The Journal of Systems and Software 122 (2016) 16–39

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Software architectures for robotic systems: A systematic mapping study

Aakash Ahmad[a], Muhammad Ali Babar[a,b]

[a] CREST - Centre for Research on Engineering Software Technologies, Software and Systems Section, IT University of Copenhagen, Denmark
[b] CREST - Centre for Research on Engineering Software Technologies, School of Computer Science, University of Adelaide, Australia

ARTICLE INFO

Article history:
Received 4 May 2015
Revised 5 August 2016
Accepted 8 August 2016
Available online 21 August 2016

Keywords:
Software architecture
Robotic systems
Evidence-based software engineering
Software architecture for robotics
Systematic mapping study

ABSTRACT

Context: Several research efforts have been targeted to support architecture centric development and evolution of software for robotic systems for the last two decades.
Objective: We aimed to systematically *identify* and *classify* the existing solutions, research progress and directions that influence architecture-driven modeling, development and evolution of robotic software.
Research Method: We have used Systematic Mapping Study (SMS) method for identifying and analyzing 56 peer-reviewed papers. Our review has (i) taxonomically classified the existing research and (ii) systematically mapped the solutions, frameworks, notations and evaluation methods to highlight the role of software architecture in robotic systems.
Results and Conclusions: We have identified eight themes that support architectural solutions to enable (i) *operations*, (ii) *evolution* and (iii) *development* specific activities of robotic software. The research in this area has progressed from *object-oriented* to *component-based* and now to *service-driven* robotics representing different architectural models that emerged overtime. An emerging solution is *cloud robotics* that exploits the foundations of service-driven architectures to support an interconnected web of robots. The results of this SMS facilitate knowledge transfer – benefiting researchers and practitioners – focused on exploiting software architecture to model, develop and evolve robotic systems.

1. Introduction

Robotic systems are increasingly being integrated in various aspects of everyday life. The robotic applications range from mission critical (Parker, 1998) to infotainment and home service tasks (Mäenpää et al., 2004, Kwak et al., 2006). Robotic systems are expected to assist or replace their human counterparts for efficient and effective performance of all sorts of tasks such as industrial operations (Angerer et al., 2009) or surgical procedures (Kazanzides et al., 1992, Buzurovic et al., 2010). A robotic system is a combination of various components – hardware for system assembling and software for system operations – that must be seamlessly integrated to enable a robotic system's function as expected. To support the vision of a robotic-driven world[1], academic research (Parker, 1998, Hu et al., 2012, Brugali et al., 2007), industrial (Jackson and Coll, 2008, Katz and Some, 2003) and open source solutions (Garage, 2011, Bonarini et al., 2014) are striving to provide cost-effective and efficient solutions for the development, evolution and operations of robotics systems. Researchers (Brugali et al., 2007) and practitioners (Jackson and Coll, 2008) are increasingly focusing on exploiting software engineering methodologies to abstract complexities and enhance efficiency for modeling, developing, maintaining and evolving robotic systems cost-effectively (Oliveira et al., 2013, Zhi et al., 2013, Elkady and Sobh, 2012).

Software Architecture (SA) represents the global view of software systems by abstracting out the complexities of low-level design and implementation details (Wohlin, 2014). SA plays a vital role in ensuring the fulfillment of functional and non-functional requirements (Schmerl and Garlan, 2004). It is considered that architecture-centric software development helps increase quality, modularity and reusability (Brugali et al., 2007) through the use of patterns (Gomaa, 2000) and decrease complexity by applying model-driven development (Boyatzis, 1998). Researchers from different communities (such as robotics, software engineering, industrial engineering, and artificial intelligence) have exploited architectural models to design, reason about, and engineer robotic software. Architecture-centric robotics research and practice can be

E-mail addresses: aahm@itu.dk (A. Ahmad), ali.babar@adelaide.edu.au (M.A. Babar).

[1] EUROP, the European Robotics Technology Platform is an industry-driven platform compromising the main stakeholders in robotics. EUROP aims at enabling research and practices through *Robotic Visions to 2020 and Beyond - The Strategic Research Agenda for Robotics in Europe.*

http://dx.doi.org/10.1016/j.jss.2016.08.039
0164-1212/

characterized by various architectural models that emerged overtime such as: (i) object-oriented robotics (OO-R) enabling modularity (Miller and Lennox, 1991), (ii) component-based robotics (CB-R) supporting reusability (Jung et al., 2010), and (iii) service-driven robotics (SD-R) exploiting dynamic composition of software (Cepeda et al., 2011).

Since the early 90s, there has been a continuous stream of reported research on software architectures for robotic systems. It is a timely effort to analyze the collective impact of existing research on architectural solutions for robotic software. We decided to conduct a mapping study by following the guidelines reported in (Petersen et al., 2008) to investigate the state-of-the-art that promotes architectural solutions to model, develop and evolve robotic software. The objective of this mapping study is to: '*systematically identify, analyze, and classify the software architectural solutions for robotic systems and provide a mapping of these solutions to highlight their potential, limitations along with emerging and future dimensions of research*'. This study was motivated by a number of research questions, whose answers are expected to disseminate a systematized knowledge among researchers and practitioners who are interested in the role of software architecture for robotic systems. The key contributions of this study are:

- Systematic selection and analysis of the collective impact of the existing research on software architecture related aspects of robotic systems for identifying (i) predominant *research themes*, (ii) architectural *solutions* for each of the themes, (iii) *framework* support along with (iii) *modeling notations, evaluation methodologies* and *application domain* of architectural solutions.
- Reflections on the (i) *progression* and maturation of research overtime along with (ii) existing and emerging software architectural solutions for robotic systems.

The results of this study suggest that ***architectural solutions*** support (i) *operations* enabling information and resource sharing (ii) *evolution* with runtime adaptation and design-time re-engineering, along with (iii) development such as modeling, designing and programming of robotic software. A number of ***architectural frameworks*** - open source, academic and industrial solutions - are available for development; UML or its derivative notations are predominantly used for ***architectural modeling***. A majority of the architectural solutions support home service, mission critical and navigation robots. Architectural solutions in general have been evaluated using controlled experiments and simulation based techniques. However, there is a lack of reporting on ***architecture specific evaluations*** against functional and quality requirements (Orebäck and Christensen, 2003). An incremental progress of research from OO-R (Miller and Lennox, 1991) to CB-R (Jung et al., 2010) and more recently SD-R (Cepeda et al., 2011) (***architectural generations***) have resulted in an emergence of cloud robotics (Hu et al., 2012); while model-driven robotics is also gaining momentum (Boyatzis, 1998). The results of this SMS benefit:

- Researchers who are interested in knowing the state-of-the-art of software architecture for robotics systems. The systematic classification of the existing research provides a body of knowledge for deriving new hypotheses to be tested and identifying the areas of future research.
- Practitioners who may be interested in understanding the reported solutions in terms of frameworks, modeling notations, tools and validation techniques for architectural development and evolution of robotic systems.

The rest of this paper is organized as follows. Section 2 briefly introduces robotic systems, software architecture and related studies. Section 3 describes the research methodology used. Section 4 reports the years, classification and mapping of the research. Based on the classification, various architectural solutions and frameworks for robotic software are presented in Section 5. Modeling notations, validation techniques and application domains that complement architectural solutions are presented in Section 6. We discuss the progression, future dimensions, and emergence of various architectural models in Section 7. Validity threats are presented in Section 8. Section 9 presents key conclusions from this study.

2. Background and related studies

In this section, we briefly introduce robotic systems (in Section 2.1) and software architecture (in Section 2.2). We also discuss some existing secondary studies (in Section 2.3) that are related to our SMS.

2.1. An overview of robotic software systems

A robotic system is a combination of hardware and software components as two distinct layers that can be integrated to build a robot (Jackson and Coll, 2008, Yool et al., 2006). The ISO 8373:2012 standard provides a vocabulary of robots and various robotic devices that operate in industrial and non-industrial environments. The hardware components such as the sensors, robotic arms, navigation panels enable the assembly of a robot. Hardware components are controlled and manipulated by Control Layer that is essentially a collection of drivers (as system specific code) to interact with the hardware as in Fig. 1. For more complex functions of a robot, specialized software is provided for integration and coordination of hardware components to manipulate the robotic behavior. In Fig. 1, this refers to as Application Layer that utilizes the control layer to support robotic operations. For example, considering a home service robot (Schofield, 1999), the control layer provides a driver that enables access to a robotic arm. Depending on specific requirements, a software system at *application layer* must be provided. Such software system is expected to utilize drivers from *control layer* to enable the movement of arm for home service robot at certain degrees of precision and/or avoiding any obstacles.

Fig. 1 highlights that from a software perspective robotic systems are an example of a layered design, where each layer encapsulates a specific functionality and depends on the layer(s) above or below to complete a system's functionality. The focus of this study is to review the research state-of-the-art for architectural solutions supporting robotic software – *application layer*.

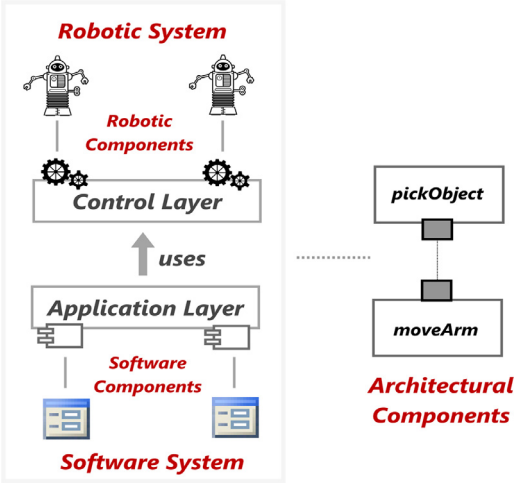

Fig. 1. A reference model for robotic software systems.