Stochastic models for quality of service of component connectors

Moon, Y.J.

Chapter 1

Introduction

In Service-oriented Computing (SOC), services distributed over a network are composed according to the requirements of service consumers. Services are platform- and network-independent applications that support rapid, low-cost, loosely-coupled composition. Services typically run on the hardware of their own providers, in different containers, separated by fire-walls and other ownership and trust barriers. Their composition requires additional mechanisms (e.g., process work-flow engines, connectors, or glue code) to impose some form of coordination (i.e., orchestration and/or choreography).

The holy grail of service and component-based software engineering is to develop truly reusable software services and components that can be sold off-the-shelf and reused to build software systems [88]. Research on software composition plays a key role in this quest, as it offers flexible ways of plugging together components. Some approaches to software composition use textual glue code [64, 71, 38], usually in a scripting language, whereas others offer a more visual approach, where 'channels' or 'connectors' are used to compose components into a system (e.g. [2, 80, 14]). Connectors play the role of coordinating software, yet their functionality is traditionally more limited than scripting languages. This has changed with the advent of the notion of compositional connectors [2, 64]. In such a setting, connectors are formed by composing simpler connectors, such as channels, together. Several *coordination languages* have been proposed for software composition.

Coordination languages express various coordination patterns exhibiting combinations of synchronization, mutual exclusion, non-deterministic choice, and state-dependent behavior. Some have been used as component connector models, including Reo [2], Ptolemy [58, 36], Ptolemy II [59, 36], Orc [64], MoCha [80], Manifold [7], Linda [41], BIP [15], and pipe and filter architectures [81]. Although these models overlap in philosophy and functionality, Reo is the only one that enables propagation of synchrony though composition, mutual exclusion through connectors, and combination of synchrony and asynchrony [78, 73, 77].

1.1 Quantitative analysis of systems

In recent years, there has been an increasing interest in studying the behavior of software systems from a quantitative perspective. Consider a service-based system running in a call center that matches calling clients with the appropriate representatives that can provide them with the specialized customer service that they need. Challenges that the center might face include minimizing the number of customers waiting to be matched at any point (while not having to increase their number of employees and servers too much) and improving the quality of the matching service. The relevance of being able to propose solutions for such challenges cannot be underestimated, since resources are neither infinite nor free. In addition, the answers to *quantitative* questions have to be adapted according to the context: different services have different constraints. For instance, in the context of safety critical and time critical applications (like airplane and automobile control systems), if a request is waiting for more than a few seconds there could be disastrous consequences, whereas in other applications, such as a ticket booking website, a few seconds will not have too much of a negative impact.

As mentioned above, distributed services are platform independent and, therefore, heterogeneous, in the sense that, for instance, they are written in different programming languages. In such a setting, even if the QoS properties of every individual service and connector are known, it is far from trivial to build a model for and make statements about the end-to-end QoS of a composed system. For this purpose, over the past few decades, several stochastic methods, such as Stochastic Petri Nets (SPN) [79, 65] and Stochastic Process Algebra (SPA) [63, 49, 45], have been suggested in various application areas. SPN are useful for the analysis of computer systems since they allow the system operations to be precisely described by means of a graph which then translates into a Markovian model used to obtain performance estimates. Due to its graphical representation, it can easily be understood. In addition, the derivation of the Markovian model and its solution can be automated and transparent to the users. However, as typical of state-based models, they suffer from the state-explosion problem, and often, for a large SPN model, exact solutions cannot be computed. In addition, SPN essentially deal with asynchronous events and, hence, the synchrony of events is not propagated through composition [4]. SPA, on the other hand, offers a compositional specification framework. A complicated system can be modeled by first modeling its sub-systems and then the interaction between them. The main disadvantage of SPA is the lack of expressiveness of the timing distributions that can be used in the modeling: only negative exponential distributions are allowed.

In this thesis we focus on Reo, a channel-based coordination language which provides a flexible and expressive model for compositional construction of connectors that coordinate distributed services over networks. Reo has been around for many years now and much research has been done in order to turn it into an expressive, modular and usable language. One of the main streams in this research concerns formal semantic models for Reo. There have been several proposals: a coalgebraic model [9], colouring tables [30] which are used in the animation tool of Reo connectors in Extensible Coordination Tools (ECT) [35], and several automata models, particularly suitable for verification. Among the proposed automata models, each of which offers different expressiveness and modeling advantages, we mention Constraint Automata (CA) [12], Intentional Automata (IA) [31] and Reo automata [19]. CA are a basic and compact automaton model, which unfortunately does not support context dependency directly. Context dependency expresses behavior that depends on both the positive and negative availability of I/O requests on the boundary ports of a connector. To overcome this limitation of CA, IA and Reo automata were recently proposed. The Reo automata model is compact, quite close in spirit to the CA model, whereas the IA model is more verbose. In this thesis, we provide *quantitative* extensions of both IA and Reo automata.

First steps have been taken to extend Reo in order to accommodate QoS aspects of a system. In [5], Quantitative Reo and Quantitative Constraint Automata (QCA) were introduced. The QCA model integrates the QoS aspects of components/services and connectors that comprise an application to yield the QoS properties of that application, ignoring the impact of the environment on its performance, such as throughput and delays. QCA provide a useful model for service selection and composition [61], but, because it ignores the interaction with the environment, it does not provide a faithful model for the end-to-end QoS of a system. The latter can crucially depend not only on the internal details of a system, but also on how it is used in an environment, as determined, for instance, by the frequencies and distributions of the arrivals of I/O requests. Such stochastic aspects are not investigated in [5].

1.2 Thesis overview and contributions

The main aim of this thesis is to provide an expressive model wherein the specification of the overall end-to-end QoS of a composed service in a distributed environment can be carried out compositionally. We use as basis of our model Reo, which we extend with the power to specify stochastic aspects of a system. We provide two formal semantic models for this extension of Reo, based on the IA and Reo automata models mentioned above. Furthermore, in order to enable practical analysis of the end-to-end QoS of a system, we provide translation methods from the specification models into stochastic models (Markov Chains and Interactive Markov Chains). We have implemented all the methods presented in this thesis as plug-ins for the ECT tools [8] and, using them, we have modeled and analyzed a real application, the ASK system [83].

In Chapter 2 we mention the basic preliminaries of Reo and its semantics models. In addition, Stochastic Reo, a stochastic extension of Reo, is introduced, in which it is possible to specify the end-to-end QoS of a system. Stochastic Reo constitutes the only original contribution of this chapter and it is based on the paper:

[6] Farhad Arbab, Tom Chothia, Rob van der Mei, Sun Meng, Young-Joo Moon, and Chrétien Verhoef. From Coordination to Stochastic Models of QoS. In *COORDINATION*, volume 5521 of *Lecture Notes in Computer Science*, pages 268–287. Springer, 2009

In Chapter 3 we introduce Quantitative Intentional Automata (QIA), as a semantic model for Stochastic Reo. QIA extend the semantics of Reo by representing Reo channels and their channel ends separately and admitting annotation on them to describe data-flows through those channels and I/O request arrivals at the channel ends as stochastic events. In addition, QIA can be considered as an intermediate model for translation into stochastic models, in particular Continuous-Time Markov Chains (CTMCs), for stochastic analysis. The translation method from Stochastic Reo into CTMCs via QIA is also introduced in this chapter. This chapter is based on the following paper:

[6] Farhad Arbab, Tom Chothia, Rob van der Mei, Sun Meng, Young-Joo Moon, and Chrétien Verhoef. From Coordination to Stochastic Models of QoS. In *COORDINATION*, volume 5521 of *Lecture Notes in Computer Science*, pages 268–287. Springer, 2009

QIA can be seen as an extension of IA. In the above paper, the structure and basic definitions of QIA are different from the ones we now present in Chapter 3 since the reference for IA [31] was not available when the above paper was written. For the sake of consistency and coherence, we have completely rewritten the above paper to keep the definitions closer to the IA definitions.

In Chapter 4 we introduce Stochastic Reo Automata as an alternative semantic model for Stochastic Reo. In general, QIA have a large number of states, mainly due to the separate representation of I/O request arrivals and data-flows. Stochastic Reo Automata were designed to provide a more compact semantic model for Stochastic Reo. More importantly, Stochastic Reo Automata also enable an easy formal proof for their compositionality, which is lacking in the case of QIA. For general QoS aspects, Stochastic Reo Automata were extended with reward information to accommodate concerns such as CPU computation time and memory space. As an alternative model to QIA, Stochastic Reo Automata are also used to generate corresponding CTMCs. In addition, in this chapter, we discuss why Interactive Markov Chains (IMCs) [43] are not an appropriate semantic model for Stochastic Reo, and show the translation from Stochastic Reo into IMCs via Stochastic Reo Automata. This chapter is based on the following papers:

[68] Young-Joo Moon, Alexandra Silva, Christian Krause, and Farhad Arbab. A Compositional Semantics for Stochastic Reo Connectors. In *FOCLASA*, volume 30 of *EPTCS*, pages 93–107, 2010

[67] Young-Joo Moon, Alexandra Silva, Christian Krause, and Farhad Arbab. A Compositional Model to Reason about end-to-end QoS in Stochastic Reo Connectors. To apper in *Science of Computer Programming*, 2011

In Chapter 5 we describe the Reo2MC tool which is available as a plug-in for the ECT. Reo2MC is a fully automated tool which is able to automatically derive the QIA semantics of Reo models and their corresponding CTMCs. In addition, it provides bridges to existing third-party tools for stochastic analysis, such as PRISM[1] [57, 48], Maple, and MATLAB, by generating the input files for those tools. We also explain the usage of the Reo2MC tool. This chapter is based on the following paper:

[8] Farhad Arbab, Sun Meng, Young-Joo Moon, Marta Z. Kwiatkowska, and Hongyang Qu. Reo2MC: a tool chain for performance analysis of coordination models. In *ESEC/SIGSOFT FSE*, pages 287–288. ACM, 2009

In Chapter 6 we show a case study using the ASK system [83], an industrial software developed by the Dutch company Almende [1], and marketed by their daughter company ASK Community Systems [10]. The ASK system is a communication software product that acts as a mediator between service consumers and service providers. We model the ASK system using Stochastic Reo, and then translate the model into a CTMC in order to analyze it using PRISM. The rates used in this model were obtained by applying statistical analysis techniques on the raw values that we obtained from the real logs of an actual running ASK system. Since the translation target model is a CTMC, only exponential distributions are allowed as rates in the modeling. However, not all the distributions we obtained from the statistical analysis were exponential. In the case of properties involving rates that follow a non-exponential distribution, we also show in this chapter how to use the Reo simulator to obtain insights in the behavior of the system. This chapter is based on the following paper:

[66] Young-Joo Moon, Farhad Arbab, Alexandra Silva, Andries Stam, and Chrétien Verhoef. Stochastic Reo: a Case Study. Accepted for publication in TTSS 2011

1.2.1 Contributions

We summarize in the table below the main contributions of this thesis and the chapters where they can be found.

Stochastic Reo: a compositional model for specifying composite systems, where non-functional (QoS) aspects and the influence of the environment on their performance are taken into account.	Chapter 2
Quantitative intentional automata (QIA): an operational semantic model for Stochastic Reo	Chapter 3
Methods to translate QIA into CTMC	Chapter 3, Section 3.3

[1] http://www.prismmodelchecker.org/

Stochastic Reo Automata (SRA): an alternative compact semantic model for Stochastic Reo	Chapter 4
Methods to translate SRA into CTMC and IMC	Chapter 4, Sections 4.4 and 4.5
Formal proof of compositionality of SRA	Chapter 4, Section 4.2.1
Extension of SRA to specify more general QoS (reward information)	Chapter 4, Section 4.3
Reo2MC: a tool for the analysis of Stochastic Reo models	Chapter 5
Case study of a real commercial system, the ASK system, using the Reo2MC tool and the Reo simulator	Chapter 6