Chapter X

Service Patterns for Enterprise Information Systems

Constantinos Constantinides
Concordia University, Canada

George Roussos
University of London, UK

Abstract

This chapter introduces service patterns for SOA-based enterprise systems. The authors believe that the deployment of such patterns would be of considerable value both as a best-practice guide for practitioners as well as a starting point for further research in their role in software engineering. A comprehensive catalog of service patterns is included in this chapter. In the catalog, each pattern is discussed in the context of selected examples and in terms of a brief description of its role, functionality, and deployment. For each pattern there are recommendations on implementation and a practical usage scenario.

Introduction

Modern enterprise information systems constitute a core component in business support. Not only must they provide reliable infrastructures for the organization itself but they also must be capable of seamlessly connecting to the systems of other businesses in their value added network of partnerships. These operating conditions
demand that enterprise systems should operate transparently and should be flexible. Over the past few years, service-oriented architectures (SOA) have emerged as a framework that addresses this requirement both effectively and efficiently. Furthermore, time after time systems developers discover that successful and cost-effective design of information architectures requires a combination of theory and practical experience as well as reuse of robust and proven designs that form solutions to frequently occurring problems. Effective reuse can speed up the development process, reduce costs, increase productivity, and improve the quality of software.

**Design-Level Reuse**

As software products need to satisfy both technical and nontechnical criteria, developers find it essential to combine theory and experience in order to reuse proven designs. The importance of reuse lies on the fact that it can speed up the development process, cut down costs, increase productivity, and improve the quality of software. Design-level reuse is viewed as the attempt to share certain aspects of an approach across various projects. There is, however, no single approach that can support reuse across all architectural layers. Object- and component-based systems offer a wide spectrum of techniques to reuse designs on different levels, ranging from system architectures, frameworks, and patterns to libraries and programming languages. Developers can view reuse on a range of different levels of granularity, ranging from effective consistency sharing in low-level programming to sharing subsystem architectures or overall structure. Libraries and languages constitute an approach on how to best program *in the small*. Patterns, on the other hand, fall in the part of the spectrum that supports the sharing of interaction architectures, and they constitute an approach on how to best program *in the large* (Szyperski, 2002).

**Patterns**

The notion of a pattern originates from the discipline of Architecture from the book by Alexander et al. (1977), and it refers to solutions of occurring problems in the constructions of towns and buildings. Along the same lines, a pattern in Software Engineering refers to a solution to a recurring problem that arises during software development that can be used in different contexts. This definition leaves space for overloading and the term, indeed, tends to be overloaded in the literature, as patterns are becoming available over a wide spectrum of granularity, ranging from very general design principles to language-specific idioms.

- **Design patterns:** Design patterns are micro-architectures that describe the abstract interaction between objects collaborating to solve a particular problem. They have become popular through the seminal book by Gamma et al. (1995), also referred to as the Gang of Four (GoF). In Gamma et al. (1995), each pattern is discussed in terms of purpose (what the pattern does) and scope (whether the pattern applies primarily
Related Content

OntoFrame: An Ontological Framework for Method Engineering
[www.igi-global.com/chapter/ontoframe-ontological-framework-method-engineering/23788?camid=4v1a](www.igi-global.com/chapter/ontoframe-ontological-framework-method-engineering/23788?camid=4v1a)

Music Melodies Suited to Multiple Users' Feelings Composed by Asynchronous Distributed Interactive Genetic Algorithm

Asking Questions: Applying Survey Techniques in Building Successful Enterprise
[www.igi-global.com/article/asking-questions/201207?camid=4v1a](www.igi-global.com/article/asking-questions/201207?camid=4v1a)

Matilda: A Generic and Tailorable Framework for Direct Model Execution in Model-Driven Software Development
[www.igi-global.com/chapter/matilda-generic-tailorable-framework-direct/37036?camid=4v1a](www.igi-global.com/chapter/matilda-generic-tailorable-framework-direct/37036?camid=4v1a)