

Chapter 15

Identifying Services in Procedural Programs for Migrating Legacy System to Service Oriented Architecture

Masahide Nakamur
Kobe University, Japan

Hiroshi Igaki
Tokyo University of Technology, Japan

Takahiro Kimura
Nihon Unisys Ltd., Japan

Kenichi Matsumoto
NAIST, Japan

ABSTRACT

In order to support legacy migration to the service-oriented architecture (SOA), this paper presents a pragmatic method that derives candidates of services from procedural programs. In the SOA, every service is supposed to be a process (procedure) with (1) open interface, (2) self-containedness, and (3) coarse granularity for business. Such services are identified from the source code and its data flow diagram (DFD), by analyzing data and control dependencies among processes. Specifically, first the DFD must be obtained with reverse-engineering techniques. For each layer of the DFD, every data flow is classified into three categories. Using the data category and control among procedures, four types of dependency are categorized. Finally, six rules are applied that aggregate mutually dependent processes and extract them as a service. A case study with a liquor shop inventory control system extracts service candidates with various granularities.

DOI: 10.4018/978-1-4666-2649-2.ch015

INTRODUCTION

Due to rapid changes in business environment, enterprise software systems are required to be more agile and flexible to keep up with the changes. However, most enterprise systems have been built upon a highly proprietary and monolithic architecture, without considering interoperability among other systems. Such monolithic systems are usually fragile for the changes. A simple update of a business process may result in huge cost for updating the system.

The *service-oriented architecture (SOA)* (Erl, 2007; Newcomer & Lomow, 2004; Papazoglow & Georgakopoulos, 2003) is an architecture paradigm to cope with the problem. In the SOA, features of a system are exhibited as self-contained *services*, corresponding to elementary business units. A service has open interface that encapsulates implementation-specific logic and data. A business process can be rapidly created or modified by assembling the existing services, where the services are *loosely coupled*. Thus, the SOA is believed to make the system robust for the business changes.

To receive the benefit of the SOA within the existing assets, the *legacy migration to SOA* is now a great concern (Cetin, Altintas, Oguztuzun, Dogru, Tufekci, & Suloglu, 2007; Lewis, Morris, Smith, & Simanta, 2008). Most of the conventional SOA development frameworks e.g., SOMA (Arsanjani, Ghosh, Allam, Abdollah, Gariapathy, & Holley, 2008), SOMF (Bell, 2008), and BMM (Berkem, 2008) adopt a top-down approach, which starts with the business process analysis, identifies elementary processes, and implements them as services. Since the system is designed optimally for the SOA, the top-down approach is well applied to development of brand-new systems. However, it does not consider much how to reuse the existing legacy system.

To support the SOA legacy migration effectively, this paper presents a pragmatic method that extracts candidates of SOA services from procedural programs. In the proposed method, we extensively analyze *dependencies* among processes (i.e., procedures) in the source code. Each service is derived as an aggregation of mutually-dependent processes, so that the service has open-interface, self-containedness and coarse granularity for the business.

For implementing the method, we first obtain the *Data Flow Diagram (DFD)* (DeMarco, 1979) by applying reverse-engineering techniques to the given source code (O'Hare & Troan, 1994; Benedusi, Cimitile, & Carlini, 1989). We then classify every data flow in the DFD into three categories (*external*, *system* and *module*). Using the data category, we identify four types of dependency (*system data*, *module data*, *transaction* and *condition*) between processes. Finally, we aggregate mutually-dependent processes as self-contained services, which is systematically performed by the proposed six rules.

To evaluate the proposed method, we have conducted two kinds of experiments with a liquor shop inventory control system. The experimental results show that reasonable service candidates with various granularities are successfully extracted from the source code. We also investigate the derived services through the comparison with the classical software metrics: *cohesion* and *coupling* metrics (Al-Ghamdi, Shafique, Al-Nasser, & Al-Zubaidi, 2001; Lakhotia, 1993; Yourdon & Constantine, 1979).

The paper is organized as follows. In the next section, we briefly survey the service oriented architecture and the problem to be tackled. We then present the proposed service extraction method. We conduct the experiment with the inventory control. Next, we evaluate the proposed method with several related work and finally, we conclude the paper.

17 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/identifying-services-procedural-programs-migrating/72553

Related Content

Evidential Reasoning-Based Evaluation of Services

Van Nam Huynh, Minh Chau Doan, Michitaka Kosaka and Yoshiteru Nakamori (2014). *Progressive Trends in Knowledge and System-Based Science for Service Innovation* (pp. 191-204).

www.irma-international.org/chapter/evidential-reasoning-based-evaluation-of-services/87918

Service Composition Based Software Solution Design: A Case Study in Automobile Supply Chain

Tong Mo, Jingmin Xu, Zhongjie Wang, Yufei Ma, Heyuan Huang, Yuan Wang, Ying Liu, Jun Zhu and Xiaofei Xu (2012). *Technological Applications and Advancements in Service Science, Management, and Engineering* (pp. 103-115).

www.irma-international.org/chapter/service-composition-based-software-solution/66288

Identification, Specification, and Development of Web-Oriented Architectures

Gunnar Thies and Gottfried Vossen (2012). *International Journal of Information Systems in the Service Sector* (pp. 1-21).

www.irma-international.org/article/identification-specification-development-web-oriented/62247

Service Science, Management, Engineering, and Design (SSMED): An Emerging Discipline - Outline & References1

Jim Spohrer and Stephen K. Kwan (2011). *Information Systems and New Applications in the Service Sector: Models and Methods* (pp. 194-226).

www.irma-international.org/chapter/service-science-management-engineering-design/50237

Service Management of Special Care Units: Lessons Learned in Manufacturing

Gad Vitner, Shirly Bar-Lev, Erez Nadir, Michael Feldman and Shmuel Yurman (2011). *International Journal of Information Systems in the Service Sector* (pp. 39-51).

www.irma-international.org/article/service-management-special-care-units/50566