Chapter 5
Service Identification and Specification with SoaML

Michael Gebhart
Gebhart Quality Analysis (QA) 82, Germany

ABSTRACT
This chapter focuses on the identification and specification of services based on prior modeled business processes and legacy systems. The resulting service interfaces and service components formalized by using the Service oriented architecture Modeling Language (SoaML) describe the integration of legacy systems into a service-oriented application landscape. The legacy systems provide services for integration purposes and represent the implementations of service components. Additionally, the resulting architecture allows functionality of legacy systems to be replaced with functionality provided by external cloud services. According to model-driven development concepts, the formalized service interfaces and service components as part of the service designs can be used to automatically derive service interface descriptions using the Web Services Description Language (WSDL). These descriptions enable the technical integration of legacy systems. If necessary, service implementations based on the Service Component Architecture (SCA) and the Business Process Execution Language (BPEL) can be generated.

INTRODUCTION
Nowadays, companies organize their Information Technology (IT) by means of services. This means that systems within the application landscape provide and require services business aligned and supporting certain business processes. In order to reuse existing legacy systems and their capabilities, their integration into such a service-oriented application landscape is targeted. This integration requires a determination and realization of required services that constitute the gateways to existing functionality. These services can be provided by the legacy systems in order to enable their integration or these services can be required by the systems in order to invoke remotely provided functionality, for example, cloud services as part of a public cloud (Chang, Abu-Amara, & Sanford, 2010, p. 46). In this case, these external services are embedded into an internal service-oriented application landscape.

Before services are realized, their detailed planning is necessary. This means that before
performing the implementation phase they must be methodically identified and specified as part of a systematic design phase. The final service designs are formalized using common and widespread modeling languages. This enables the usage of modeling tools and an analysis and revision of the services already during the design phase, which reduces effort and costs.

The following chapter introduces a methodology for the systematic identification and specification of services that enable the integration of legacy systems into a service-oriented application landscape in order to integrate already implemented functionality and thereby increase its reuse. For this purpose, the methodology combines a derivation of services that support prior defined business requirements with a subsequent revision that considers functionality provided by legacy systems.

Since the systematic identification of services expects a clear understanding of the underlying business requirements, this chapter also introduces artifacts that are created during the requirements analysis phase and their formalization using common, standardized, and wide-spread modeling languages, such as the Unified Modeling Language (UML) (Object Management Group, 2010) and the Business Process Model and Notation (BPMN) (OMG, 2011). Afterwards, the identification and subsequently the specification of services are described. Also during these phases, the modeling of resulting artifacts is considered. In this context, the Service oriented architecture Modeling Language (SoaML) (OMG, 2009) is applied as it is a standardized metamodel and UML profile for modeling service-oriented architectures. It has been created out of a request for proposal (OMG, 2006) and gains increasing vendor and tool support. The usage of these common and wide-spread languages enables the application of existing development tools and the integration of this methodology into existing tool chains.

The chapter concludes with an outlook into the implementation phase based on the prior specified services. In this context, the transformation of the created artifacts into service implementations using the Web Services Description Language (WSDL) (World Wide Web Consortium, 2007b), Service Component Architecture (SCA) (Open Service Oriented Architecture, 2009), and Business Process Execution Language (BPEL) (Organization for the Advancement of Structured Information Standards, 2007) is outlined. This embeds the methodology into a model-driven development scope as introduced by Stahl, Voelter, Bettin, and Stockfleth (2006) and applied in Hoyer, Gebhart, Pansa, Link, Dikanski, and Abeck (2009, 2010).

BACKGROUND

This section gives a brief introduction into the foundations that constitute the basis for the contribution of this chapter. Elementary terms, such as service-oriented architectures or cloud computing, are taken for well known. Only divergences or aspects that are considered as important are repeated. Instead, this section focuses on the specifics of this contribution, such as service designs and their formalization using SoaML. The definitions of these terms are taken from existing work and enhanced with additional aspects if necessary.

Service Design

The design of services constitutes an elementary task when establishing a service-oriented architecture and integrating existing systems. The result of this design phase is a set of so-called service designs with each of them concretely specifying one certain service. The elements of a service design can be derived by considering development processes for services as they are introduced in existing work.

According to Erl (2006, 2008), within the here-mentioned service-oriented design phase especially the service interfaces are created. The elements of a service interface are oriented towards the specification of WSDL and explained in Erl
Related Content

A Service-Oriented User Interface for the Next Generation Web
[www.igi-global.com/chapter/service-oriented-user-interface-next/71998?camid=4v1a](www.igi-global.com/chapter/service-oriented-user-interface-next/71998?camid=4v1a)

An Adaptive System for Retrieval and Composition of Learning Objects
[www.igi-global.com/article/adaptive-system-retrieval-composition-learning/61315?camid=4v1a](www.igi-global.com/article/adaptive-system-retrieval-composition-learning/61315?camid=4v1a)

Recommending Mechanisms for Modularizing Mobile Software Variabilities
[www.igi-global.com/chapter/recommending-mechanisms-modularizing-mobile-software/66474?camid=4v1a](www.igi-global.com/chapter/recommending-mechanisms-modularizing-mobile-software/66474?camid=4v1a)

MoDSEL: Model-Driven Software Evolution Language
[www.igi-global.com/chapter/modsel-model-driven-software-evolution/71833?camid=4v1a](www.igi-global.com/chapter/modsel-model-driven-software-evolution/71833?camid=4v1a)