Chapter 26
Building Elastic Java Application Services Seamlessly in the Cloud:
A Middleware Framework

Rostyslav Zabolotnyi
Vienna University of Technology, Austria

Philipp Leitner
University of Zurich, Switzerland

Schahram Dustdar
Vienna University of Technology, Austria

ABSTRACT

Cloud computing is gaining increasing attention from the industry and research; however, there is a lack of advanced Cloud software development tools. While Platform as a Service (PaaS) brings convenient software development platform for application development, it often comes with limitations in terms of application architecture functionality and requires provider lock-in. The Infrastructure as a Service (IaaS) model may sound like a solution to these problems by enabling application development freedom; however, it necessitates operation at the lower level of virtual machines and snapshots. In this chapter, the authors present CloudScale: a low-overhead middleware framework that migrates Java applications seamlessly to the Cloud with minimal changes in the application code. They focus on the main ideas behind CloudScale and its influence on solving Cloud software development and deployment problems with minimal overhead and Cloud-awareness required from developers.

DOI: 10.4018/978-1-4666-6178-3.ch026

INTRODUCTION

In the past few years, the advancement of Cloud computing (Mell & Grance, 2011) has transformed the IT industry, and has given new opportunities and abilities to developers and users. Moreover, Cloud computing simplifies the implementation of innovative ideas for small companies or individuals, and lowers production and maintenance costs for industrial applications (Armbrust et al., 2010). Applications designed with the Cloud in mind (so-called Cloud-native applications) allow developers to elastically adapt to market changes and optimize resource consumption.

Developers can adapt to the Cloud computing model at different levels. The most basic Cloud service model is the *IaaS* (Infrastructure as a Service) (Mell & Grance, 2011) approach. At this level, Cloud service providers offer virtualized resources with requested configuration and operating system (usually in the form of hard drive images and virtual machines) to satisfy application computation requirements (Bhardwaj, Jain, & Jain, 2010). For many use cases, this layer is preferable to *PaaS* (Platform as a Service), as it gives more freedom to the developers, requires less migration effort, and is better standardized than PaaS.

However, building IaaS-based *elastic Cloud applications* is not an easy task, and requires developers to face an entirely new range of challenges. For instance, developers have to introduce a significant amount of *platform-dependent boilerplate code* that allows them to control virtual machines rented from the Cloud, monitor the state of virtual machines, and elastically scale applications up and down according to the current or future load. These tasks are orthogonal to the mission of the applications, introduce significant complications, and bury the application's actual business logic deep under a mountain of platform-dependent code. This boilerplate code has to be developed over and over again for each new application or platform version. In addition, this platform-dependent integration code not only slows down application development, but also causes vendor lock-in, as each Cloud service provider has its own API that developers have to use to be able to interact with the Cloud.

In this chapter, we describe how the *CloudScale*[1] research prototype (Leitner, Satzger, Hummer, Inzinger, & Dustdar, 2012) solves the challenges described above. The CloudScale framework allows developers to declare application Cloud scaling and interaction rules declaratively with the help of Java annotations, thus allowing developers to focus on the business logic of the application. The CloudScale framework injects the IaaS platform-dependent code necessary to scale the application over the Cloud via bytecode manipulation. This approach significantly simplifies the development of Java-based Cloud-native applications and avoids vendor lock-in, as the Cloud-specific code is separated from the application business logic and can be easily changed.

The content of this chapter is structured as follows. In the next section, we provide some background information on the current state-of-the-art research and development attempts to simplify Cloud software development without losing control over the available equipment and infrastructure resources. Next, we introduce our illustrative example application and describe the CloudScale architecture, as well as main design decisions and limitations by presenting the research idea and application development process that enables the development of Cloud applications transparently and seamlessly, without thinking about platform-dependent code and Cloud performance monitoring.