COMPONENT-BASED SIMULATION FRAMEWORK FOR COMPONENT TESTING USING SPRINGDM

POTUZAK, T[omas]; SNAJBERK, J[aroslav]; LIPKA, R[ichard] & BRADA, P[remysl]

Abstract: In this paper, a proposal for component-based simulation framework for testing of real software components in simulated environment is presented. Using this framework, the real software components can be tested directly, without necessity for creation of a simulation model of the real components. The main issues of this framework are presented in the paper and possible solutions are discussed.

Key words: Software, components, simulation, framework, testing

1. INTRODUCTION

Component-based software development is an important trend in software engineering in last decade. Using this approach, a system can be constructed as a set of individual components (i.e. pieces of software with well-defined functionality). Software components are designed to maximize reusability and third party composition. Each software component is a blackbox entity, which provides concrete predefined functionality (Szyoperski, 2002). Hence, the testing of the components is an important branch in contemporary research.

The components are usually not tested for their functionality, since functionality according to the specification is considered to be implied. Instead, the simulation tests are often focused on extra-functional properties and quality of services. Examples can be found in (Becker et al., 2009) and (Heam et al., 2010). Nevertheless, it is common to use only models of the real components in simulation tests (Cansado et al., 2010).

In this paper, we present a proposal for simulation framework, which will enable testing of real components in a simulation environment. This means that the software components will be tested directly and there will be no need for creation of some component’s models for the simulation. In order to maximize generality and modularity of the simulation, the framework itself will not be a monolithic application, but rather will be constructed from components as well.

2. COMPONENT MODELS AND FRAMEWORKS

Component models specify how software components look, behave, and interact. A component framework is an implementation of a concrete component model. There are many component models and even more component frameworks used and developed in industry and in research sphere. Three example frameworks are briefly described in following subsections.

2.1 OSGi

The OSGi framework implements a dynamic component model and offers service platform for Java programming language. OSGi provides an environment, where component-based applications and components alone can be remotely installed, started, stopped, updated, and uninstalled without requiring a reboot (Rubio, 2009).

The OSGi framework is commonly used in many different industry areas such as vehicles, mobile phones, PDAs, application servers, and so on (Rubio, 2009).

2.2 Spring

The Spring framework offers variety of features to support development of enterprise-grade applications. The most characteristic features of Spring framework are the inversion of control, aspect-oriented programming, data access, transaction management, and remote access (Rubio, 2009).

2.3 SpringDM

The SpringDM (Spring Dynamic Modules) for OSGi service platform makes it possible to develop OSGi components using Spring framework. Moreover, SpringDM improves manageability of OSGi services (Rubio, 2009). As such, it is an ideal candidate as the basic of our component-based simulation framework.

3. SIMULATION FRAMEWORK

As it was said, the goal of our research is to create a component-based framework for simulation testing of real software components. The simulation framework will be created using SpringDM component framework. There are several main issues, which are described in following sections.

3.1 Combining of Simulated and Real Components

There will be three main types of components within the scope of resulting simulation framework.

First type is represented by the components, which form the simulation framework and ensure its functionality. Among these components, the calendar is most important. This component incorporates all events of the simulation based on loaded scenario and controls its progress. Of course, there must be other components, which ensure for example logging, visualization of the simulation, and so on.

Second type is represented by the real components of some component-based application, which shall be tested in the simulation framework. There can be one or more real components, which can be tested simultaneously.

Generally, the tested real components provide some defined services, but they also may require services from other components in order to be able to work. In order to provide services, which are required by the tested real components, there is the third type of components – the simulated components. Together with the components of the framework (i.e. the first type), these components form an environment, in which the real component are running. All three types of components can be seen in Fig. 1.

The real components must not be aware that they are in simulated environment. Hence, the simulated environment (formed by first and second type components) must act like the real environment of the tested real components. On the other hand, both the simulated environment and the tested real components must be under complete control of the framework. Hence, all interactions of the real and simulated components (i.e. method invocations) must be performed using calendar and events. This enables to tests extra-functional properties such as method response time and also easy logging of all interactions.
Another option the same problem connection method. For example, the simulation framework is unaffected by the changes in the Java API methods. The replacement of the imports can be performed either by direct manipulation of the components’ bytecode or using the AOP (Aspect Oriented Programming), which is provided by the SpringDM. The solution will be selected during the implementation of the simulation framework.

3.3 Case study

As a case study, we selected an application for mobile phones and PDAs. The application consists of five components (see Fig. 1). There is a GPS component, which provides GPS coordinates, Accelerometer component, which provides information about the steps performed by the user. The Statistics calculator component calculates various health statistics, the Google maps component connects to internet and provides maps of the user’s current location and finally, the Representer component displays all statistics and maps to user.

The components of the described application will be used for testing in the simulation framework. The GPS and Accelerometer components will be simulated and the remaining three components will be used as real tested components (see Fig. 1). The described configuration allows us to verify all part of the simulation framework’s functionality.

4. CONCLUSION AND FUTURE WORK

In this paper, we presented a proposal for component-based simulation framework for testing of real software components in simulated environment. Using this framework, the real software components can be tested directly, without necessity for creation of a simulation model of the real component. The main issues of this framework are presented in the paper and possible solutions are discussed.

In our future work, we will focus on implementation of the proposed framework. The completed framework will be used for testing of functional and extra-functional features of the software components.

5. ACKNOWLEDGEMENT

This work was supported by the Grant Agency of the Czech Republic under grant 201/08/0266.

6. REFERENCES


