User Interface Prototyping Based on UML Scenarios and High-Level Petri Nets

Mohammed Elkoutbi and Rudolf K. Keller

Département d’informatique et de recherche opérationnelle
Université de Montréal
C.P. 6128, succursale Centre-ville, Montréal, Québec H3C 3J7, Canada
voice: (514) 343-6782
department: (514) 343-5834
e-mail: {elkoutbi, keller}@iro.umontreal.ca
http://www.iro.umontreal.ca/~{elkoutbi, keller}

Abstract: In this paper, we suggest a requirement engineering process that generates a user interface prototype from scenarios and yields a formal specification of the system in form of a high-level Petri net. Scenarios are acquired in the form of sequence diagrams as defined by the Unified Modeling Language (UML), and are enriched with user interface information. These diagrams are transformed into Petri net specifications and merged to obtain a global Petri net specification capturing the behavior of the entire system. From the global specification, a user interface prototype is generated and embedded in a user interface builder environment for further refinement. Based on end user feedback, the input scenarios and the user interface prototype may be iteratively refined. The result of the overall process is a specification consisting of a global Petri net, together with the generated and refined prototype of the user interface.

Keywords: User interface prototyping, scenario specification, high-level Petri net, Unified Modeling Language.

1 Introduction

Scenarios have been identified as an effective means for understanding requirements [16] and for analyzing human computer interaction [14]. A typical process for requirement engineering based on scenarios [7] has two main tasks. The first task consists of generating from scenarios specifications that describe system behavior. The second task concerns scenario validation with users by simulation and prototyping. These tasks remain tedious activities as long as they are not supported by automated tools.

For the purpose of validation in early development stages, rapid prototyping tools are commonly and widely used. Recently, many advances have been made in user
interface (UI) prototyping tools like UI builders and UI management systems. Yet, the development of UIs is still time-consuming, since every UI object has to be created and laid out explicitly. Also, specifications of dialogue controls must be added by programming (for UI builders) or via a specialized language (for UI management systems).

This paper suggests an approach for requirements engineering that is based on the Unified Modeling Language (UML) and high-level Petri nets. The approach provides an iterative, four-step process with limited manual intervention for deriving a prototype of the UI from scenarios and for generating a formal specification of the system. As a first step in the process, the use case diagram of the system as defined by the UML is elaborated, and for each use case occurring in the diagram, scenarios are acquired in the form of UML sequence diagrams and enriched with UI information. In the second step, the use case diagram and all sequence diagrams are transformed into Colored Petri Nets (CPNs). In step three, the CPNs describing one particular use case are integrated into one single CPN, and the CPNs obtained in this way are linked with the CPN derived from the use case diagram to form a global CPN capturing the behavior of the entire system. Finally, in step four, a prototype of the UI of the system is generated from the global CPN and embedded in a UI builder environment for further refinement.

In our previous work, we have investigated and implemented the generation of UI prototypes from UML scenarios using exclusively the UML, most notably UML Statecharts [5, 13, 21, 22]. In this Statechart-based approach, Statecharts are used to integrate the UML scenarios and capture object and UI behavior. In the work presented in this paper, we decided to take a Petri-net based approach, with CPNs taking the role of UML Statecharts. We opted for Petri nets because of their strong support of concurrency, their ability to capture and simulate multiple copies of scenarios in the same specification, and for the wealth of available tools for analyzing, simulating, and verifying Petri nets. A comparison of the two approaches is provided in Section 5 of the paper.

In our approach, we aim to model separately the use case and the scenario levels. We also want to keep track of scenarios after their integration. Thus, we need a PN class that supports hierarchies as well as colors or objects to distinguish between scenarios in the resulting specification. We adopted Jensen’s definition of CPN [10] which is widely accepted and supported by the designCPN tool [3] for editing, simulating, and verifying CPNs. Object PNs could also have been used, but CPNs are largely sufficient for this work. In our current implementation, UI prototyping is embedded into the Visual Café environment [23] for further refinement.

Section 2 of this paper gives a brief overview of the UML diagrams relevant to our work and introduces a running example. In Section 3, the four activities leading from scenarios to executable UI prototypes are detailed. Section 4 reviews related work, and in Section 5, we discuss a number of issues related to the proposed approach. Section 6 concludes the paper and provides an outlook into future work.

2 Unified Modeling Language

The UML [19], which is emerging as a standard language for object-oriented modeling, provides a syntactic notation to describe all major views of a system using