Chapter 116
Towards an Understanding of Performance, Reliability, and Security

Ye Wang
Zhejiang Gongshang University, China

Bo Jiang
Zhejiang Gongshang University, China

Weifeng Pan
Zhejiang Gongshang University, China

ABSTRACT

Service-oriented computing is a promising computing paradigm for software applications. However, there are a lot of key research issues in the service-oriented computing paradigm. Quality of service (QoS) is a key factor to resolve these issues as well as a crucial aspect in the design of service-oriented systems as it directly touches the concerns of the service users. Therefore, it is necessary and important to achieve a deeper understanding of different types of QoS properties. This chapter introduces three QoS properties, namely performance, reliability, and security, and provides an overview of these QoS properties and offers an in-depth analysis of the issues, challenges, and research opportunities of QoS properties in designing and developing service-oriented systems.

INTRODUCTION

According to J. Spohrer, service-oriented systems are "value co-creation configurations of people, technology, internal and external service systems connected by value propositions, and shared information (such as language, laws, measures, models, and so on)" (Spohrer, 2007). A service-oriented system (Espadas et al., 2013) delivers a set of services that are cooperating with each other over the Internet, which are therefore called "web services". Web services are delivered in a new computing paradigm called "service-oriented computing". Service-oriented computing paradigm plays an increasingly crucial

DOI: 10.4018/978-1-5225-7598-6.ch116

role in modern world industries. Even in cloud computing, services are irreplaceable elements (Wang & Sun, 2015). In service-oriented systems, software developers create software applications through composing web services. Web service composition helps developers to solve complex problems by combining published basic services and ordering them to meet specific service requirements from user requests (Song & Lee, 2013).

Service-oriented systems offer many benefits to software developers and software users. For example, they allow developers to support many customers with a single version of software and provide services to their customers 24/7. Over the past decade, a large number of service-oriented systems have been developed to aid a wide range of real-world applications (Wang et al., 2011). For example, E-tourism service systems (see for example: www.visitcalifornia.com) have been used by tourists to plan their trip and book tickets as well as hotels. PayPal (www.paypal.co.uk), an online payment service, has been used by many e-commerce applications worldwide. Many investment banks or brokerages nowadays offer their customers trading services such as order management services, stock trading services, bond trading services and so on, and have been playing an increasingly pivotal role in the financial sector (Yang et al., 2011).

There are a lot of key research issues with regard to service-oriented systems, involving service modeling and specification (Cardoso et al., 2010; De Virgilio, 2010), service selection (Sweeney et al., 2016), web service composition algorithms (Hiel & Weigand, 2010; Leitner et al., 2010), verification of composition correctness (Rai et al., 2015), and handling composition faults (Lemos et al., 2016), etc. Rather than only concentrating on functionality, recent work puts more emphasis on Quality of Services (QoS) properties (Calheiros et al., 2015). With many businesses providing same or similar services, QoS properties have become an important indicator for good services (Zhao et al., 2007). Yet, satisfying QoS requirements in service-oriented systems has proved to be more challenging than satisfying functional requirements, due to the following characteristics (Yang et al., 2011): First, QoS properties are system-level requirements which cannot be assigned directly to individual system components; instead, they need to be planned at the infrastructure-level as a whole, with their design aspects then entrusted to system components. Second, QoS properties are often interdependent and their realization in a system requires a collective and coordinated behavior of the system components and a system-level design strategy. Third, QoS properties are application-specific and their fulfillment necessarily requires a specific design approach germane to their applications. Finally, QoS properties are not only a design time concern, but also most crucially a runtime concern. Satisfying QoS properties means designing runtime mechanisms that can maintain the system's QoS properties throughout the execution time.

Through the investigation of several service-oriented systems in different sectors (Wang & Sun, 2015; Wang et al., 2011; Yang et al., 2011), this chapter has identified a core set of common QoS properties necessary for most service systems, which are performance, reliability and security. It is believed that these QoS properties are also universal to any service-oriented system. The aim of this chapter is therefore to provide an overview of these common QoS properties, give a precise definition of these QoS properties and offer an in-depth analysis of the issues, challenges and research opportunities of QoS properties in designing and developing service-oriented systems.