Managing Conflicting Requirements in Systems of Systems by Adaptation

Conference or Workshop Item

Managing Conflicting Requirements in Systems of Systems by Adaptation

Thiago Affonso de Melo Novaes Viana
thiago.viana@open.ac.uk
Department of Computing and Communications/ Faculty of Mathematics, Computing and Technology

Supervisors name/s: Prof. Andrea Zisman and Dr. Arosha Bandara
Status: Full-Time
Probation viva: Before
Starting date: 01/02/2016

I. INTRODUCTION

A System of Systems (SoS) is an arrangement of useful and independent complex systems, which are integrated into a bigger system that delivers unique capabilities [1]. In this context, it is important to notice that each independent system should be capable of operate and fulfil useful purposes on its own environment, but, when these independent systems are together and become a SoS they are able to achieve new objectives that could not be achieved by the individual systems. This means that, while at the same time achieving key quality objectives such as performance, reliability or security, a SoS can offer more functionality than the sum of the constituent systems [2].

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. A common problem that is present into all types of systems is the conflicting requirements. Due to voluminous requirements documents, constant changes into requirements, complexity and presence of different stakeholders, conflicts arise within each component system and also across the SoS due to unexpected interactions between components, users and SoS goals [4].

In the SoS environment, there are different component systems that often came from different domains, developed by different teams, under different circumstance and time. Also, each component system evolves into a different rhythm. Therefore, the SoS environment is even more likely to present and be affected by conflicting requirements. Actually, there are approaches to handle with conflicting requirements, however, no one of them takes into account the differences present in the SoS environment: the independence of each component system, the increased order of complexity, the distributed nature and the dynamic and flexible boundaries. On the other hand, adaptation is way to support a system and help it to modify their behaviour and structure in response to their perception of the environment [5]. Moreover, the SoS is known to be adaptive by nature [6]. Thus, this research aims to propose a way to support the SoS in its task to manage conflicting requirements by adaptation.

II. CONFLICTING REQUIREMENTS

Conflicting requirements isn't a new area of study. Conflicting requirements are the requirements held by two or more stakeholders that provoke an inconsistency [4]. Also, [7] define inconsistency as "any situation in which two parts of a specification do not obey some relationship that should hold between them". So, conflicting requirements brings inconsistency, which means that conflicts reflect into problems in the relationship of two or more parts in the system. In the SoS context, there are three main parts that may influence the appearance of conflicting requirements: each component system, the SoS and the users preferences.

Thus, as each component system exists into an independent context so they will present their own requirements and these requirements may be conflicting with some of the SoS global requirements, moreover all them could be conflicting with the user's preference as well. In fact, conflicting requirements will exists and this can't be avoided, the way to handle with them is by managing the inconsistency that triggers them [7]. In this context, there are many examples of tackling the problem of conflicting requirements [8] [9] [10]. However, all these approaches are based on design-time considerations. But, knowing that each component system was designed at different times and under different circumstances. Also, that the interaction between them will exist at runtime and that the conflicts will arise by the evolution of them. Then, these approaches can't be directly applied to manage the conflicting requirements problem in the SoS environment, because this is a runtime problem. Thus, there are some examples of tackling the problem of managing conflicting requirements at runtime [11] [12] [13] [14]. But, even considering conflicting requirements at runtime, all these approaches don't take the SoS environment in consideration.

There are many different and important issues to be taken into account and that is able to bring challenges to the process of manage them. It is important to take into account that: 1. Each component system is independent and the SoS has a limited control over them; 2. There are different levels of requirements (the component systems level and the SoS level); 3. The SoS is a complex distributed system and 4. The boundaries aren't fixed. Thus, by taking account the SoS environment, [15] presents ReMinds, a flexible framework that is able to monitor events in the SoS at runtime. Further, they applied ReMinds at an industrial SoS and presented a promising result when working with realistic event loads [16]. However, this approach is only able to monitor the SoS and is

focused just at unexpected events, it doesn't take account how to handle with conflicting requirements and all the implications presents in the inconsistency management. Therefore, conflicting requirements are harder to manage in the SoS environment. Also, the existing approaches can't be directly applied without reasoning about this environment and its particularities.

III. Adaptation in the SoS environment

The main features of a SoS are autonomy, connectivity between the systems and emergent behaviours [17]. All these features are very important to a SoS, moreover, they can be considered as basic features to the adaptation process. In fact, adaptation is present and necessary to the SoS environment [6]. So, in other words, the SoS is adaptive by nature [6]. As adaptation is a natural mechanism to the SoS handle its problems, it might be a way to support the problem of managing conflicting requirements. Indeed, [18] argue about the importance of the self-adaptive system be aware of its own requirements at runtime, in order to reason if they are being attended and, if necessary, manage conflicts between them. This means that the adaptation process is not just able to receive enough information about the conflicting requirements, but also, the adaptation process is into a perfect position to manage them at runtime. Also, [19] uses adaptation to propose a component to manage some conflicts between Java classes that arise from integration and evolution. Also, [20] introduce MobiWeb and argue that it is able to manage conflicting requirements by using a priority scheme in the adaptation process. Finally, [21] proposes an extension to the KAOS method by incorporating "adaptive goals", in order to represent adaptation strategies to tackle conflicting goals.

IV. Conclusion and research proposal

In fact, Conflicting requirements exists in all kinds of systems and they are an important problem to manage. In the SoS context, these problems are more acute. This happens because the SoS is an arrangement of complex and independent component systems, so, the differences between each component system, as well as the difference between their goals and stakeholders increase the frequency and impact of conflicting requirements. Actually, the approaches to manage conflicting requirements at runtime don't take into account all the issues and differences present in the SoS environment: the independence of each component system and the limited control to them by the SoS; The complexity of each component system and the powerful sum of them; The distributed nature of the SoS and its components; The dynamic and flexible boundaries; The many dimensions where the conflicts may happen. On the other hand, adaptation is a natural issue to the SoS context. Furthermore, the adaptation process operates at runtime and is able not just to monitor the requirements, but it may also be able to reason about them, identify conflicts and propose actions in the SoS environment, in order to support the management of conflicting requirements. Thereby, as actually there is no approach able to support the SoS in its tasks to manage the conflicting requirements, and as adaptation is a natural mechanism to the SoS, so, this research aims to the following question: How to support the management of conflicting requirements in the SoS environment by using 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: ODUSD(A&T)SSE, 2008.

[2] S. W. Popper, S. C. Bankes, R. Callaway, and D. De-Laurentis, '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] W. N. Robinson and S. D. Pawlowski, 'Managing requirements inconsistency with development goal monitors', *Softw. Eng. IEEE Trans. On*, vol. 25, no. 6, pp. 816–835, 1999.

[5] R. De Lemos, H. Giese, H. A. Müller, M. Shaw, J. Andersson, M. Litoiu, B. Schmerl, G. Tamura, N. M. Villegas, T. Vogel, and others, *Software engineering for self-adaptive systems: A second research roadmap*. Springer, 2013.

[6] M. P. Romay, C. E. Cuesta, and L. Fernández-Sanz, 'On self-adaptation in systems-of-systems', in *Proceedings of the First International Workshop on Software Engineering for Systems-of-Systems*, 2013.

[7] S. Easterbrrok and B. Nuseibeh, 'Using viewpoints for inconsistency management', *Softw. Eng. J.*, vol. 11, no. 1, pp. 31–43, 1996.

[8] B. Boehm and H. In, 'Identifying quality-requirement conflicts', *IEEE Softw.*, vol. 13, no. 2, p. 25, 1996.

[9] A. Kozlenkov and A. Zisman, 'Discovering, recording, and handling inconsistencies in software specifications', *Int. J. Comput. Inf. Sci.*, vol. 5, no. 2, pp. 89–108, 2004.

[10] B. Nuseibeh, S. Easterbrook, and A. Russo, 'Leveraging inconsistency in software development', *Computer*, vol. 33, no. 4, pp. 24–29, 2000.

[11] N. Bencomo, J. Whittle, P. Sawyer, A. Finkelstein, and E. Letier, 'Requirements Reflection: Requirements As Runtime Entities', in *Proceedings of the 32Nd ACM/IEEE International Conference on Software Engineering - Volume 2*, New York, NY, USA, 2010.

[12] M. S. Feather, S. Fickas, A. Van Lamsweerde, and C. Ponsard, 'Reconciling System Requirements and Runtime Behavior', in *Proceedings of the 9th International Workshop on Software Specification and Design*, Washington, DC, USA, 1998, p. 50–.

[13] L. Baresi, L. Pasquale, and P. Spoletini, 'Fuzzy Goals for Requirements-Driven Adaptation', in *2010 18th IEEE International Requirements Engineering Conference*, 2010, pp. 125–134.

[14] F. Kneer and E. Kamsties, 'Model-based Generation of a Requirements Monitor.', in *REFSQ Workshops*, 2015, pp. 156–170.

[15] M. Vierhauser, R. Rabiser, P. Grunbacher, and J. Thanhofer-Pilisch, 'The ReMinds Tool Suite for Runtime Monitoring of Systems of Systems', in *Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on*, 2015, pp. 777–782.

[16] M. Vierhauser, R. Rabiser, P. Grünbacher, K. Seyerlehner, S. Wallner, and H. Zeisel, 'ReMinds: A flexible runtime monitoring framework for systems of systems', *J. Syst. Softw.*, vol. 112, pp. 123–136, 2016.

[17] M. W. Maier, 'Architecting principles for systems-of-systems', in *INCOSE International Symposium*, 1996, vol. 6, pp. 565–573.

[18] P. Sawyer, N. Bencomo, J. Whittle, E. Letier, and A. Finkelstein, 'Requirements-aware systems: A research agenda for re for self-adaptive systems', in *Requirements Engineering Conference (RE), 2010 18th IEEE International*, 2010, pp. 95–103.

[19] R. Keller and U. Hölzle, 'Supporting the integration and evolution of components through binary component adaptation', *Univ. Calif. St. Barbara Tech. Rep. TRCS97*, vol. 15, 1997.

[20] M. Margaritidis and G. C. Polyzos, 'Application-assisted adaptation of real-time streams over wireless links', in *2nd Annual UCSD Conference on Wireless Communications*, 1999.

[21] L. Baresi and L. Pasquale, 'Live goals for adaptive service compositions', in *Proceedings of the 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems*, 2010.