

Introduction

Mathias Weske

1.1 State-of-the-art and Motivation

While service oriented computing has recently gained extensive momentum in both industry and academia, reality lags far behind expectations. Major software vendors hook on the service paradigm and tailor their software systems towards services, but a thorough and consistent design of applications based on loosely coupled services has yet to be achieved.

From its beginning, standardisation has been an important factor in service computing. However, rather than being integrated, standardisation efforts are rather independent from each other, so that the complex interplay between the various aspects in service computing is not accounted for by standardisation efforts.

While there are service based software applications in place, they realise just a minimal potential of service oriented computing, for instance standardised data and message formats. While Burbeck [37] has already identified dynamic binding of services at runtime as a core functionality of service based environments in 2000, dynamic binding of services has yet to be achieved. The main reason is the lack of rich service specifications, concepts, and tools to process them. The limitations of state-of-the-art service oriented systems can be characterised as follows:

- *Static Discovery and Binding*: Static binding of services means that an application invokes a service during execution time. The service is hard wired to the application, so that when the service fails, the application will also fail, unless additional measures are taken. Static binding of services can be realised by syntactic service specifications during the development of service based applications. These services are discovered manually by browsing and searching facilities, typically provided by a dedicated software component, called service registry. Services are then hard-wired to the application, which effectively realises a strong coupling, which contradicts the intended service paradigm of loose coupling.

The main reason for this limitation is the lack of rich specifications of services as well as proper domain ontologies that would facilitate dynamic service discovery and binding. Service description languages like *Web Services Description Language (WSDL)* [44] can only specify the technical syntactic interface of a service. Thus, what the service actually does, the semantics of the service, remains unspecified.

- *Fixed Service Landscape*: Static service binding restricts service based applications to a fixed service landscape. This means that new and improved services that become available after the application was developed but before the application is executed, cannot be used without modifying the application. This is a severe limitation, because adapting existing applications to use new services requires changing the application code and, thus, incurs considerable overhead.
- *Static Service Composition*: To fulfil particular goals, multiple services need to be performed according to underlying execution constraints. These execution constraints are specified in service compositions. Static service composition characterises the composition of existing services as a manual task. The service based application is more flexible and can adapt to changing service landscape with reduced effort, if services are composed dynamically, based on rich semantic specifications.
- *Poor Service Level Agreement Specification*: The service based application needs to define non-functional parameters for externally invoked services, for instance, related to response time or cost of using the services. As a result, service platforms need to be aware of non-functional quality of service parameters. Currently available platforms do not have these required capabilities. There is also no established interoperable specification language for defining, agreeing on, and monitoring such contracts between the consumer and provider of services.

This book introduces advanced concepts in service provisioning and service engineering, including semantic concepts, dynamic discovery and composition, and illustrates them in a concrete business use case scenario. To prove the validity of the concepts and technologies, a semantic service provisioning reference architecture framework as well as a prototypical implementation of its subsystems and a prototypical realisation of a proper business scenario are presented. The book goes way beyond current service based software technologies by providing a coherent and consistent set of technologies and systems functionality that realises advanced concepts in service provisioning.

In particular, the following advanced topics are addressed in this book. These are features on the technological level, which are appointed by the application of a compliant platform for service provisioning. On a business level, these achievements lead to a reduction of costs in development and maintenance. Furthermore the semantic service provisioning platform supports continuous adaptation of service centric applications towards environmental changes and upcoming business needs.

- *Seamless integration of heterogeneous existing services*: Service integration is achieved by the provision of an elaborated methodological approach embedded

into a set of integrated tools. These tools assist the user in her task of integrating existing heterogeneous services into the platform. Starting point for integration is a domain-specific ontology from which a specific type system is derived. The semantic service provisioning platform supports the mapping to this specific type system. Integrated services are augmented with semantic descriptions regarding service functionality and non-functional properties. In terms of the business dimension, this achievement reduces the maintenance and modification costs for large numbers of available services. It has the potential to ease integration of services and data formats. The envisaged architecture allows an easy discovery and interoperability of integrated services.

- *On-demand creation of service compositions:* Automated semantic-enabled service composition allows creation of new services by composing existing services on-demand and at run-time for individual user requests. For this purpose, the semantically described functionality of services is used. Current service integration approaches base on manual programming, making it hard to cost-effectively maintain and modify the complex service world. Therefore, the interface to services is lifted from a manual to a logical level, and supports service composition based on automated tools. The adaptive service composition implies a cost reduction in service provisioning.
- *Reliable service provision with assured quality of service:* The semantic service provisioning platform supports the description, negotiation, and realisation of non-functional service quality parameters. In cases of service failures or violations of agreed quality parameters, the platform reacts adaptively through re-enactment, re-binding or re-composition activities. The future service world will be based on global and dynamic services, which can be composed to answer the needs of complex service requests. Dynamic service re-enactment, re-binding or re-composition provide alternative solutions for non-reliable services, thus provides the end-customer with a reliable service delivery.

1.2 Scope and Organisation

This book introduces concepts and technologies in semantic service engineering and provisioning. A use case scenario illustrates the concepts and shows the validity of the technologies used. The organisation of this book follows roughly the lifecycle of a service based application, from the specification of concepts and services to service integration, composition, and finally enactment.

Chapter 2 introduces the foundation of this book by providing common terminology and a use case scenario that is used throughout the book. Finally, the phases involved in service provisioning are discussed.

Chapter 3 introduces semantic concepts in service engineering based on ontologies. In particular, ontology languages are introduced to capture the main concepts and their relationships. These ontology languages are used to express domain ontologies and service ontologies. Domain ontologies represent the semantic concepts of the particular domain of a service based application, while service ontologies define

how services are specified. A section on matchmaking explains how semantic specifications of services and domain ontologies can be used to generate new knowledge in service engineering, for instance, to decide which service actually fulfils the needs of a given service specification.

Service enabling is studied in Chap. 4. At a technological level, the generation of code for legacy integration is essential to service enabling, because it allows efficient legacy integration. Concepts and technologies in software generation are used to develop wrappers to existing software that realise the integration of services. Based on the specification of domain ontologies, the semantic specification of services is introduced.

One of the most important concepts in service provisioning is the composition of services to value-added service compositions. Chapter 5 looks at various techniques to compose services, ranging from manual composition to assisted composition and, finally, to runtime composition. To provide a broad perspective on research work in this area, Chap. 5 completes with a section on service composition and binding as developed in the Integrated Project SUPER, supported by the EU in the Sixth Framework Programme.

The enactment of service compositions is addressed in Chap. 6. Based on a set of enactment strategies, service monitoring and service profiling is addressed. The chapter completes with technologies to handle faults, based on the rich specification of services.

The service infrastructure is presented in Chap. 7, starting with middleware concepts and technologies, and proceeding to Web services technologies and a discussion of the service infrastructure for semantic service provisioning.

A service engineering methodology is introduced in Chap. 8. This methodology explains in detail the steps involved in the conceptual design and technological development of service based applications using the approach presented in this book.

While the introduced use case scenario is used to illustrate the concepts and solutions in the chapters, a consistent and sufficiently complete discussion of the realisation of this scenario is provided in Chap. 9.



<http://www.springer.com/978-3-540-78616-0>

Semantic Service Provisioning

Kuropka, D.; Tröger, P.; Staab, S.; Weske, M. (Eds.)

2008, XII, 226 p. 71 illus., Hardcover

ISBN: 978-3-540-78616-0