Chapter 15
Design and Deployment of Service Oriented Applications with Non-Functional Requirements

László Gönczy
Budapest University of Technology and Economics, Hungary

Dániel Varró
Budapest University of Technology and Economics, Hungary

ABSTRACT

As the use of SOA became a mainstream in enterprise application development, there is a growing need for designing non-functional aspects of service integration at the architectural level, instead of creating only technology specific assets (configuration descriptors). This architectural design supports flexibility and early validation of requirements. This chapter presents a model-driven method supporting the automated deployment of service configurations. This deployment technique is supported by an extensible tool chain where (i) service models are captured by a service-oriented extension of UML enabling to capture non-functional requirements, and (ii) configuration descriptors for the target deployment platform are derived by automated model transformations within the VIATRA2 framework.

INTRODUCTION

Service-Oriented Architectures (SOA) provide a flexible and dynamic platform for implementing business services. Due to the rapid increase in the number of available services, more emphasis is put on their reliability, availability, security, etc. In order to meet such non-functional requirements, a service needs to be designed for reliability by making design decisions on an architectural level. Recently, various non-functional parameters of services have been identified by various
XML-based web service (WS) standards such as WS-Reliability, WS-ReliableMessaging, WS-Security, etc. While these properties are attached to business-level web services, they, in fact, specify the configuration and behavior of the service infrastructure, i.e., services that are not part of a specific application, but play a dedicated role in the underlying service middleware. A focal issue in the service infrastructure is to provide reliable messaging between services, where the reliable delivery of a message can be transparently guaranteed by the underlying platform.

Non-functional properties are captured at a low implementation-level by using dedicated XML deployment descriptors. This is unfortunate since the core business functionality of a service is typically designed using high-level visual notations like the Unified Modeling Language (UML) or the Business Process Modeling Notation (BPMN). As a consequence, service configurations cannot be designed at a high architectural level.

Up to very recently, Web services messaging standards used to capture a different subset of non-functional parameters making even closely related standards incompatible with each other. For instance, the definition of timing parameters or message ordering (e.g., InvokeInOrder parameter) was different in the WS-ReliableMessaging and WS-Reliability standards, although the underlying concepts were closely related. While recently, WS-ReliableMessaging has evolved to being the de facto standard for reliable message delivery, the standard is continuously subject to minor, non-conceptual changes, which complicates the maintenance of existing configuration files. Furthermore, each middleware typically implements the standard with middleware-specific extensions. As a consequence, the portability of service configurations is also problematic.

To tackle these problems, we propose a model-driven approach to efficiently design and deploy standards-compliant service configurations with non-functional parameters (with special focus on reliable messaging). First, we expect the service developer to create a structural diagram of the service configuration and extend it with constraints on the communication using a high-level, visual modeling notation like UML. Such constraints (implemented as contracts between communicating parties) may capture requirements on the security and reliability of the communication expressed using standard platforms like WS-Security [WS-a] and WS-Reliability [WS-b] issued by OASIS.

From such engineering models, we automatically generate service descriptors (in Web Service Description Language, WSDL), and configuration descriptors for standard platforms like Apache Axis, including the definition of non-functional requirements, and server-side deployment artifacts. These artifacts have to be extended with the Java source code of service implementations and client-side stubs of services to allow the service requester to seamlessly integrate the server-side functionality without any a priori knowledge on the service communication (including acknowledgements, encoding, etc).

Hereby, we present a model-driven method supporting the automated deployment of service configurations. This deployment technique is supported by an extensible tool chain where (i) service models are captured by a service-oriented extension of UML enabling to capture non-functional requirements, (ii) configuration descriptors for the target deployment platform are derived by automated model transformations within the VIATRA2 framework.

This chapter is structured as follows: we first present structural modeling of services on a case study with non-functional requirements. Then we present a non-functional extension in UML for modeling services which show conceptual resemblance to Service-Level Agreements (SLAs) based upon existing reliability and security standards of Web service messaging. Then we present an overview of our automated model-based deployment technique including a discussion of target deployment artifacts as well as relevant