SID 자바 컴포넌트 빌더: SID 시뮬레이터를 위한 자바 컴포넌트 통합 개발환경

익산, 페비, 권진백
선문대학교 컴퓨터공학과
e-mail : putra.ikhsan@gmail.com, havban@gmail.com, jbkwon@sunmoon.ac.kr

SID Java Component Builder: An Integrated Development Environment for Java Component for SID Simulator

Ikhsan Putra Kurniawan, Febiansyah Hidayat, Jin Baek Kwon
Dept. of Computer Science Engineering, Sun Moon University

Abstract

Embedded system developers use design and testing tools to make their product faster. Previously [1, 3] developed a virtual development environment for embedded software (VDEES) using open source software, mainly the SID simulation framework for a simulator engine and the Eclipse platform for a development platform. VDEES enables developers to develop SID Component in C++. A bridge module for developing SID Component in Java has been developed and available [2]. However, using this module, developers have to build their SID Java component from scratch. In this work, we developed SID Java Component Builder Plug-in as an additional features to VDEES. This tools enables developers to build SID Component in Java faster and easier.

1. Introduction

A Virtual Development Environment for Embedded Software VDEES [1, 3] enables developers to develop SID Component in C++. A bridge module for developing SID Component in Java has been developed and available [2]. However, in this this module, developers have to build their SID Java component from the scratch. In this work, we developed SID Java Component Builder Plug-in as additional features to VDEES. This tool enables developers to build SID Component in Java faster and easier.

The rest of the paper is organized as follows: We introduce the background knowledge of our work in section II. And we present the structure of plug-in, a structure of running project/SID Java Component, and GUI design in section III. Section IV demonstrates the functionality and the usage of Java Component Builder with an example. Finaly we conclude in section V.

2. Background

This section introduces the background knowledge of our work. This chapter consists of knowledge about VDEES, JavaBridge Module, SID, and Eclipse.

VDEES is a virtual development environment for embedded software, which provides a fully featured integrated development environment (IDE) for embedded software without a real target platform. VDEES is implemented based on the Eclipse platform, an open source project for building IDEs. The virtual environment provides a configuration tool for composing a virtual target, a code editor for writing simulated components, software to be run on the target, building tools for binary images, a debugger for investigation of the software running on the virtual target, and a system monitor for the investigation of the virtual target[1, 3].

JavaBridge Module is a software that enables SID simulation environment to use Java Component transparently[2].

SID is a framework for building virtual platforms as GPL-licensed open source software. Specifically, a simulation is comprised of a collection of loosely coupled components and SID defines a small component interface that serves to encapsulate them tightly [1, 3, 7].

The Eclipse platform [1, 3, 4] is designed for building IDEs as an open source project. The most important feature of Eclipse is extendibility. Eclipse can be extended to support any kind of programming language, tools, or application. The Eclipse platform [4] integrates the individual tools into a single product, providing a rich and consistent experience for its users. The Eclipse platform supports Java IDE by adding Java development components (e.g., the JDT [5]) and it is turned into a C/C++ IDE by adding C/C++ development components (e.g., the CDT [6]). JDT and CDT are examples of Eclipse plug-ins that can be added into Eclipse.

SID Java Component Builder is additional plug-in for VDEES. It is similar to one of VDEES features: SID Component Builder which enables developers to build SID Component in C++. We developed SID Java Component Builder to enable developers to build SID Component in Java.

3. SID Java Component Builder

This section presents the structure of plug-in, a structure of running project/SID Java Component and GUI design for easy of use.

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency)" (NIPA-2011-C1090-1131-0004)

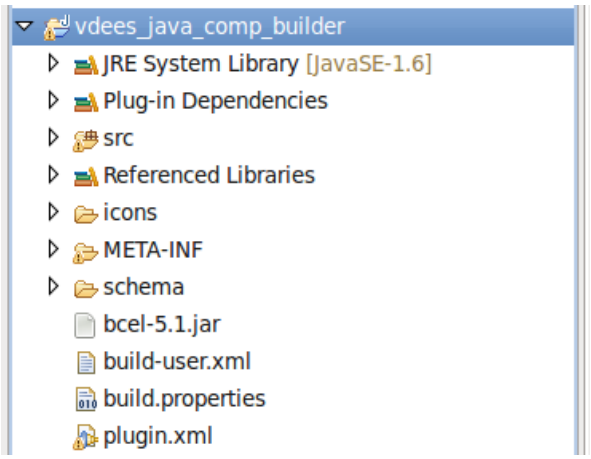
Before we describe about the structure of plug-in, we will explain about parts of this plug-in first.

This plug-in consists of four parts: Create Project/SID Java Component, Add Extra Input Pin or Internal Bus, Build Project/SID Java Component, and Build VDEES / .jar.

Here is a brief explanation about these parts:

1. Create Project/SID Java Component
   This part is used when a developer wants to create a new component. We provide two type of templates for developer: i. Extends Simple-Component, ii. Extends SIDJComponent. Template that extends SimpleComponent is a simple version of SID Java Component. It is similar to "Hello World" in Java Programming. Template that extends SIDJComponent is a regular version of SID Java Component. The project that we created is associated to **javanature,** this will makes this project to be recognized as java project in eclipse. Informations about Input Pin, Output Pin, Internal Bus, External Bus will be asked to developer here. When developer is not sure about these informations, he/she could add extra Input Pin or Internal Bus later.
2. Add Extra Input Pin or Internal Bus
   This part is used, when a developer needs to add extra Input Pin or Internal Bus.
3. Build Project / SID Java Component
   Because this plug-in uses javanature as its nature, build project uses default buildproject that is available in eclipse.
4. Build .jar
   After SID Java Component is built at number 3, developer could build its component into .jar file. This .jar is ready to be run using SID runtime and JavaBridge component

The structure of this plug-in can be seen on image below.



(Figure 1) VDEES Java Component Builder Structure

This plug-in contains of 7 folders: JRE System Library, Plug-in Dependencies, src, Referenced Libraries, icons, META-INF, schema. Most of these folders are default folders needed to create eclipse plug-in.

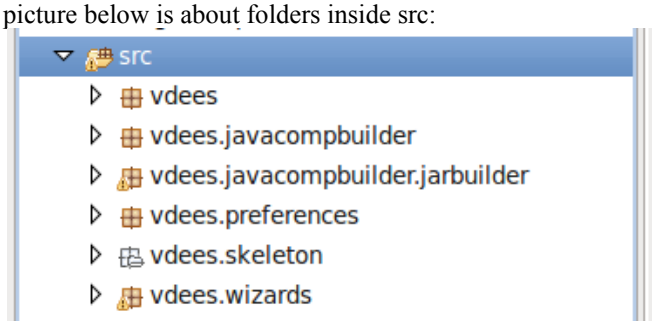
Folders contain of:

1. JRE System Library, Plug-in Dependencies, META-INF are default folder that needed to create plug-in in eclipse.
2. Icons is a folder to save image for icon.
3. Schema is contain of jarutil.exsd, needed by "Fat Jar Eclipse Plug-In" [8] (we reuse part of this plug-in for Build VDEES feature).

And files contain of:

1. build-user.xml, plugin.xml, build.properties are default files needed to create eclipse plug-in.
2. bcel-5.1.jar is needed by "Fat Jar Eclipse Plug-In".

Most of our work are located inside src folder. This picture below is about folders inside src:



(Figure 2) Structure of src folder

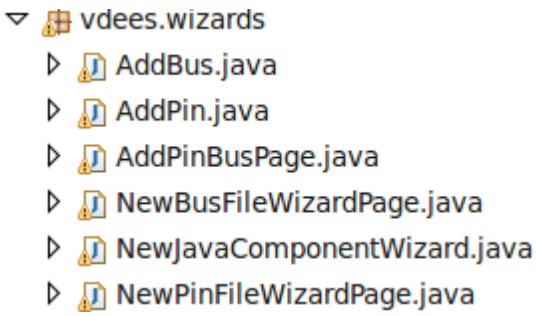
There are six folders inside src folder:

1. vdees
   There are two files inside this folder: Java-Resources.java and String.properties. It is needed as a bundle that contains informations like project name, project description, and etc. Those informations can be retrieved using JavaResources-.java.
2. vdees.javacompbuilder
   There are three files inside this folder:
   - JavaCompBuilderPlugin.java, this is the main class to be used in the desktop.
   - JavaPluginConstants.java, this contains variables like PLUGIN_ID, BUILDER_ID, NATURE_ID.
   - JavaPluginTools.java(contains makeStatus method, used by reportError method, in NewJavaComponentWizard.java which shows errors occur during project creation).
3. vdees.javacompbuilder.jarbuilder
   This folder is responsible for handling BuildVDEES feature. There are two files inside this folder:
   - JavaManifestData.java, used by JavaVdees-Builder.jar
   - JavaVdeesBuildJar.java, will create ".jar" file for running project.
4. vdees.preferences
   This folder responsible for providing preference window for user to choose the location of

JavaBridge.jar and target location of ".jar". There are three files in this folder: JavaCompDev-PreferencePage.java, JavaPreferenceConstants.java, JavaPreferenceInitializer.java. Those three classes will define preferences for developer including: SID_JAVABRIDGE_PATH and SID_JAVA_COMPLIB_PATH. SID_JAVABRIDGE_PATH is the location where "bridge" is located. And SID_JAVA_COMPLIB_PATH is the location where to put .jar for running component/project.

5. vdees.skeleton
This folder contains class templates for VDEES Java Component. There are four files inside this folder. Template is the main class template that extends SimpleComponent, and template_sid-jcomponent is the main class template that extends SIDJComponent. Another two files are inputpin and internalbus. Both used as template for inputpin and internalbus.

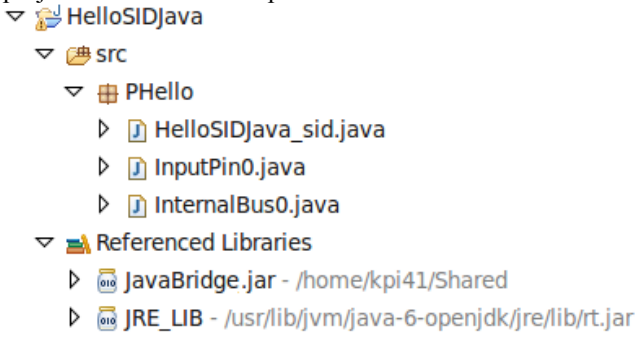
6. vdees.wizards
This folder is resposible to provide window during creating project, and adding extra inputpin or internalbus. It contains of five files. Picture below shows folders inside vdees.wizards:



(Figure 3) Inside folder vdees.wizards.

NewJavaComponentWizard.java and AddPinBusPage will provide windows during project creation phase. It will gather information about project name, package name, number of total pin and bus. AddPin.java and NewPinFile-WizardPage.java will provide windows for addition of inputpin. AddBus.java and NewBusFileWizard.java will provide windows for addition of internalbus.

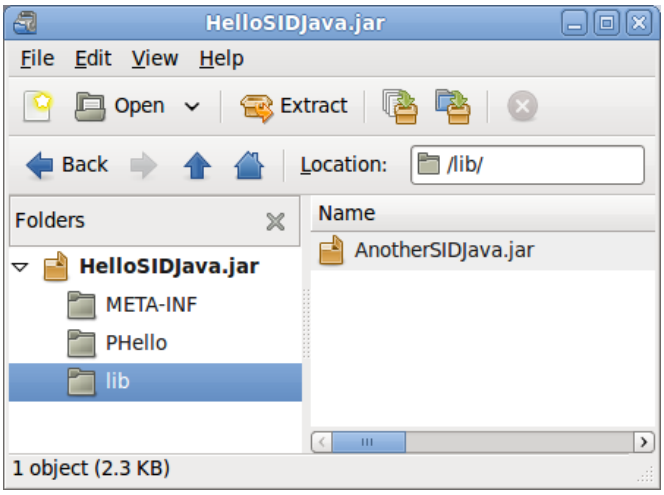
Next, we will describe about the structure of running project or SID Java Component.



(Figure 4) SID Java Component Structure

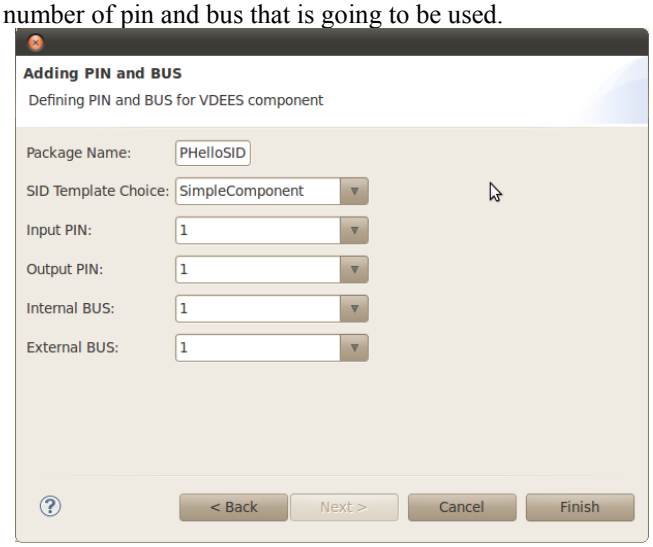
As we can see from Figure 4., running project/SID Java Component consists of two folders: src and Referenced Libraries. In src, main class of SID Java Component, its input pins and internal buses are located here. In Referenced Libraries there is JavaBridge.jar and another libraries that needed by developer are located.

Structure of Project/SID Java Component after .jar is built:



(Figure 5) SID Java Component after .jar is built

We made friendly GUI for easy of use for developer to develop SID Java Component. GUI for user is created to handle new project creation, adding extra input pin or internal bus. For example, when a user tries to create a new project, user will be asked to choose from combo box total number of pin and bus that is going to be used.



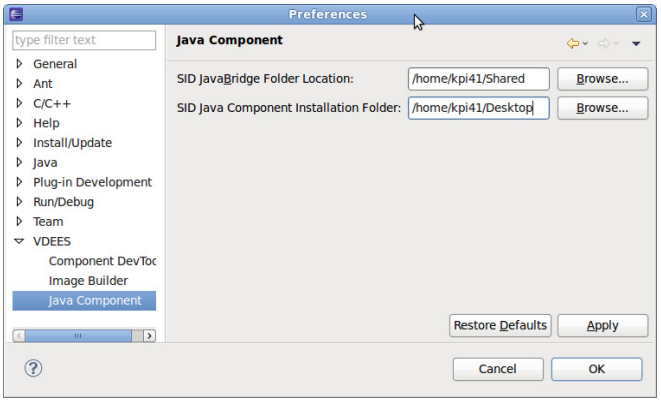
(Figure 6) Combo box for total number of pin and bus

4. Example

This section describes how to start developing SID Java Component using this plug-in, how to add extra input pin/internal bus, how to add/remove library, how to update some parameters, and how to build project/component.

Before starting to create SID Java Component, called project in eclipse, user needs to define the location of JavaBridge.jar and target location of ".jar". A ".jar" target

location is a target folder to put ".jar" result of this project. These parameters can be set by clicking Windows > Preferences on main menu. Then choose VDEES > Java Component. Find the JavaBridge.jar location and ".jar" target location.



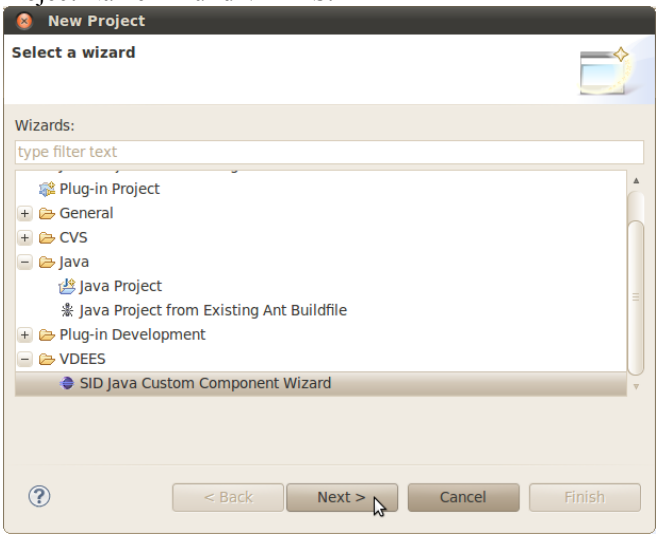
(Figure 7) Defining library location

To start developing a project, click File > New > Project on menu. Choose VDEES > SID Java Custom Component Wizard. Fill in Project name, click Next. On the next page on that window, fill in Package name, choose from combo box: SID Template choices, number of total input pin, output pin, internal bus, external bus, and then click finish.

An SID Java Component template is ready to be modified to build a component.

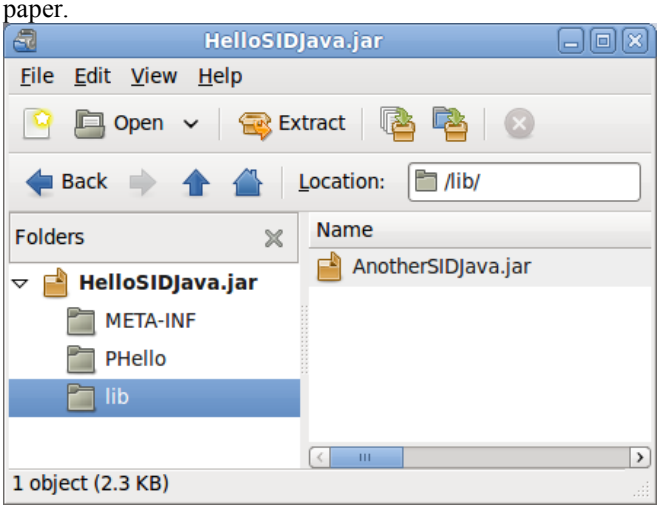
During the development of project, if an extra pin or bus is needed, user could simply add it by right clicking on Package Folder at Package Explorer > New > Other. New window will appear, choose VDEES > Add Bus or Add Pin.

Once the project is developed, it is time to build the project and make ".jar" of that project as a component. To build project is very simple. Right click on Project Name > Build Project. To build ".jar" is easy too. Right click on Project Name > Build VDEES.



(Figure 8) A Window to Start Developing Project

To add and remove library, it can be done by opening properties window of the project. Right click on the project name > properties > Libraries. For example, "Add External JARs..." or "Remove" button. There is another documentation to find detail about this part, not part of this paper.



(Figure 9) Example of SID Java Component after BuildVDEES

5. Conclusion

In this work, we developed SID Java Component Builder as an additional plug-in for VDEES. It enables developers and researchers in developing SID Java Component faster and easier.

References

[1] Hadipurnawan Satria, Budiono Wibowo, Jin B. Kwon, Jeong B. Lee, Young S. Hwang, "VDEES: A Virtual Development Environment for Embedded Software Using Open Source Software", IEEE Trans. Consumer Electron., vol. 55, pp. 959-966, May 2009.

[2] Hasrul Ma'ruf, Febiansyah Hidayat, Jin Baek Kwon, "Java bridge module and Java API for SID simulator", Proceedings of the 10th WSEAS international conference on Software engineering, parallel and distributed systems, February 2011.

[3] VDEES. http://april.sunmoon.ac.kr/vdees/.

[4] Eclipse Platform. Http://www.eclipse.org/.

[5] Eclipse Java Development Tools (JDT). http://www.eclipse.org/jdt/.

[6] Eclipse C/C++ Development Tools (CDT). http://www.eclipse.org/cdt/.

[7] SID. http://sourceware.org/sid/.

[8] Fat Jar Eclipse Plug-In. http://fjep.sourceforge.net/.