A Study of Developing Virtual Prototyping by Using JavaBean Interface Tool and SystemC Engine*

Husni Teja Sukmana, Jeong B. Lee, Jong Il Kim, Young J. Jung, Jin B.Kwon, Kee W. Rim, and Young R. Lee

Sun Moon University, Dept. of Computer Science,
Asan, Chungnam, 336-708, South Korea
husniteja@yahoo.com, jblee@sunmoon.ac.kr, rumpet0@empal.com, yjjung.kr@gmail.com, jbkwon@sunmoon.ac.kr, rim@sunmoon.ac.kr, yrlee@sunmoon.ac.kr

Abstract. SystemC is a popular open source library in C++ for developing embedded system design, from the abstract System Level Design until the accurate Register Transfer Level Design. SystemC simulation, however, runs in console mode (text-based), thus making it difficult for user to interact with the simulation. To extend the capabilities of SystemC simulation, it is necessary to create Graphical User Interface. In this paper we recommend to use RapidPLUS tool for making the interface for embedded system prototype to reduce time to market. We also propose the connectivity between RapidPLUS and SystemC by using socket communication that attached in JavaBean RapidPLUS object

Keywords: SystemC, Embedded System, RapidPLUS, Simulation, Socket, virtual prototyping.

1 Introduction

In process to making the good embedded system products, usually we are not only faced with quality but also must deal with time. The average time to market constraint has been reported as having shrunk to only 8 month [1]. There are many ways to reduce time to market; one is by using prototyping such as real and virtual prototyping.

In this paper we want to suggest how to manage these problems by providing a suitable environment. The environment that we propose will use the SystemC as a simulation engine along with RapidPlus as a tool for making the interface.

SystemC can be used to make a virtual prototyping, where virtual prototyping is one way to speed up the development process [3]. However, SystemC fails to provide a graphical user interface (GUI). It only supports text-based console application. As a feedback with a user during simulation, the user only can use printf or cout[5,6].

* This research was supported by Ministry of Information and Communication, Korea, under the ITRC (IT Research Center) support program supervised by Institute of Information Technology Assessment.

R. Obermaisser et al. (Eds.): SEUS 2007, LNCS 4761, pp. 389–393, 2007.

To handle this weakness, we use RapidPLUS to build the system interface. RapidPlus is one of good tool for making interface prototypes. It is a comprehensive software package for the generation of simulation and prototypes of embedded systems [7]. Furthermore, to connect both of them, we propose to use the socket mechanism.

We also want to show an example how to make an interface in RapidPlus. In the previous research, we have implemented the SystemC in company with Java as a GUI. This paper does not show how to implement the connectivity between SystemC and RapidPLUS instead of just show how the RapidPLUS can be communicate with SystemC. The implementation still not yet finished, however we are still doing to implement.

The paper is organized as follows: Section 2 describes the related work, and Section 3 presents the system environment. In Section 4, we describe out the RapidPlus testing, and in Section 5 we conclude our work.

2 Related Works

Many researchers have been worked to improve the ability of SystemC. We can look some of example in [5, 6]. The architecture, as given in the Figure 1, is comprised of three main parts: SystemC, Java and CommunicationLib.

In SystemC terminology, the processes are methods. It recognizes two kinds of processes, SC_METHOD and SC_THREAD. For utilization of these processes, specific port and signal must be set. Because of this procedure, processes will be triggered during data transmission by changes of signal and port. Declaration of the process and the sensitive are registered in constructor (SC_CTOR) which is included in the module. Child module initialization and interconnection will be declared in constructor. Of these circumstances, SC_MODULE, SC_THREAD, SC_METHOD and SC_CTOR are macros in SystemC.

SystemC also has own mechanism to connect modules which consists important role due to number of dependant modules. Single module port is responsible to input data (SC_IN) and pairs for output data (SC_OUT), but others can be responsible to input and output data. Single SC_OUT can receive more than one SC_IN.

2.1 Client Side

For the Client, Java Swing was used to develop the Television Simulation for GUI application. This is due to Object Oriented enhancement and provision of various libraries. The Television Simulation can be divided into input and output modules. Input modules are used to input data from users via push buttons or text field. Received inputs conveys to SystemC through CommuLibrary API. Output modules receive data from SystemC through the SCJLib API which enables interface manipulation to help users to understand how the server operates.

2.2 SCJLib API

SCJLibrary is an API for bridging SystemC (server) and GUI simulation (Java Swing). Integration between server (back end) and client (front end) are very important for the users, due to user friendly interaction and efficiencies.

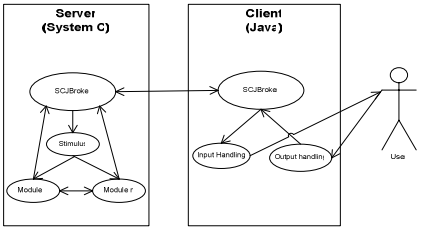

Fig. 1. The Server (SystemC) and The Client (Java) System Architecture

There are three main classes in this library. SCJBroker class is responsible to provide connection between server and client (see figure 1), and each part must have only one instance of this class. SCJOutput class is used to send data from client to server. Unlike the SCJBroker class, it has only one instance in each part, whereas SCJOutput class must instantiate for each number of data variations for delivery. In simulation using SCJLib API, at least one instance of SCJOutput should be created.

3 System Environment

The new environments for the recommendation system will changes client side. Besides using the Java as an Interface, the new system will apply to use the RapidTools. The detail of RapidTools will be emphasized in this section. However, before we give details about the RapidTools, Figure 2 presents our new suggestion for the new system architecture.

There are six stages that be used to track the RapidPLUS progress. First, we will use the Object Layout to create the application's user interface. This is very important stages since the market sales record condition depend on the interface.

The object, JavaBean, plays the basic rule for communicate to the SCJLib. The JavaBean acquire data from the other objects properties such as position of frame object or the value of the buttons and than send data to the SystemC. On the other hand, the respond data from SystemC will traverse to the JavaBean object, and they will give back to the other RapidPLUS Objects.

The second until the fourth stages are to define the modes and their transition, including the triggers. The modes can be illustrated as a state machine. They will transfer from one mode to others mode by waiting the triggers. The triggers can be event or condition triggers.

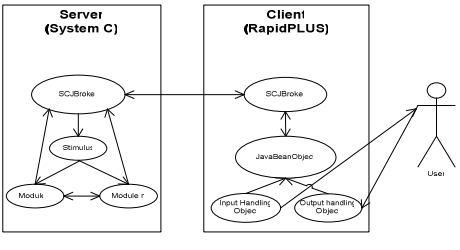

Fig. 2. The New System Architecture, SystemC as a Server, RapidPlus as a Client

Defining the activities is the fifth stages. In this stage the data from among the objects transfer each one another. The activity has three modes. Entry mode, mode, and exit mode. All activities in entry mode will be confirmed any time the source is accessed. The mode's mode will run continuously while the source is read. In contrast, the exit mode just can be run while the source exits its mode.

4 RapidPLUS Testing

Since RapidPLUS can be applied for creating the good embedded system interface, the following section will give an example how to use and test the RapidPLUS tool. The example will utilize the six stages that have been mention in the last section.

We made an elevator simulation where it contains some function such open and close door, running up, running down and automatic waiting time. Figure 3 depict the elevator interfaces that build in RapidPLUS.

4.1 Adding Object

There are some objects that must be integrated in order making the runtime simulation. As a normal elevator, the button objects must have the floors button, open and close door button. Furthermore, the bitmap picture objects have been chosen to illustrate the exact car door and car wall.

Figure 3 represent all the graphical objects for our runtime testing elevator. In addition, we also should insert some non graphic objects such as timers.

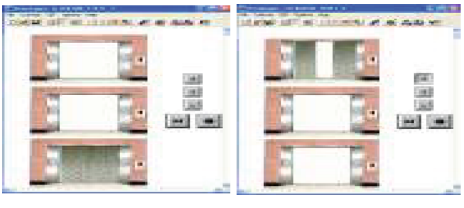

Fig. 3. The elevator in Runtime Test: (a) show the idle time, (b) show the automatic open and close door

4.2 Insert Triggers, Actions and Activities

Most developers who run RapidPLUS the first time will said that better to work in Java coding or another GUI language than just use the RapidPLUS tool. We, however, can prove that assume is not true, since we understand the behavior of trigger, action and activities.

Simply in our simulation, the triggers mostly associate with press button in, press button out and timers. For example, to open the door from idle time, we should press the door open button and than the button in condition will go to the activity mode. The activity mode will trigger the destination mode, in this case door open. Door open will keep door open until another trigger, door open button, is push out. The stage will continuo to another destination, close door.

The actions are similar to activities, but they take place only during transitions. We utilize some actions along with internal transitions

5 Conclusion

RapidPLUS is one tool for designing interface easily than using the traditional write code. It contains many objects that can be used for making interface design. In our experience, RapidPLUS can be used as the client side for making virtual prototyping along with SystemC as server side.

As a conclusion, this paper studied about the ability of RapidPLUS as a tool for making virtual prototyping in order to reduce time to market. In addition we propose to use the SystemC as an engine instead of direct to use hardware description language, because the time to market is our constraint. This paper are still implementing, however, the prototyping interface have been build with elevator example. Furthermore, the next research may deal to create the communication library and systemC engine.

References

1. Vahid, F., Givargis, T.: Embedded System Design: A Unified Hardware/Software Introduction, 1st edn. John Wiley & Sons, Chichester (2002)
2. SystemC 2.0 User Guide, SystemC.org, http://www.systemc.org/
3. Simulation Based Design Center of Univ. of New Orleans. Primer on Virtual Prototyping, http://www.gcrmtc.org/sbdc/protoprimer print.html
4. Sukmana, T., Satria, H., Kwon, J.B., Lee, J.B., Kee, W.: User-level Virtual Prototyping for Television Simulation using SystemC and Java GUIHusni
5. "RapidStart 8.0", pp. 1, e-SIM. Ltd. (2004)