Chapter 95
A Model-Driven Approach to Service Composition with Security Properties

Stéphanie Chollet
Laboratoire d'Informatique de Grenoble, France

Philippe Lalanda
Laboratoire d'Informatique de Grenoble, France

ABSTRACT

The software engineering community is striving to handle a significant number of new critical demands. Productivity, quality, and runtime flexibility are only few of them. To satisfy such requirements, new development paradigms are regularly proposed. Service-Oriented Computing (SOC) is one of them. SOC is based on the notion of services, which are well-defined composition units, to support rapid application development. This chapter presents a brief state of the art of services. Although SOC brings properties of major interest, it suffers from usual limitations of reused-based approach. In particular, service composition is much more complicated than often pretended. In this chapter, we propose to present a model-driven approach to service composition dealing with non-functional aspects, including security.

INTRODUCTION

The need for advanced development technologies and tools providing appropriate support in the software industry is greater than ever. We believe that this is essentially due to the diversity of software developments undergone nowadays and to an ever increasing pressure. Software systems pervade every aspects of our life and, subsequently, place very different and demanding requirements on the development projects. In some domains, unprecedented pressure is put on cost and time-to-market. Time-to-market in the mobile phone industry for instance has dropped down to six months. In other domains, the main difficulty resides in new, stringent non functional requirements. For instance, pervasive applications are characterized by complex requirements in terms of dynamism,

DOI: 10.4018/978-1-4666-4301-7.ch095

context awareness and autonomy. New development paradigms have been proposed to face these demanding conditions. For instance, approaches like component-based software engineering, aspect-oriented programming or service-oriented computing have been recently defined to facilitate the development of modular, dynamic applications.

It however turns out that the community is striving to use efficiently these new technologies and to deliver the expected level of software quality. We believe that an effective approach to improve productivity and quality is to hide technical complexity through the use of generative programming techniques. The purpose of generative programming is to enable the production of software through the automatic generation of applications from specifications written in a Domain-Specific Language (DSL). A DSL offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain. It allows developers to manipulate familiar domain concepts at the different phases of software production. Although, tools based on DSL are naturally expected by the industry, very few of them are actually delivered. One of the main blocking factors is the cost of producing such tools that must incorporate many software engineering practices in order to cover the whole software lifecycle. Also, in order to be really useful, a tool has to be dedicated to the domain it deals with. Unfortunately, developing a tool in a narrow domain is generally not cost effective and very few companies can afford such an investment. Most companies then rely on generic tools that are getting unfitted to modern software.

This chapter presents a model-based approach for the composition of secured services. The purpose of this approach is to generate part of the code related to security, which is complex and error-prone, from domain-specific models. It is based on the separation of concerns principle in the sense that different models, and associated meta-models, are used to specify different aspects of the service composition.

It is structured as it follows. First, we define the Service-Oriented Computing (SOC), the service composition and the service technologies used in different domains such as application integration, pervasive environment… The second section deals with the security in SOC, particularly the security between the service consumer and the service provider. We detail the security problems, solutions and technologies adapted to the service domain. The third section presents a model-driven approach to facilitate the composition of heterogeneous and secured services. The last section presents the results of experiments applied to an implementation of the model-driven approach.

SERVICES

Service Definition

Service-Oriented Computing (SOC) is a reuse-based approach for the development of modern software applications. The key concept of this approach is the notion of service. Several definitions have actually been proposed for software services. Let us review the major ones.

First, Papazoglou (2003) proposed the following definition:

Services are self-describing, platform agnostic computational elements.

A service is then a software composition unit defined through an explicit description. A service can be discovered, selected and used by a consumer based upon this description. Service consumers are usually unaware of the service implementation technology or of the underlying runtime platform. Similarly, a service does not know the runtime contexts in which it is used. This double independence is a major property of the service-oriented approach, as it facilitates loose-coupling at the application level. This definition is however limited in the sense that it presumes that a service