Architectural reconfiguration of interacting services

(PhD Proposal)

Supervisor: *Luis Soares Barbosa*, Universidade do Minho
www.di.uminho.pt/~lsb
Research Group: HASLab - High Assurance Software Laboratory
wiki.di.uminho.pt/twiki/bin/view/DI/FMHAS/

1 The problem

Consider a service-based application for mobile devices that synchronizes video and audio streams coming from various sources and redistributes to several clients following different consumer protocols.

- How to predict, compositionally from its component services and the underlying architectural glue, key (qualitative and quantitative) properties characterizing its global behaviour and evolution in an open, changeable environment?
- How to guarantee satisfaction of such behavioural properties at runtime?
- How to ensure the application meets the performance requirements specified in service-level agreements with its clients?

2 The theme

Service-based software systems [8] are built by plugging services together which interact by exchanging data, performing computation, and modifying their environment. Services are dynamic entities, running on different platforms often owned by different organisations, interacting through public interfaces, and typically remaining loosely coupled, if not utterly unaware of each other.

Designing such systems right is very difficult, because their complexity is beyond the current practical reach of formal methods. In particular, existing models for service behavior specification lack means for reasoning about real-time and performance characteristics/requirements of interacting services. Such inadequacy is explained by systems complexity (which entails the need for flexible compositional approaches), inappropriate simplifications (e.g., the assumption that QoS parameters are independent of each other) and the inability to deal with insufficient data about service behavior and/or QoS characteristics.

> The questions above are about predicting properties and ensuring they are maintained, within certain limits of invariance, along the system's life. Change being the norm rather than the exception, architectural dynamic reconfiguration, i.e., the ability of a system to adjust itself at runtime in response to perception of context, to maintain behavioural and QoS invariants, is a major, open research issue, which frames this PhD project.

3 Objectives

This PhD project aims at developing

> new and effective formal techniques for modeling, analyzing and reasoning about architectural reconfiguration of (service-based) time/performance critical systems, encompassing monitoring, reassembling and redeployment at runtime.

This entails both:

- The development of a QoS-aware architectural model and an associated modeling language.
- The semantic characterization of a notion of architectural reconfiguration over such a model and the development of the corresponding theory.
- The design of a framework to derive on-the-fly reconfigurations of systems, simulate their effects and analyze their impact on a (model) of the application.

4 Envisaged approach

- It is expected that the whole research will be framed in the context of Reo[3, 4], a model for exogenous coordination of software with strict service decoupling to support loose inter-component dependency, under development at CWI, Amsterdam. The dissertation will contribute to the development of Reo, at different levels (semantics, analysis and tools); its tool-oriented deliverables will be developed as plug-ins for ReoTools [11].
- In this sense, the project will start with the development (within the context of Reo) of a QoS-aware architectural model and specification language, on top of which specific architectural patterns could be defined and used in the sequel. Current work on QoS annotations and forms of stochastic composition for Reo, and associated (partial) tool support, will be taken as the starting point.
- Such architectural patterns should be annotated with QoS attribute values and equipped with some notion of *robustness*, measuring the quality of response to unpredictable change and avoidance of failure. Robustness is studied in several contexts. A possible approach relies on game-theoretic semantic models [2, 1], resorting to a notion of adversary to model environmental factors beyond the system control. Game-theoretic approaches enable a fine-grained understanding of the logical principles underlying interaction based on the system's structure, which may provide an interesting alternative to model endurance to failure.
- QoS analysis of architectural patterns and their reconfigurations may resort to suitable model-checking tools, namely Prism [10, 12], for probabilistic model-checking, and Ivy [5], a model based tool for the analysis of interactive systems design under current development at Minho.
- A challenging, but fundamental issue in this PhD project concerns modeling reconfigurations and the development of a suitable calculus of reconfigurations whose objects are expressions describing architectural patterns (in the language defined for the project's QoS-aware architectural model). Graph-based formalisms, in particular, graph rewriting techniques [6, 7, 9] will, most probably, be interesting candidates.
- The overall PhD project will include (and, to a certain extent, be driven by) 2 to 3 case studies in the envisaged area of application.

5 Context

This PhD project is integrated in a research line on *Formal techniques for architectural design* within the HASLab group. The candidate is expected to take part in the group regular scientific activities as well as to help in running exercise classes and supervising teamwork projects within graduate curricular units managed by HASLab staff. All of these activities will be suitably certified and mentioned in the candidate's diploma supplement.

It is envisaged a close collaboration with the Reo project, including short-term research stages at Prof. F. Arbab's group in CWI, Amsterdam.

References

[1] S. Abramsky. Algorithmic game semantics: A tutorial introduction. In H. Schichtenberg and R. Steinbrüggen, editors, *Proceedings of the NATO Advanced Study Institute, Marktoberdorf*, pages 21–47. Kluwer Academic Publishers, 2001.

[2] S. Abramsky and G. McCusker. Game semantics. In H. Schwichtenberg and U. Berger, editors, *Computational Logic: Proceedings of the 1997 Marktoberdorf Summer School*, pages 1–56. Springer-Verlag, 1999.

[3] F. Arbab. Abstract behaviour types: a foundation model for components and their composition. In F. S. de Boer, M. Bonsangue, S. Graf, and W.-P. de Roever, editors, *Proc. First International Symposium on Formal Methods for Components and Objects (FMCO'02)*, pages 33–70. Springer Lect. Notes Comp. Sci. (2852), 2003.

[4] F. Arbab. Reo: a channel–based coordination model for component composition. *Mathematical Structures in Comp. Sci.*, 14(3):329–366, 2004.

[5] J. C. Campos and M. D. Harrison. Interaction engineering using the ivy tool. In *ACM Symposium on Engineering Interactive Computing Systems (EICS 2009)*, pages 35–44, New York, NY, USA, 2009. ACM.

[6] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation - part i: Basic concepts and double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations*, pages 163–246. World Scientific, 1997.

[7] A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation - part ii: Single pushout approach and comparison with double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations*, pages 247–312. World Scientific, 1997.

[8] J. L. Fiadeiro. Software services: scientific challenge or industrial hype? In K. Araki and Z. Liu, editors, *Proc. First International Colloquim on Theoretical Aspects of Computing (ICTAC'04), Guiyang, China*, pages 1–13. Springer Lect. Notes Comp. Sci. (3407), 2004.

[9] R. Heckel. Tutorial introduction to graph transformation. In H. Ehrig, R. Heckel, G. Rozenberg, and G. Taentzer, editors, *Proc. 4th International Conference on Graph Transformations, ICGT 2008, Leicester, United Kingdom, September 7-13, 2008.*, pages 458–459, 2008.

[10] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In H. Hermanns and J. Palsberg, editors, *Proc. 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06)*, pages 441–444. Springer Lect. Notes Comp. Sci. (3920), 2006.

[11] C. Koehler, A. Lazovik, and F. Arbab. Reoservice: Coordination modeling tool. In *Conf. on Service-Oriented Computing (ICSOC-07)*, Lecture Notes in Computer Sciences 4749, pages 625–626. Springer, 2007.

[12] M. Kwiatkowska, G. Norman, and D. Parker. A framework for verification of software with time and probabilities. In K. Chatterjee and T. Henzinger, editors, *Proc. 8th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'10)*, volume 6246, pages 25–45. Springer Lect. Notes Comp. Sci. (6246), 2010.