A rule based approach to the service composition life-cycle

Yang, J.; Papazoglou, M.; Orriëns, B.

Published in:
Proceedings of the 4th International Conference on Web Information Systems Engineering

Publication date:
2003

A Rule Based Approach to the Service Composition Life-Cycle

Jian Yang, Mike P. Papazoglou, Bart Orriëns, and Willem-Jan van Heuvel
Infolab, Tilburg University
PO Box 90153, 5000 LE, Tilburg, Netherlands
{jian,mikep,b.orriens}@uvt.nl

Abstract

Web services are becoming the prominent paradigm for distributed computing and electronic business. This has raised the opportunity for service providers and application developers to develop value-added services by combining existing web services. However the current web service composition solutions, even for the applications developed on the basis of the standard Business Process Execution Language for Web Services (BPEL for short) , are rather restricted and inflexible as they lack proper support for generating dynamic compositions and for managing the service composition life cycle. The ReServCom project proposed here aims to remedy this situation by introducing a rule based approach for web service composition which combines best practices from rule base systems and software engineering to support parameterization, dynamic binding, and flexible service compositions.

1. Introduction

The Web has become the means for organizations to deliver goods and services and for customers to search and retrieve services that match their needs. Web services are self-contained, Internet-enabled applications capable not only of performing business activities on their own, but also possessing the ability to engage other web services in order to complete higher-order business transactions. The platform neutral nature of web services creates the opportunity for building *composite services* by combining existing elementary or complex services, possibly offered by different enterprises. For example, a `travel plan` service can be developed by combining several elementary services such as `hotel reservation, ticket booking, car rental, sightseeing package,` etc., based on their WSDL description. We use the term *composite service* to signify a service that employs and synthesizes other services. The services that are used in the context of a composite service are called its constituent services. Although some standards (e.g., BPEL) are emerging, the current web service composition solutions, even for the applications developed on the basis of the standard BPEL [1], are rather restricted and inflexible as they lack proper support for the generation of dynamic compositions and the management of the service composition life cycle. The on going **ReServCom** project aims to remedy this situation by introducing a rule based approach for web service composition which combines best practices from rule base systems and software engineering to support parameterization, dynamic binding, and flexible service compositions.

Rules are logical statements about how a system operates. Some of these rules may be expressed in the language of the business, referring to real-world business entities, and are therefore called Business Rules. Business rules can represent among other things typical business situations such as escalation ("send this document to a supervisor for approval"), managing exceptions ("make sure that we deal with this within 30 min or as specified in the customer's service-level agreement"). Our conviction is that business rules can be used in the context of service composition to determine how the composition should be structured and scheduled, how the services and their providers should be selected, and how run time service binding should be conducted.

With a vast service space to search, a variety of services to compare and match, and different ways to construct composed services, service composition is too complex and too dynamic to handle manually. The alternative to manual control is an automated process of service composition governed by rules and administrated by rule engines. In the context of the **ReServCom** project, rule driven mechanisms will be developed to steer the process of service composition in terms of four broad composition phases spanning *abstract definition, scheduling, construction, execution,* and *evolution*.

Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE'03)
0-7695-1999-7/03 $17.00 © 2003 **IEEE**

2. Service Composition and Business Rules

Generally speaking web service composition falls under three major categories:

- *Explorative composition:* This category requires that service compositions are generated on the fly on the basis of a request expressed by a client (application developer). The client specifies the desired service functionality in a high-level request language and the ensuing services are then compared with potentially matching Universal Definition and Discovery Infrastructure (UDDI) published constituent service specifications.

- *Semi-fixed composition:* Semi-fixed compositions require that the entire service composition is specified statically but the actual service bindings are decided at run time. When a composite service is invoked, the actual composition specification is generated on the basis of a matching between the constituent services that are specified in the composition and potentially available services.

- *Fixed composition:* A fixed composite service requires that its constituent services be synthesized in a fixed (pre-specified) manner. The composition structure and the component services are statically bound. Requests to such composite services are performed by sending sub-requests to its constituent services.

Service composition needs to be flexible in the way that is defined, scheduled, and constructed. It also needs to adapt to changes so that compositions do not need to be re-generated whenever changes occur. To support various types of service composition, we first need to identify the rules and policies that determine the structure of the composition and prescribe alternative services and provider selection. Frequently, these rules are buried in the business application domain, whereas in other cases that can be derived from a user request. For example, in the domain of e-travel, booking a reservation at a hotel entails an availability notification message that is often sent in conjunction with two other messages: a message that communicates the rates that apply to the availability, and a message that communicates the restrictions that apply to the availability and rates. These messages include a complex set of controls that indicate whether the hotel has available inventory, i.e., closed or open for booking. In addition, booking restrictions that apply to each individual rate, such as a minimum length of stay must also be communicated to the booking agent so that the hotel guest is informed of all the regulations that govern their reservation. In other situations, where user input is expected, rules can be specified to drive the selection of alternative services, for example, "always select the cheapest reservation first". To achieve the desired flexibility and adaptability in the process of service composition, we need to separate the rules that govern the composition from the composition specifications. This means developing a rule based mechanism that manages the entire life cycle of service composition in terms of service composition definition in the abstract, scheduling, construction, and execution. Flexibility comes from the fact that the process of service composition is governed and guided by the rule mechanism. This comes in contrast to solutions provided by classical workflow integration practices, where service composition is pre- predetermined and pre-specified, has narrow applicability and is almost impossible to reuse.

The goal of this research project is to develop a rule based driven mechanism that supports all the phases of service composition, including abstract definition, scheduling, construction, and execution. Research activities have to take the following three factors into account:

1. Business rules are difficult to grasp and complicated to derive. This means that we need to have a thorough investigation into business rules, analyze them, and categorize them. Particularly we need to look at rules and constraints including: the constraints on scheduling, the criteria and conditions of task and resource selection, run-time constraints for service execution, time, cost and quality concerns regarding the selection of service providers.

2. Service composition is a complicated process. This means that we need to take a software engineering approach to analyze the phases involved in service compositions, the activities in each phase, and the rules which may impact each phase.

3. Business rules and underling services can change. This means that there is a need to have a change management system that analyzes the changes and brings the existing compositions to the most updated and consistent status with minimal effort.

3 The Approach

This project advocates a phased approach to service composition. The activities in this phased approach are collectively referred to as the service composition life-cycle [11]. The purpose of these activities, or phases, is to first describe services in the abstract and then to generate concrete executable service processes from these abstract specifications. Abstract service descriptions can be either derived from a request specified in a high-level language (explorative compositions) or by a client specified service definition (semi-fixed and fixed compositions). Hence, the service composition life-cycle spans all three modes of service

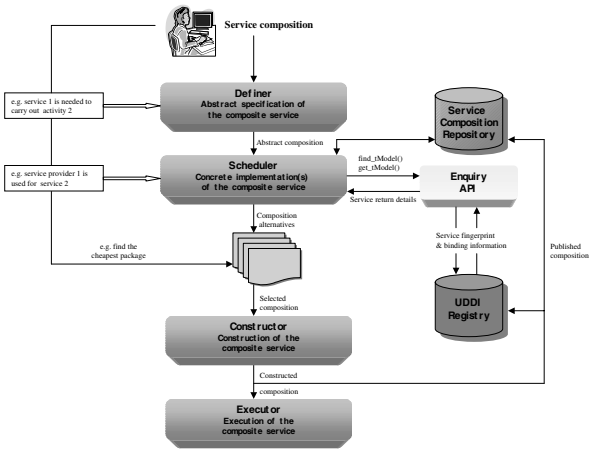

Figure 1. Rules and Service Composition Phases

composition and is characterized by the five phases given below. It must be stressed that this project places emphasis on client specified definitions and thus concentrates on semi-fixed and fixed compositions:

- Definition phase: the definition phase allows defining abstractly composite services. Composite service definitions employ WSDL in conjunction with a language that allows defining business processes by orchestrating web services, viz., BPEL.

- Scheduling phase: the scheduling phase is responsible for determining how and when services will run and preparing them for execution. Its main purpose is to give concrete definitions to the constructs supplied by the definition phase by composing abstract services, by assessing their composability and conformance capabilities, by correlating messages and operations, and then by synchronizing and prioritizing the execution of constituent web services according to their definition.

- Construction phase: The outcome of this phase is the construction of a concrete and unambiguous composition of services – out of a set of desirable or potentially available/matching constituent services – that are ready for execution.

- Execution phase: the execution phase implements composite service bindings on the basis of the scheduled service composition specifications and executes the services in question.

- Evolution phase: the evolution phase will transform existing compositions to the most updated status when the business rules or the constituent services change.

Figure-1 indicates how the service composition life-cycle are related to each other [11].

Based on the composition phases of definition, scheduling, construction, and execution, we can analyze and classify business rules and determine how they impact service composition. Although previous work on business rules such as that described in [10] introduces a simple classification scheme that classifies rules as relationship rules, constraint rules, authorization rules, choice rules, and action rules, these are of general nature and do not consider service composition requirements. Nevertheless, these rules can form a sound basis for extension and application to the service composition life-cycle phases.

There are several ways to design a rule system according to [10]: (1) Bargaining counter, in this case rules are associated with the interfaces of services (and requested services) so that matching and selecting services can be done automatically, (2) Rules as a component/service, in this case rules are decoupled from composition specifications, (3) Rules as specification, in this case rules are embedded in the composition specification. All these approaches have pros and cons in terms of flexibility and adaptability. Another design issue which is orthogonal to the above is whether rules are centrally or decentraly managed. This decision will have an impact on the performance of the overall system. The different design options need to be carefully evaluated and balanced in order to address domain-independent (generic application) service composition requirements.

Service and rule evolution can also be handled according to the phased approach we advocated in this project. First we need to investigate how business rules affect the service composition phases and then service transformation rules that handle the changes and them consistent.

In summary we use a phase approach as part of service composition framework to classify business rules, design rule systems, and manage changes to business rules and services.

4 Related Work

Most of the research work in service composition has focused on using work flows either as a engine for distributed activity coordination or as a tool to model and define service composition. Representative work is described in [4] where the authors discuss the development of a platform specifying and enacting composite services in the context of a workflow engine. The eFlow system provides a number of features that support service specification and management, including a simple composition language, events and exception handling.

The workflow community has recently paid attention to configurable or extensible workflow systems which present some overlaps with the ideas reported in the above. For example, work on flexible workflows has focused on dynamic process modification [7]. In this publication workflow changes are specified by transformation rules composed of a source schema, a destination schema and of conditions. The workflow system checks for parts of the process that are isomorphic with the source schema and replaces them with the destination schema for all instances for which the conditions are satisfied.

The approach described in [6] allows for automatic process adaptation. The authors present a workflow model that contains a placeholder activity, which is an abstract activity replaced at run-time with a concrete activity type. This concrete activity must have the same input and output parameter types as those defined as part of the placeholder. In addition, the model allows to specify a selection policy to indicate which activity should be executed.

The work presented in [5] proposes some interesting ideas in workflow interoperation. It provides infrastructure to support dynamic aspects in planning, scheduling, and execution by introducing workflow schema templates. Reuse of existing workflow schema and templates can be achieved by schema splicing. However how this approach can be used in service composition is not clear.

Work related to web-services and coordination/composability can also be found in CSCW and groupware publications [8]. In this publication the authors examine the potential of using coordination technology to model electronic business activities and illustrate the benefits of such an approach.

The workflow approaches provide some basic mechanisms that can be used for supporting dynamic service co-ordination and composition. However, as the authors pointed out in [2, 3], workflow systems do not cater for the dynamic and distributed nature of service composition for two reasons: (1) a common workflow modeling and management environment is impossible to achieve especially across different enterprises since no WfMS vendor shares the same workflow syntax and semantics; (2) workflow systems do not offer facilities such as changing flow definitions which is a fundamental requirement for service composition. Therefore, these solutions may work only for semi-fixed and fixed compositions, however, they do not work well with explorative composition which requires the service composition structure to be generated on the fly and the composition itself to be changeable. Moreover, they do not support parameterization, reuse, specialization, and nesting of service compositions.

Based on the above arguments [2] proposes the idea of defining B2B protocols for inter-enterprise process execution. B2B protocols expose the public processes while WfMSs implement the private processes of an enterprise. This approach provides an interesting way of binding private and public processes together which lays a foundation for service description, monitoring and contracts. However, it is not clear how these can be used in service composition.

Finally In [3], a Composition Service Definition Language (CSDL) was proposed, which supports dynamic service selection, data mappings and extraction. The Composite Service Engine is very much like a workflow engine.

References

[1] Business Process Execution Language, www-106.ibm.com/developerworks/library/ws-bpel/

[2] C. Bussler, "The Role of B2B Protocols in Inter-Enterprise Process Execution", in Procs. of the 2nd VLDB-TES Workshop, September, Rome, 2001.

[3] F. Casati and Ming-Chien Shan, "Dynamic and Adaptive Composition of e-services", Information Systems, Vol 26(2001), page 143-163, 2001.

[4] F. Casati, S. Ilnicki, L. Jin, V. Krishnamoorthy, M.C. Shan, "Adaptive and Dynamic Service Composition in eFlow", HP Lab. Technical Report, HPL-2000-39.

[5] V. Christophides, R. Hull, A. Kumar, and J. Simeon, "Workflow Mediation using VorteXML", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2000.

[6] D. Georgakopoulos, H. Schuster, D. Baker, and A. Cichocki, "Managing Escalation of Collaboration Processes in Crisis Mitigation Situations", Proceedings of ICDE 2000, San Diego, CA, USA, 2000.

[7] G. Joeris and O. Herzog, "Managing Evolving Workflow Specifications with Schema Versioning and Migration Rules", TZI Technical Report 15, University of Bremen, 1999.

[8] G. A. Papadopoulos and F. Arbab, "Modelling Electronic Commerce Activities Using Control-Driven Coordination, Ninth International Workshop on Database and Expert Systems Applications, Vienna, Austria, August 1998, IEEE Press.

[9] M. Papazoglou, M. Aiello, M. Pistore, and J. Yang, "Planning for Requests against We Services", IEEE Bulletin on Data Engineering, Vol 25(4), December, 2002.

[10] R. Veryard, "Rule Based Development", CBDi Journal, July/August, 2002.

[11] J. Yang and M. Papazoglou, "Service Component for Managing Service Composition Life-Cycle", Information Systems, forth coming.