Chapter 7
Service Composition and Interaction in a SOC Middleware Supporting Separation of Concerns with Flows and Views

Dickson K. W. Chiu
Dickson Computer Systems, Hong Kong

Zhe Shan
City University of Hong Kong, Hong Kong

Qing Li
City University of Hong Kong, Hong Kong

S. C. Cheung
Hong Kong University of Science & Technology, Hong Kong

Patrick C. K. Hung
University of Ontario, Canada

Yu Yang
City University of Hong Kong, Hong Kong

Matthias Farwick
University of Innsbruck, Austria

ABSTRACT

Service-Oriented Computing (SOC) has recently gained attention both within industry and academia; however, its characteristics cannot be easily solved using existing distributed computing technologies. Composition and interaction issues have been the central concerns, because SOC applications are composed of heterogeneous and distributed processes. To tackle the complexity of inter-organizational service integration, the authors propose a methodology to decompose complex process requirements into different types of flows, such as control, data, exception, and security. The subset of each type of flow necessary for the interactions with each partner can be determined in each service. These subsets collectively constitute a process view, based on which interactions can be systematically designed and managed for system integration through service composition. The authors illustrate how the proposed SOC middleware, named FlowEngine, implements and manages these flows with contemporary Web services technologies. An experimental case study in an e-governmental environment further demonstrates how the methodology can facilitate the design of complex inter-organizational processes.

DOI: 10.4018/978-1-4666-2044-5.ch007

Copyright © 2013, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

INTRODUCTION

Recently, Service-Oriented Computing (SOC) (Papazoglou & Georgakopoulos, 2003) has gained a lot of attention within both the industry and academy. SOC is a computing paradigm that utilizes services as fundamental elements, which relies on Service-Oriented Architecture (SOA) (Francisco et al., 2003) to build inter-relating service models. Services-oriented applications share specific characteristics that are distinguishable from traditional distributed applications (Papazoglou & van den Heuvel, 2007). Firstly, they are technology neutral: they interact through standardized protocols in a highly heterogeneous environment, thus offering a broader choice for integration and are loosely coupled. Also, they declaratively define their functionalities, capabilities, and quality-of-service (QoS) requirements in a repository (Curbera et al., 2005) in order to enable dynamic and automated service discovery. Finally, service-oriented applications are created as compositions of services, which benefit from features like abstract modeling and adaptable monitoring.

However, the SOC characteristics stated above cannot be easily solved by using existing distributed computing technologies. Middleware technology provides a common interface for applications to interact with each other as well as their runtime environments (Britton & Bye, 2004). Through such an interface, applications are relieved from handling tedious tasks of resource management, event monitoring, distributed transaction services, and so on. Typically, middleware is implemented to adapt to environmental fluctuations (such as network connections, memory availability, thread pools) and to provide a standard for interoperation. As a result, the applications such developed are less system dependent and therefore more interoperable. This is a particularly essential feature for the composition of service-centric applications as well as the integration of the emerging SOA standards and legacy services. Based on this, the main challenge in service composition reduces to the interaction of services and the associated requirement elicitation. Interaction refers to the interoperation and the integration with applications both internal and external to the systems. This has been a central concern because these applications are composed of heterogeneous and distributed processes. The problem is prominent in service composition, where interactions are complex and can vary across different types of organizations. Business process modeling languages like BPMN (White, 2008) try to capture all relevant aspect of composed applications in one view, leading to complex models that are hard to understand. To tackle this problem, first of all, we need a sophisticated middleware to implement such interactions and to facilitate their management. In addition, because of the complexity, ad hoc use of the middleware is inadequate to solve the problem. A comprehensive methodology is required to make good use of the middleware for a systematic design of service composition and the management of interactions among services.

In this paper, we propose to use both the paradigm of *views* and *flows* to tackle the complexity of service composition and interaction in a SOC middleware. Our approach is based on the principle of separation of concerns, which is often considered as one of the key principles in software engineering. Process (workflow) view (Chiu, Karlapalem, & Li, 2001; Liu & Shen, 2003) has been proposed as a novel approach to address business-to-business interoperation. A process view is a structurally correct subset of a process (Chiu et al., 2004). The use of process views facilitates sophisticated interactions among services and allows these interactions to interoperate in a gray-box mode (that is, they can access each other's internal information to some pre-defined extent). Each view can be regarded as a separate concern based on some specific client requirements of a service scenario. Therefore, the artifact of process views is a handy mechanism to enact and enforce cross-organizational process interoperability for service composition.

30 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:
www.igi-global.com/chapter/service-composition-interaction-soc-middleware/74393

Related Content

Database High Availability: An Extended Survey
Moh'd A. Radaidehand Hayder Al-Ameed (2009). *Selected Readings on Database Technologies and Applications (pp. 21-43).*
www.irma-international.org/chapter/database-high-availability/28571

ONTOMETRIC: A Method to Choose the Appropriate Ontology
Adolfo Lozano-Telloand Asunción Gomez-Perez (2004). *Journal of Database Management (pp. 1-18).*
www.irma-international.org/article/ontometric-method-choose-appropriate-ontology/3308

A Tool for Fuzzy Reasoning and Querying
Geraldo Xexéoand André Braga (2008). *Handbook of Research on Fuzzy Information Processing in Databases (pp. 381-406).*
www.irma-international.org/chapter/tool-fuzzy-reasoning-querying/20361

The Expert's Opinion
Journal of Database Management (1992). *Journal of Database Administration (pp. 30-32).*
www.irma-international.org/article/expert-opinion/51108

Mutual Recognition Mechanism Based on DVCS Oracle in the Blockchain Platform: DVCS Oracle in the Global Supply Chain
Vladimir Nikolaevich Kustovand Ekaterina Sergeevna Selanteva (2022). *Utilizing Blockchain Technologies in Manufacturing and Logistics Management (pp. 81-103).*
www.irma-international.org/chapter/mutual-recognition-mechanism-based-on-dvcs-oracle-in-the-blockchain-platform/297159