Unified Modelling Functional and Non-Functional Aspects of Web Services Composition Using PTCCS

Fangxiong Xiao, Guangxi University of Finance and Economics and Nanjing University of Aeronautics and Astronautics, China
Zhiqiu Huang, Nanjing University of Aeronautics and Astronautics, China
Zining Cao, Nanjing University of Aeronautics and Astronautics, China
Jun Hu, Nanjing University of Aeronautics and Astronautics, China
Linyuan Liu, Nanjing University of Aeronautics and Astronautics, China
Min Yuan, Nanjing University of Aeronautics and Astronautics, China

ABSTRACT
Unified modelling and analyzing functional and non-functional aspects of Web services composition reduces the gap caused by modelling them in different models respectively, increases development speed, and reduces development cost. Though existing process algebras have advantages over modelling concurrent and composite systems, including Web services composition, they lack the capability of modelling cost that is an important non-functional property. This paper proposes a process algebra called PTCCS (Priced Temporal Calculus of Communicating Systems) based on the existing TCCS (Temporal Calculus of Communicating Systems) by associating action of TCCS with price information. The authors present the syntax and semantics of PTCCS and prove that PTCCS extends TCCS with cost modelling capability. Cost state space for a process in PTCCS is proposed and an algorithm to generate it to support cost optimal Web services composition or time optimal Web services composition is presented. The authors present another algorithm that can generate cost optimal path for cost optimal Web services composition and time optimal path for time optimal Web services composition based on cost state space. An experiment shows that PTCCS can effectively support unified modelling functionality, time, and cost of Web services composition.

Keywords: Cost, Non-Function, Services Composition, Time, Unified Modelling

DOI: 10.4018/jwsr.2011100103
1. INTRODUCTION

Web services are autonomous software systems identified by Universal Resource Identifiers (URIs) which can be advertised, located, and accessed through messages encoded according to XML-based standards and transmitted using Internet protocols. Web services encapsulate application functionality and information resources and make them available through programmatic interfaces, as opposed to the interfaces provided by traditional Web applications which are intended for manual interactions. In addition, since they are intended to be discovered and used by other applications across the Web, Web services need to be described and understood both in terms of functional capabilities and non-functional properties, i.e., Quality of Service (QoS), such as time, cost, reliability, maintainability, reputation, etc. (Zeng, 2004). Web services composition is a new pattern to construct a larger application by composing existing Web services from different organizations to meet functional requirements of the larger application, and QoS-driven Web services composition features that a larger application will be constructed with optimal QoS by selecting Web services with QoS constraints besides functional requirements. Though Web services composition is a new pattern to construct applications, it also has similar lifecycle like that of the classical software engineering. In modelling or designing phase of Web services composition, building abstract models of Web services composition and verifying whether both functional and non-functional aspects are met, will be helpful to reduce risks for later Web services composition development, increase development speed, and reduce development cost. Currently, there are several popular modelling tools for Web services composition, e.g., Web Service Business Process Execution Language (WS-BPEL), Unified Modelling Language (UML), and formal modelling tools. WS-BPEL is an XML-based programming language to describe high level business processes that can be executed in WS-BPEL engines. UML is a standardized general-purpose modelling language in the field of software engineering and includes a set of graphical notation techniques to create abstract models of specific systems. Formal modelling tools include Automata, Petri Net, and Process Algebra (PA), etc. (Peled, 2001). All these tools are useful for modelling Web services composition, but they are used in different phases for different purposes. Generally speaking, in terms of abstract level, formal modelling tools are high, UML is medium, and WS-BPEL is low. Software developers may often enjoy using UML’s graphical notations to model software, for graphical notations are easily understood, compared with other tools. For Web services composition applications, developers sometimes directly build up the application without UML modelling. How to make sure UML models and BPEL programs are correct? Formal verification is a useful method besides testing to make sure the correctness, since formal tools take some advantages over other tools (Peled, 2001). First, they are in high abstract level and some trivial model details can be ignored, which is helpful to understand and capture the essence of software. Second, formal notations are unique and unambiguous, which is useful to enhance reliability of formal models. Third, lots of formal tools have been available to support formal modelling and verification. There are three ways to set up formal models for Web services composition. The first way is to directly set up formal models in abstract level (Peled, 2001). This way is mainly for simply systems or for case studies. The second way is to map models built with UML to formal models (Lam, 2008). The third way is to map programs written with WS-BPEL to formal models (Lucchi, 2007; Salaün, 2004). Once formal models are set up, functional aspects and some non-functional aspects can be verified with formal tools. After formal verification, users will be high confidence with the Web services composition applications. Since our modelling method is located in abstract level, in this paper we follow the first way to set up formal models and slightly present how to map BPEL to formal model by the way.
Enhancing the Web Service Description and Discovery Processes with QoS
[www.igi-global.com/chapter/enhancing-web-service-description-discovery/26077?camid=4v1a](www.igi-global.com/chapter/enhancing-web-service-description-discovery/26077?camid=4v1a)