Managing Conflicting Requirements in Systems of Systems

Conference or Workshop Item

How to cite:


For guidance on citations see FAQs.

© [not recorded]

Version: Version of Record

Copyright and Moral Rights for the articles on this site are retained by the individual authors and/or other copyright owners. For more information on Open Research Online’s data policy on reuse of materials please consult the policies page.

oro.open.ac.uk
Managing Conflicting Requirements in Systems of Systems

Thiago Affonso de Melo Novaes Viana
thiago.viana@open.ac.uk

School of Computing and Communications/ Faculty of Science, Technology, Engineering and Mathematics

Supervisors name/s: Prof. Andrea Zisman and Dr. Arosha Bandara
Starting date: 01/02/2016 (Full-Time)

I. INTRODUCTION

Software systems have evolved from being stand-alone systems to being composed into Systems of Systems. A System of Systems (SoS) is defined as an arrangement of independently created, discovered, and selected systems, which are integrated into a single system in order to deliver unique capabilities [1]. In this context, each participating system can operate and support different goals in its own environment (viz. local goals), as well as support new goals of the SoS as a whole (viz. global goals), that could not be achieved separately by the participating systems. An SoS presents many features including operational and managerial independence, geographic distribution of participating systems, and emergent behaviors [2]. Examples of SoSs are found in several different domains including, but not limited to, transport network systems, household energy management systems, personal nutrition management systems, smart homes, smart cities and intelligent healthcare systems.

However, as the SoS is formed by the integration of independent complex systems, this will increase the complexity of the SoS to at least one more order than its component systems [3]. This means that problems in the SoS environment are harder to handle than in the component system environment. Therefore, this will bring a number of software engineering challenges regarding their specification, design, construction, and operation. Among these, one important challenge is concerned with managing emerging conflicting behavior that leads to inconsistency. In an SoS, the various participating systems are often from different domains; are developed by different teams of people under different circumstances and time; have distinct functionalities; and are used by different stakeholders. All of these factors contribute to the existence of inconsistent and conflicting requirements.

This research presents an overview of a framework called MaCoRe_SoS (Managing Conflicting Requirements in Systems of Systems) that supports conflict management in SoSs. The management of conflicting requirements in the proposed framework involves three main steps, namely (a) conflict identification, (b) conflict diagnosis, and (c) resolution based on the use of a utility function. In order to support the main steps, the framework uses a Monitor-Analyze-Plan-Execute-Knowledge (MAPE-K) architectural pattern[4].

II. MOTIVATING EXAMPLE

In order to illustrate and evaluate the proposed framework, this research uses FeedMe FeedMe [5], an exemplar of an SoS scenario composed of different systems to address food security problems at different levels of granularity, namely individuals, groups, cities and nations. At the individual level (viz. AnalyseMe), FeedMe FeedMe presents smart devices to monitor, analyse and provide suggestions about the nutritional and health status of a person. At the group level (viz. HomeHub), FeedMe FeedMe uses smart home appliances that interoperate to create a more precise family meal plan, based on the family resources and budget. At the city level (viz. SmartCity), local markets collect data from multiples families to manage their stock and to reduce food wastage. At the national level (viz. SmartNation), food producers and manufacturers collect data from different markets to forecast food needs and provide alternatives in case of food crisis. Figure 1 shows an overview of the FeedMe FeedMe SoS with its various participating systems and devices.

III. THE MACORE_SoS FRAMEWORK

The main goal of the MaCoRe_SoS framework is to manage conflicting requirements in SoSs. In the framework, the requirements represent both local goals of the participating components and global goals of the SoS environment as a whole. We distinguish these as local and global requirements.

Figure 2 shows an overview of the MaCoRe_SoS framework. As shown in the figure, the framework uses a conflict manager component to supports requirements conflict management based on three main steps, as described in the
The growth in the complexity and heterogeneity of modern software systems has led to systems that compose themselves into bigger systems to achieve more sophisticated functionalities. These systems are often System of Systems (SoS) where the management of emerging conflicting behaviors, expressed as requirements is a challenge. As a new and emergent application, the management of inconsistencies is an important task inside the SoS environment.

Therefore, as different systems are composed together, emergent and undesirable behaviors arise, leading to conflicting requirement. To address this problem, we present the MaCoRe_SoS framework, with three steps: (a) conflict identification, (b) conflict diagnosis, and (c) resolution.

We have built a prototype version of the framework and demonstrated its efficacy for scenarios based on FeedMe FeedMe, an example SoS ecosystem designed to support food security at different levels of granularity (individuals, families, cities, and nations). The initial results demonstrate that it is possible to identify and manage conflicts and that the application of a resolution method is able to support the SoS to satisfying local and global requirements.

**References**


