Chapter 8
Business Rules Enabled
Semantic Service Discovery and
Selection for B2B Integration

Andreas Friesen
SAP Research, Germany

ABSTRACT

In service-oriented business applications, B2B integration happens when a service requester invokes services of one or more service providers. Typically, there are several candidate services with similar capabilities that can be chosen by a requester in order to serve his business needs. The selection of the service to be invoked may depend on different functional and non-functional properties. The non-functional properties usually address security, reliability, performance, and so forth. The functional properties address the business process interplay at the level of the technical Web service interface and the message choreography associated with it. At the technical integration level, the description of functional and non-functional service properties has been exhaustively addressed in the scientific literature in the past. The business level however, namely, the requester’s business need, the business meaning of an offered service, and the capability of a service provider to successfully perform the requested business transaction, has been rather ignored. This chapter describes a solution for service discovery and selection at the business level, that is, at the level of offered business capability of a service provider and the ability to serve a concrete requested business transaction. The proposed solution is based on semantic interpretation of offered service capabilities, contractual restrictions, business rules of the requestor specifying selection preferences, and the parameters of the run-time service request. The applicability of the proposed solution is demonstrated on a shipper-carrier integration scenario.

DOI: 10.4018/978-1-60566-804-8.ch008
INTRODUCTION

The advent of Service-oriented Architecture (SOA) and Web Services (WS) opened new possibilities for smooth Enterprise Application Integration (EAI) in intra- and inter-enterprise scenarios in a loosely-coupled manner. In principle, Web Services enabled enterprise systems can be used by anyone, from anywhere, at any time, and on any type of platform. The providers can offer their business functionality deployed as Web Services on a Web Server and publish their specifications to a repository offering them to potential users (requesters). A potential requester can discover, select, compose, bind and invoke the offered Web Services in order to achieve its business goals.

The traditional Web Services technology stack comprises at least the following core technologies: Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL) and Universal, Description, Discovery and Integration (UDDI). SOAP is used for communication with a Web Service. WSDL describes the Web Service interface. UDDI provides publishing and discovery functionality for Web Service specifications and capabilities. Additionally, mainly industry-driven, a large set of WS-* standards emerged during the last years and covers in the meantime near to any functional and non-functional property or extensibility mechanism associated with Web Services technology (Krafzig, 2005).

However, the lack of formally represented semantic meaning in this technology stack causes the tasks of discovering, selecting, composing, and binding Web Services being considered as manual steps performed by a human.

On the other hand, Semantic Web and Semantic Web Services (SWS), mainly driven by the academic community, promises new standardized means to formally capture the representation of the semantic meaning of data and interfaces. This enables the machines to automatically reason and to draw conclusions about the “intended meaning”.

The so-called Semantic Web Services promise a higher degree on automation concerning discovery, composition, invocation, and monitoring of Web Services. Several logical languages, ontologies, and frameworks for semantic annotation or description of Web Services have been proposed, e.g., OWL, OWL-S, WSMO and SA-WSDL (McGuinness, 2004; Martin, 2004; Fensel, 2006; Farrell, 2007). All of the proposed Semantic Web Services Frameworks conceptually support, or at least provide, a vision concerning the lifecycle phases of the Web Service usage process. However, also all of them are, in a sense, rather meta-frameworks since they fall short in describing the exact realization of the single phases. This becomes obvious, if use case dependent requirements have to be taken into account, e.g., association of the single phases of the Web Service usage process with design, configuration or run-time of an integration scenario. The concrete realization of one phase influences the subsequent phases.

Furthermore, the distinction of functional and non-functional properties describing service capabilities becomes very important for the realization of single service usage phases finally leading to the selection of the preferred service to be invoked.

The non-functional properties usually address service usage aspects concerned with security, trustworthiness, reliability, performance, etc. The non-functional properties are, of course, inevitably important in open environments like SOA and B2B and are often even mission-critical for successful business transactions. Nevertheless, by their nature they play a supportive role and come into play only after functional requirements could be met at a satisfactory level.

The functional properties address the business process interplay at the level of the technical web service interface and valid message choreography behind it. Furthermore, service properties and capabilities at the business level, i.e., the “business meaning” and “business level semantics” of a provided service and the capability of a service