Towards a Framework for Managing Inconsistencies in Systems of Systems

Conference or Workshop Item

Towards a Framework for Managing Inconsistencies in Systems of Systems

Thiago Viana, Arosha K. Bandara, Andrea Zisman
School of Computing and Communications
The Open University, Walton Hall, Milton Keynes MK7 6AA, UK
{firstname.lastname@open.ac.uk}

ABSTRACT

A System of Systems (SoS) is an arrangement of useful and independent complex systems integrated into a bigger system in order to deliver unique capabilities [1]. In this context, each independent system should be able to fulfill useful objectives in their own environments, but when they are composed in a SoS they should be able to achieve new objectives that could not be achieved by the individual systems. Examples of SoSs are found in several applications including, but not limited to, transport network systems, household energy management systems, personal nutritional systems, smart homes, smart cities, and intelligent healthcare systems.

An SoS presents many challenges such as operational and managerial independence, geographic distribution of participating systems, and emergent behaviors [2]. In a 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. Therefore, several software engineering challenges are presented during the development, deployment, and use of SoSs. An important challenge is concerned with managing requirements inconsistencies in SoSs. More specifically, in an SoS, the various independent complex systems may have conflicting behavior among themselves, as well as, emerging conflicting behaviours between the SoS as a whole and the participating systems, generating inconsistent requirements.

In this paper, we articulate the challenges of inconsistency management in SoSs, from the specification of requirements that can be monitored at run-time, to the detection, diagnosis and resolution of requirements inconsistencies. We also present an overview of our framework, called *MaCoRe_SoS,* to support inconsistency management in SoSs. The framework assumes the requirements of the participating systems and of the SoSs as a whole expressed in an extension of the RELAX language [4]. The management of inconsistent requirements is based on four steps, as described in the survey from Spanoudakis and Zisman [5], namely (a) overlap detection, (b) conflict identification, (c) conflict diagnosis, and (d) resolution. The conflict identification, diagnosis and resolution steps are executed based on a Monitor-Analyze-Plan-Execute-Knowledge (MAPE-K) architectural pattern [6]. The overlap detection is performed using ontologies that merge the semantic models of the individual participating systems. The identification of conflicts is assisted by an event monitor component that detects violations of requirements expressed in the RELAX language extension. The diagnosis of the conflicts is performed by an analyzer component using requirements interaction features such as Basis, Degree, Direction, and Likelihood, as suggested in [7]. The resolution of conflicts is based on the use of a utility function and supports eight resolution methods, namely relaxation, refinement, abandonment, compromise, restructuring, reinforcement, re-planning, and postponement.

An initial prototype tool has been implemented in order to demonstrate and evaluate the framework. This initial implementation focuses on managing conflicting requirements associated with the utilization of resources by participating systems in an SoS. The different types of resources depend on the domains of the participating systems and the SoS as a whole. In order to illustrate and evaluate the work we use an SoS ecosystem designed to support food security at different levels of granularity: individuals, families, cities, and nations. We use this example scenario to illustrate the outstanding challenges of inconsistency management in systems of systems that we propose to address in our future research.

CCS Concepts

• Software system structures→Software architectures
• Software functional properties→Correctness→Consistency.

Keywords

Systems of systems; Inconsistency management; Adaptation

REFERENCES

[1] Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering, *Systems Engineering Guide for Systems of Systems*, vol. 1. Washington, DC: 2008.

[2] S. W. Popper, S. C. Bankes, R. Callaway, and D. DeLaurentis, 'System of systems symposium: Report on a summer conversation', *Proc. 1st Syst. Syst. Symp.*, 2004.

[3] A. P. Sage and C. D. Cuppan, 'On the systems engineering and management of systems of systems and federations of systems', *Inf. Knowl. Syst. Manag.*, vol. 2, no. 4, pp. 325–345, 2001.

[4] J. Whittle, P. Sawyer, N. Bencomo, B. H. Cheng, and J.-M. Bruel, 'RELAX: a language to address uncertainty in self-adaptive systems requirement', *Requir. Eng.*, vol. 15, no. 2, pp. 177–196, 2010.

[5] G. Spanoudakis and A. Zisman, 'Inconsistency management in software engineering: Survey and open research issues', *Handb. Softw. Eng. Knowl. Eng.*, vol. 1, pp. 329–380, 2001.

[6] J. O. Kephart and D. M. Chess, 'The vision of autonomic computing', *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[7] W. N. Robinson, S. D. Pawlowski, and V. Volkov, 'Requirements interaction management', *ACM Comput. Surv. CSUR*, vol. 35, no. 2, pp. 132–190, 2003.