Chapter 8.11
Validation and Verification of Software Systems Using Virtual Reality and Coloured Petri Nets

Hyggo Oliveira de Almeida
Federal University of Campina Grande, Brazil

Leandro Silva
Federal University of Campina Grande, Brazil

Glauber Ferreira
Federal University of Campina Grande, Brazil

Emerson Loureiro
Federal University of Campina Grande, Brazil

Angelo Perkusich
Federal University of Campina Grande, Brazil

ABSTRACT

Validation and verification techniques have been identified as suitable mechanisms to determine if the software meets the needs of the user and to verify if the software works correctly. However, the existing verification techniques do not support friendly visualization. Also, validation techniques with friendly visualization mechanisms do not allow the verification of the system's correctness. In this chapter, we present a method for the validation and verification of software systems through the integration of formal methods and virtual reality. Furthermore, a software tool associated with such a method is also described along with an embedded system case study.

INTRODUCTION

The complexity of software systems is increasing and, consequently, making it more difficult and necessary to determine if they work correctly. In the context of the currently applied development techniques and processes, tests are commonly used for validating software, but they cannot ensure that the software is in accordance with important behavioral properties, such as robustness or safety requirements.

Verification techniques aim to detect and aid the designer to correct mistakes during the software development, being useful when defining whether the software satisfies its requirements and specifications. Through formal modeling and verification methods, it is possible to determine if the system works correctly while considering all possible behaviors.

On the other hand, validation techniques determine if the software meets the needs of the user (Fuhrman, Djlive, & Palza, 2003). Thus, a graphical and friendly visualization of the system model is very useful in validating software in different domains. However, even though most of the formal modeling techniques have their own graphical representations, such as automata and Petri nets, they do not allow a friendly visualization and, subsequently, validation of the system's behavior. Without such friendly visualization, the final user needs to understand formal modeling concepts, which are not easily understood by nonengineers.

Coloured Petri nets (CPN) (Jensen, 1992) are used to model and verify the correctness of software systems, and virtual reality modeling language (VRML) (International Organization for Standardization, 1998) is applied to validate the software behavior considering the user acceptance.

We will present a software platform for managing the integration of CPN and VRML, easing the application of the proposed method. In order to illustrate the use of the method and the platform, an embedded software case study is presented. Finally, related approaches and concluding remarks are discussed.

Using the proposed method, it is possible to separate the formal verification and the validation activities of the system, allowing the verification of its correctness while still giving a friendly visualization for the final user. Moreover, the proposed tool aids the developers in integrating the CPN formal and the friendly VRML models, hiding the integration complexity.

BACKGROUND

As we have said in the preceding section, the verification and validation phases of the approach we present here are based on coloured Petri nets and the VRML language. Therefore, in order to provide a better understanding of the next sections, it is desirable to present some background information.

Coloured Petri Nets

Petri nets are a formal method with a graphical representation used to model concurrent systems. With Petri nets, it is possible to specify and verify properties like precedence relation and deadlocks, among others. The graphical representation of a Petri net is a bipartite graph composed of places and transitions. Arcs connect places to transitions and transitions to places but never places to places or transitions to transitions. Each place can have zero or several tokens at a given moment. This is called the marking of the place. Therefore, the marking of a Petri net model is the set of markings of all places at a given moment. The transitions represent the actions that can take place.

Different extensions to the Petri nets formalism exist, such as timed Petri nets (Wang, 1998) and coloured Petri nets. In the context of this chap-