interconnected concepts precisely captured in metamodels.

During the technical briefing we motivate our vision that MDE shows the traits needed for successfully supporting development and maintenance of MMRSs, especially regarding software. This is also the belief of the MARR [7] that clearly states: "*Model-driven software development (MDSD) and DSL (domain specific languages) are core technologies required in order to achieve a separation of roles in the robotics domain while also improving compose-ability, system integration and also addressing non-functional properties*". Similarly, the TC-SOFT has identified MDE (together with CBSE) as a key methodology for the future of robotic software engineering, and we identified MDE as a potential enabler for civilian missions of MMRSs [5] and mission-critical Internet-of-Things systems [4]. In fact, MDE provides the possibility to systematically and concurrently focus on different levels of abstraction at which the involved developers can operate for (i) improving the quality of MMRSs in terms of, e.g., safety, reliability and reusability, (ii) reducing the inherent variability and complexity of MMRSs, and (iii) promoting the reuse of software and hardware components across MMRSs.

III. Organizers

Federico Ciccozzi is Assistant Professor at Mälardalen University, Sweden. His research covers many aspects of MDE and CBSE, with focus on the embedded real-time domain. He is involved in national and European projects, and leading a national project on cross-domain MDE for embedded multicore systems with ABB Corporate Research, Alten Sweden and Ericsson AB. http://www.es.mdh.se/staff/266-Federico_Ciccozzi

Davide Di Ruscio is Assistant Professor at the University of L'Aquila. His research interests are related to several aspects of MDE. He is working on various European and Italian research projects and he is applying MDE concepts and tools in various application domains (e.g., service-based, autonomous, or open source software systems). http://www.di.univaq.it/diruscio/

Ivano Malavolta is Assistant Professor at the Vrije Universiteit Amsterdam, Department of Computer Science. His research focuses on empirical software engineering, software architecture, model-driven engineering, and mobile multi-robot systems. http://www.ivanomalavolta.com

Patrizio Pelliccione is Associate Professor at the University of Gothenburg | Chalmers University of Technology. His research topics are mainly in software engineering for robotics, software architecture, MDE, and formal methods. In his research activity Patrizio collaborated with several industries especially in Sweden and Italy. http://www.patriziopelliccione.com/

Jana Tumova is Assistant Professor at KTH Royal Institute of Tecnhology. Her research interests include formal methods, control theory and robotics in general, and temporal logic-based control of cyber-physical systems, and autonomous robots in particular. https://people.kth.se/~tumova

IV. acknowledgements

Research partly supported from the EU H2020 Research and Innovation Programme under GA No. 731869 (Co4Robots).

References

[1] Darko Bozhinoski, Davide Di Ruscio, Ivano Malavolta, Patrizio Pelliccione, and Massimo Tivoli. Flyaq: Enabling non-expert users to specify and generate missions of autonomous multicopters. In *International Conf. on Automated Software Engineering (ASE)*, pages 801–806. IEEE, 2015.

[2] D. Brugali and E. Prassler. Software engineering for robotics. *Robotics Automation Magazine, IEEE*, 16(1):9–15, March 2009.

[3] D. Brugali and P. Scandurra. Component-based robotic engineering (part I) [tutorial]. *IEEE Robot. Automat. Mag.*, 16(4):84–96, 2009.

[4] Federico Ciccozzi, Ivica Crnkovic, Davide Di Ruscio, Ivano Malavolta, Patrizio Pelliccione, and Romina Spalazzese. Model-Driven Engineering for Mission-Critical IoT Systems. *IEEE Software*, 34(1):46–53, 2017.

[5] Federico Ciccozzi, Davide Di Ruscio, Ivano Malavolta, and Patrizio Pelliccione. Adopting MDE for Specifying and Executing Civilian Missions of Mobile Multi-Robot Systems. *IEEE Access*, 2016.

[6] G. Edwards, J. Garcia, H. Tajalli, D. Popescu, N. Medvidovic, G. Sukhatme, and B. Petrus. Architecture-driven self-adaptation and self-management in robotics systems. *in Proceedings of SEAMS*, 2009.

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

[8] M. Quigley, J. Faust, T. Foote, and J.s Leibs. Ros: an open-source robot operating system. In *Proceedings of ICRA*, 2009.

[9] Davide Di Ruscio, Ivano Malavolta, Patrizio Pelliccione, and Massimo Tivoli. Automatic generation of detailed flight plans from high-level mission descriptions. In *ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, pages 45–55. ACM, 2016.