Chapter 6
Adding Semantics to QoS Requirements

Ester Giallonardo
University of Sannio, Italy

Eugenio Zimeo
University of Sannio, Italy

ABSTRACT

To lead the software-service revolution and to make real the promise of Business to Business interactions, QoS and Semantics play key roles in every phase of Web processes’ life-cycle. Ontology-based approaches for specifying QoS enable machines to reason autonomously and dynamically on QoS knowledge, aiding the development of open and large-scale distributed applications. This chapter presents the impact of Semantics for the management of QoS Requirements in Service-based Applications, focusing on the onQoS ontology, its role for specifying service requirements and to support queries for service discovery through the onQoS-QL language. The introduced concepts and the results presented in the chapter pave the way to new technology horizons where the semantics could represent the global glue for Web processes.

INTRODUCTION

In ITUT-T Recommendation E.800, Quality of Service (QoS) is defined as “collective effect of service performance that determines the degree of satisfaction by a user of the service”. QoS information is a key aspect in order to design, identify, form, put in operation and dissolve service level agreements. It influences also the software components in charge to ensure service trust: the service providers with effectively “best” QoS can be more easily identified and preferred in future similar circumstances utilizing, for example, reputation-based mechanisms. So, QoS impacts on service selection, on business process modifications and refinements and reduces the enterprise risk associated to the adoption of Web service technologies.

DOI: 10.4018/978-1-60960-493-6.ch006

Copyright © 2011, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
Adding Semantics to QoS Requirements

Figure 1. Phases of a Web process life-cycle

QoS needs to be taken into account in the overall life-cycle of a Web process. It guides a proper evolution of the process by continuously providing information about the actual behavior of each service/resource adopted. In particular, according to Van der Aalst et al. (2003), the business process life cycle presents the following phases: process design, system configuration, process enactment, and diagnosis. We retain that the process design phase objective is defining both the functional and non-functional aspects needed to getting a system to support requested business objectives. Starting from business needs, Composition, Discovery and Negotiation Components can work for making an electronic model of the business processes. This model needs to specify what are the services, the related control flow, what they have to do, and how they have to be supported, but it is not always necessary to define also who will perform them.

Discovery systems use non functional requirements in order to reduce the search space by identifying only the Web services that guarantee the desired level of QoS, while negotiation systems utilizes standard protocols in order to create agreements between service consumers and providers, based also on QoS. These agreements can specify, in addition to other aspects, when and how the service can be monitored (diagnosis points and/or monitoring tools). So, service binding and service monitoring can be driven by service level agreements.

During the phase of system configuration, Binding Components set up the system, locating the concrete Web services to bind to the previously designed abstract processes. Instead, Adaptive Components can exploit semantics in order to configure the system for reducing human intervention during process execution.

In the process enactment phase, the software component in charge of executing the process (for example a Workflow Engine as in our case) is driven by abstract and concrete process specifications to deploy and execute processes. These specifications define also QoS requirements: when the Enactment System meets process activities abstractly modeled (and not yet concretized), the system configuration phase needs to be reactivated and the Engine can act as service requestor of Discovery or Negotiation Systems.

Process diagnosis, instead, regards the runtime evaluation of the satisfaction of QoS requirements specified at design-time in the process definition phase. The objective is to systematically examine the single service execution in order to determinate the satisfaction of QoS desiderata and the progress in achieving process objectives.

Semantic technologies can be used for improving interoperability between these different components, which can so exchange and use QoS information with semantic understanding.