Chapter 7
Automating Integration Testing of Large-Scale Publish/Subscribe Systems

Éric Piel
Delft University of Technology, The Netherlands

Alberto González
Delft University of Technology, The Netherlands

Hans-Gerhard Gross
Delft University of Technology, The Netherlands

ABSTRACT

Publish/subscribe systems are event-based systems separated into several components which publish and subscribe events that correspond to data types. Testing each component individually is not sufficient for testing the whole system; it also requires testing the integration of those components together. In this chapter, first we identify the specificities and difficulties of integration testing of publish/subscribe systems. Afterwards, two different and complementary techniques to test the integration are presented. One is based on the random generation of a high number of event sequences and on generic oracles, in order to find a malfunctioning state of the system. The second one uses a limited number of predefined data-flows which must respect a precise behaviour, implementable with the same mechanism as unit-testing. As event-based systems are well fitted for runtime modification, the particularities of runtime testing are also introduced, and the usage in the context of integration testing is detailed. A case study presents an example of integration testing on a small system inspired by the systems used in the maritime safety and security domain.

INTRODUCTION

Our research focuses on the fast integration of systems-of-systems (SoS). In collaboration with industrial partners, we ensure that the results can be applied for Maritime Safety and Security (MSS) systems, which are typical event-based systems and are often built on top of a publish/subscribe architecture. An important aspect of this research covers the validation through checking the cor-
rect integration of the system in order to ensure their reliability.

MSS systems are large-scale distributed SoS in which the sub-components are elaborate and complex systems in their own right. The primary tasks of MSS SoS are sensing issues at sea, analysing these issues, thus, forming a situational awareness, and initiating appropriate action, in case of serious issues (EU Commission, 2007; Thales Group, 2007; Lockheed Martin, 2008; Embedded Systems Institute, 2007). Provision of situational awareness requires analysis and synthesis of huge volumes of data coming from the various types of sensor components such as vessel-tracking (AIS) systems, satellite monitoring, radar systems, or sonar systems. A distinguishing characteristic of MSS systems-of-systems is their data-centric, distributed and event-based nature (Muhl, Fiege, & Pietzuch, 2006), which means that the publish/subscribe paradigm is well fitted, and it is readily being used as underlying system infrastructure.

In their landmark paper on “The Many Faces of Publish/Subscribe”, Eugster et al. (2003) present an array of advantages of the publish/subscribe paradigm as a cure for developing highly dynamic large-scale systems. However, the advantages of fully decoupling the communicating entities in publish/subscribe platforms in terms of time, space and synchronization, can also be seen as a curse when it comes to runtime evolution and ensuring a system’s integrity after dynamic updates. In this chapter, we will discuss the issues of integration and acceptance testing arising from the loose coupling of communicating entities advocated by event-based systems. Because we are dealing with highly dynamic systems that have to provide constant operational readiness, we will concentrate on the challenges system engineers are facing when it comes to testing and accepting dynamic system reconfigurations.

First in section Properties of Publish/Subscribe Platforms, we will present the testing requirements in publish/subscribe systems and highlight the specific challenges of this context. In Section Integration Testing in Event-based Systems, we will describe the usage of testing methods in order to implement integration testing in such dynamic and decoupled environment. The Section Runtime Testing for Component-based Platforms provides details on the specificities of handling testing at runtime. An example of usage of the methods previously defined will then be presented in Section Runtime Integration Testing: A Controlled Experiment, using a simplified version of an MSS system. An overview of the future research to come in the domain of testing and runtime evolution will be given in Section Future Trends. Finally, Section Summary and Conclusion will summarize and conclude this chapter.

**PROPERTIES OF PUBLISH/Subscribe PLATFORMS**

The quality of a software system is a compromise between the cost of an error happening and the cost of improving the quality (cost being used with a large meaning such as time, money or physical damage). In order to improve the quality of a software system, the most commonly used technique is software testing. Due to the nature of their applications, systems based on event-driven platforms, and publish/subscribe platforms in particular, tend to have high needs for software quality and testing. Systems designed on event-driven platforms can have all kinds of applications, but some typical classes of applications are:

- Embedded software, which controls physical devices such as cars, music players, factory robots, etc. Sensors generate input data and events that are treated by software components which, in turn, send output data to controller devices.
- Web applications, which react to the requests from web users, process them and send the results.