UML Interaction Model Driven Runtime Verifier User Guide

Nanjing University

April. 2010.

Contents

Chapter 1 Introduction

Runtime verification is a lightweight approach to program reliability. Its basic idea is to gather information during program execution and use it to conclude properties about the program, either during testing or in operation, which increases the confidence in whether the program implementation conforms to its specifications.

UIMDRIVER standing for UML Interaction Model Driven Runtime Verifier is a runtime verification tool for JAVA programs. It introduces UML2.0 interaction overview diagrams and sequence diagrams to construct simple but expressive scenario-based specifications, and takes a runtime verification approach to check the consistency between JAVA programs and their specifications. It is able to perform two types of consistency checking: exceptional consistency checking and mandatory consistency checking. The exceptional consistency requires that any forbidden scenario described by a given interaction overview diagram never happens during the execution of a program, and the mandatory consistency requires that if a reference scenario described by a given sequence diagram occurs during the execution of a program, it must immediately adhere to a scenario described by a given interaction overview diagram.

The tool can check four types of specifications:

Exceptional consistency specifications require that any forbidden scenario described by a given interaction overview diagram D never happens during the execution of a program;

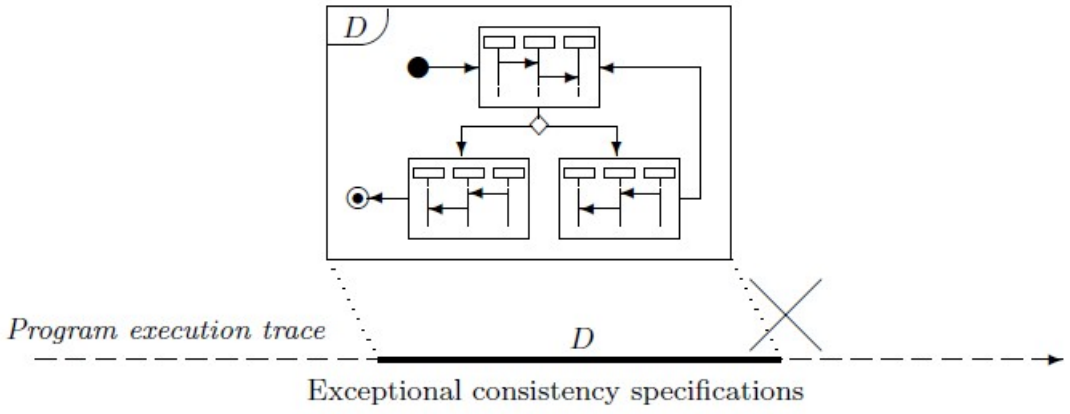


Figure 1. Exceptional consistency specifications

Forward mandatory consistency specifications require that if a reference scenario described by a given sequence diagram D occurs during the execution of a program, then a scenario described by a given interaction overview diagram G must follow immediately;

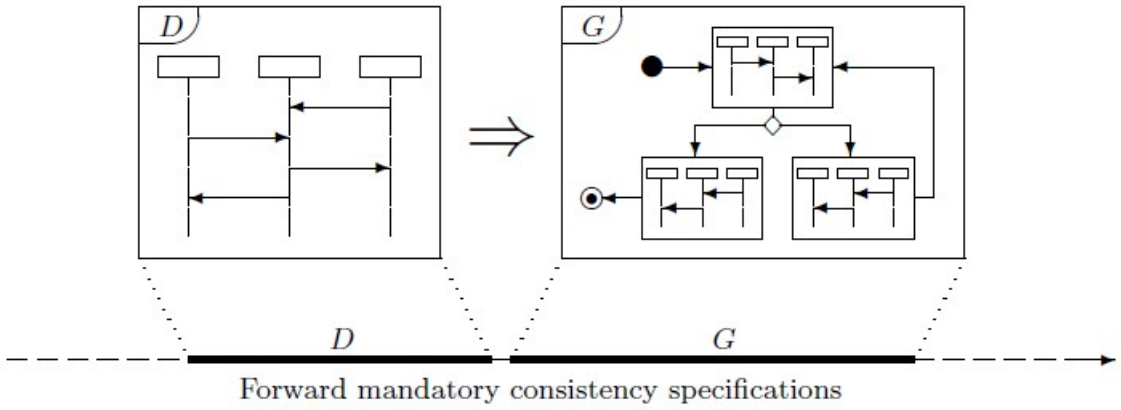


Figure 2. Forward mandatory consistency specifications

Backward mandatory consistency specifications require that if a reference scenario described by a given sequence diagram D occurs during the execution of a program, then it must follow immediately from a scenario described by a given interaction overview diagram G;

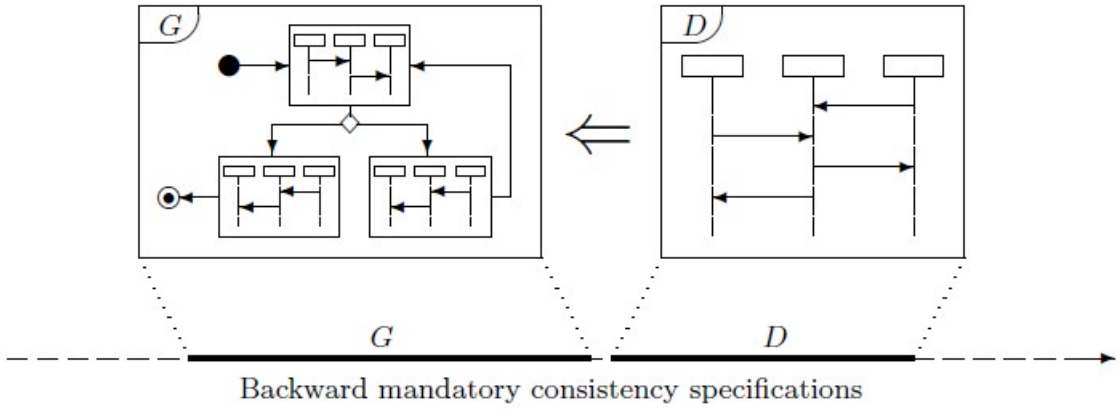


Figure 3. Backward mandatory consistency specifications

Bidirectional mandatory consistency specifications require that if a reference scenario described by a given sequence diagram D1 occurs during the execution of a program and another reference scenario described by another given sequence diagram D2 follows, then in between these two scenarios, a scenario described by a given interaction overview diagram G must occur exactly.

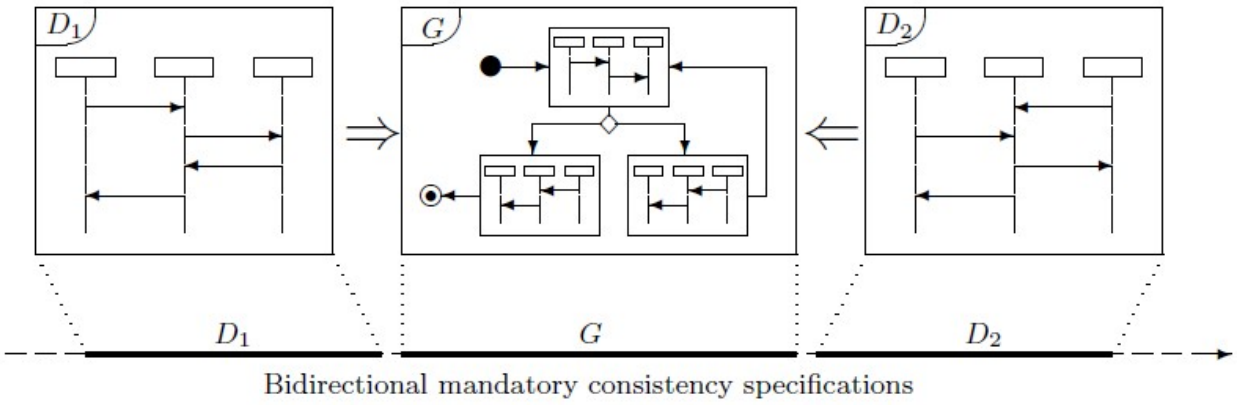


Figure 4. Bidirectional mandatory consistency specifications

UIMDRIVER is developed by the Software Engineering Group, Department of Computer Science and Technology, Nanjing University, P.R. China. It is available at http://seg.nju.edu.cn/UIMDRIVER and is free for any academic use. Any suggestion, you may contact lxd@nju.edu.cn.

Chapter 2 Installation

2.1 Requirements

2.1.1 Eclipse IDE

The UIMDRIVER is built as an Eclipse Plug-in, it requires the Eclipse IDE 3.4 or higher running on Windows XP.

2.1.2 Java Runtime Environment

The UIMDRIVER requires the JRE 1.6 or higher.

2.1.3 Third PART PLUG-INS

UIMDRIVER provides an Interaction-overview diagram editor based on the TOPCASED plug-in and the latter needs a set of plug-ins. The list of dependent plug-ins is give here:

Zip files:

- emf-sdo-xsd-SDK-2.4.2
- emf-transaction-SDK-1.2.3
- emf-validation-SDK-1.2.1
- GEF-SDK-3.4.2
- gmf-sdk-2.1.3
- mdt-ocl-SDK-1.2.2
- mdt-uml2-SDK-2.2.2
- org.topcased.sdk-2.5.0
- epfc-richtext-1.5.0.2

JAR files:

- net.sourceforge.lpg.lpgjavaruntime_1.1.0.v200803061910.jar
- dom4j-1.6.1.jar

2.2 Installation

The installation process consists of three steps. First, you need to download all the dependent plug-ins as mentioned above. Then, you are asked to install all the plug-ins into the Eclipse environment in your computer. Third, you can install the plug-ins download from our website.

2.2.1 Download all the Third PART PLUG-INS

Download all the third part plug-ins from the URL given above.

2.2.2 Install the PLUG-INS

1 Install those first nine zip format plug-ins: Unzip those three zip files, you may find that each of them consists of two folders: ‘plugins’ and ‘features’. Copy both of them to the root directory of your Eclipse environment. These two folders will be merged with folders into your Eclipse environment.

2 Install the jar format plug-ins:
net.sourceforge.lpg.lpgjavaruntime_1.1.0.v200803061910,
Copy the jar package into the ‘plugins’ folder in your Eclipse environment.

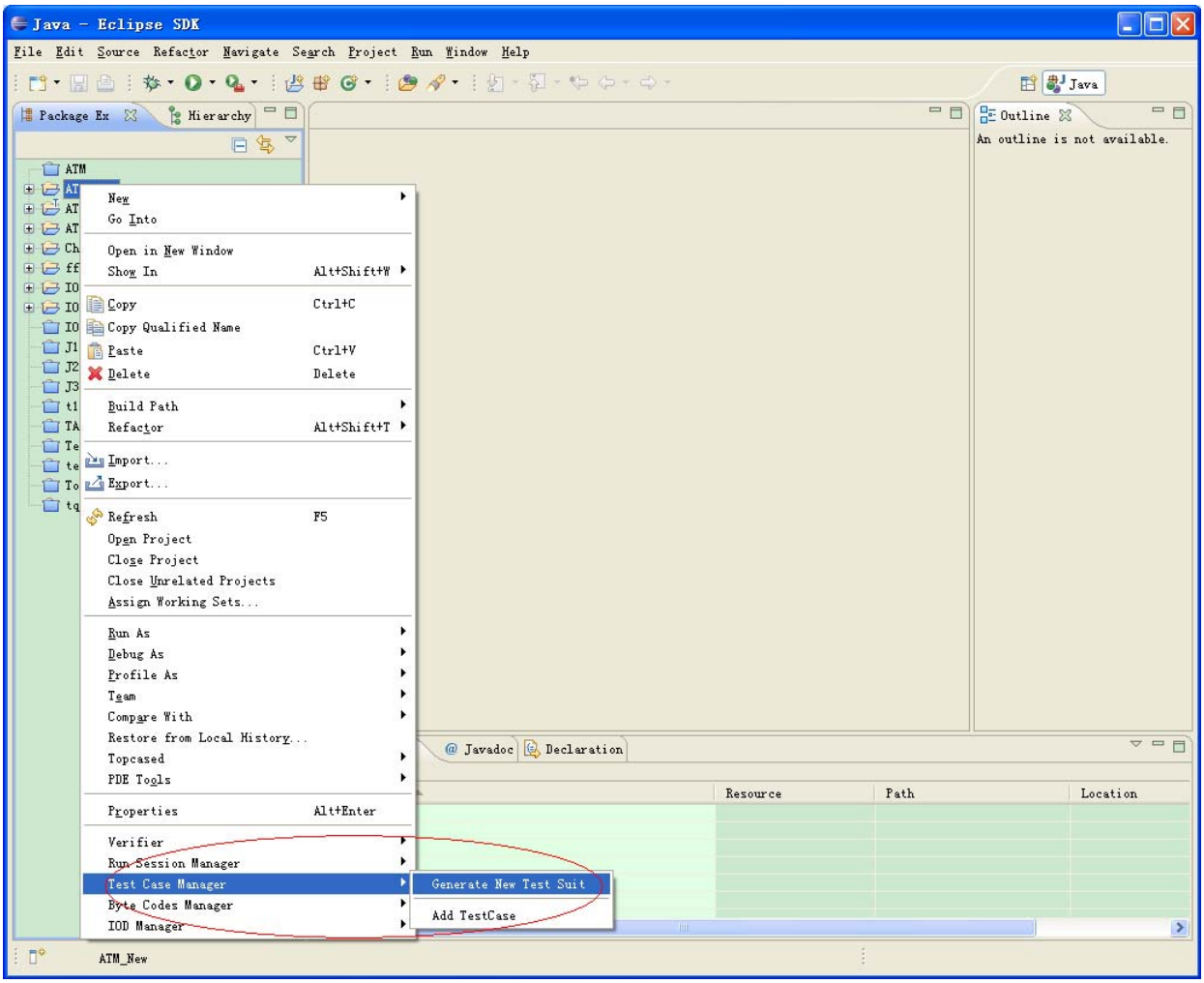
2.2.3 Install the UIMDRIVER

1 Install the plug-in dom4j-1.6.1.jar into the UMIDIRVER plug-in:
Open the cn.edu.nju.seg.uimdriver_1.0.0 with winrar.exe, and use the ‘add’ menu to add the dom4j-1.6.1.jar into the following to location:
.\Lib
.\Lib\IODVerifier.jar\.lib

2 Copy the following three jar format plugins into the ‘plugins’ folder in your Eclipse environment.
cn.edu.nju.seg.jasmine.sequencediagram_1.0.0.jar,
cn.edu.nju.seg.jasmine.interactiveoverviewdiagram_1.0.0.jar,
cn.edu.nju.seg.uimdriver_1.0.0
into the ‘plugins’ folder in your Eclipse environment.

2.3 Inspect the installation

If all the plugins are correctly installed, the Eclipse will automatically load and initialize them when starting up. You may find new items added in the right-click menu, as shown below:



Chapter 3 Use UIMDRIVER

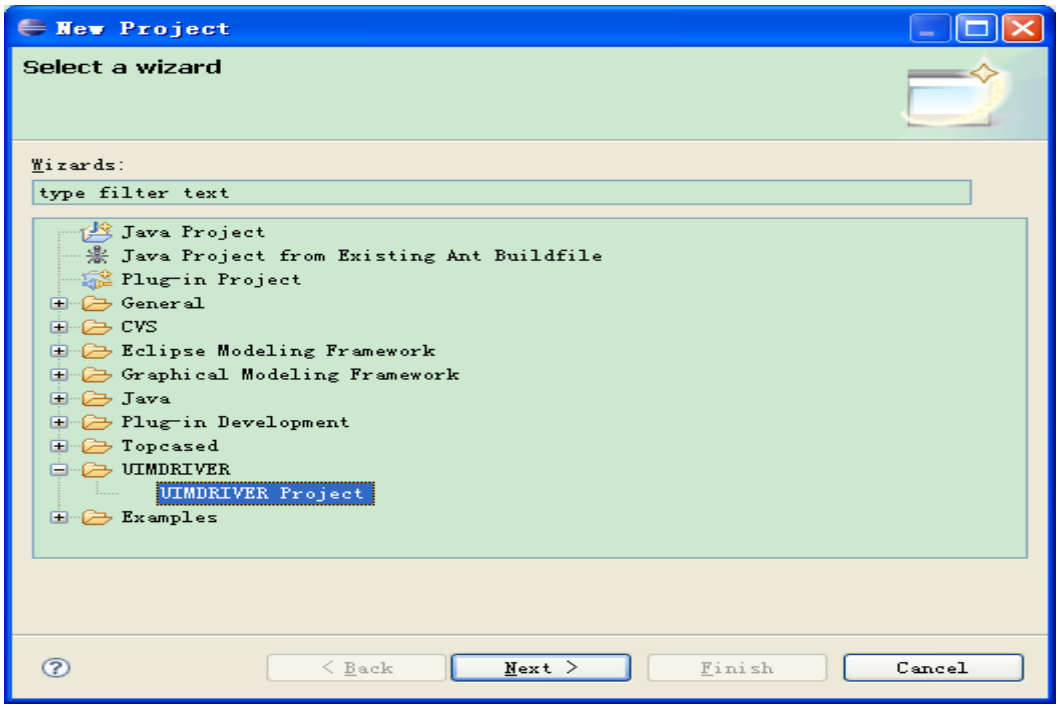
With the help of the UIMDRIVER, one may perform the run-time verification process. The whole verification working flow consists of seven steps:

- Create a verification project
- Import byte codes of the JAVA application under verification
- Create/import an IOD
- Create a Test Suite
- Create Test Cases
- Execute Test Cases
- Generate Verification results

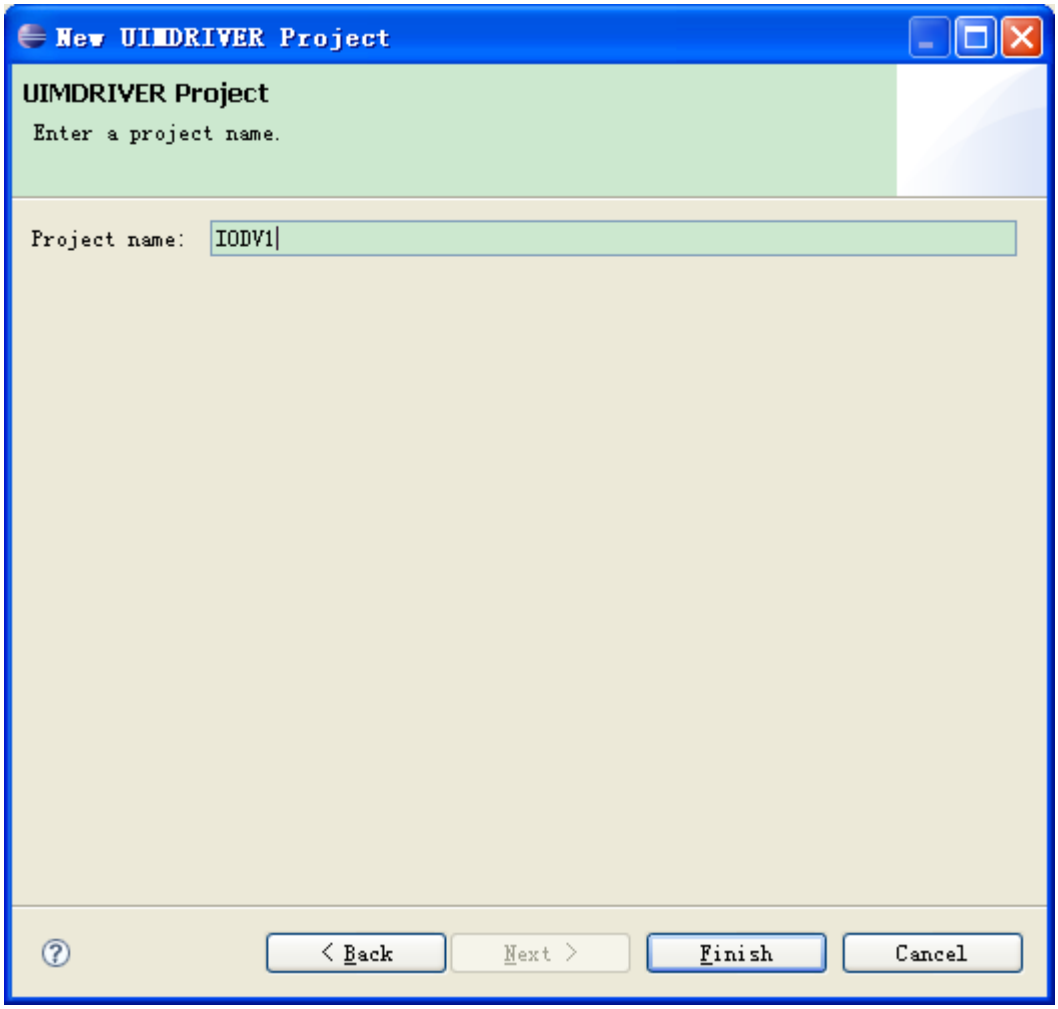
3.1 Create a verification project

A Verification Project is used to manage all the resource related to a verification process. Examples of these resources are byte codes, Interaction overview diagrams, instrumented codes, execution traces and verification results and etc.

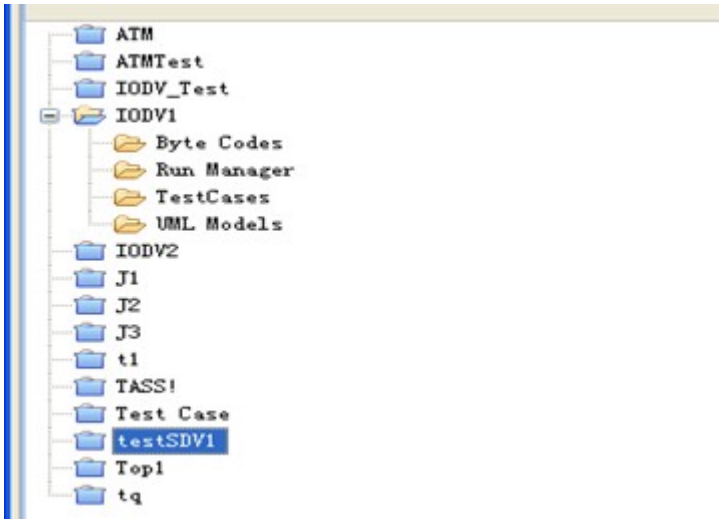
Open the menu 'File'->'New'->Project', choose the 'UIMDRIVER Project' item in the 'UIMDRIVER' folder.



Click the Next button and enter the name of the project.

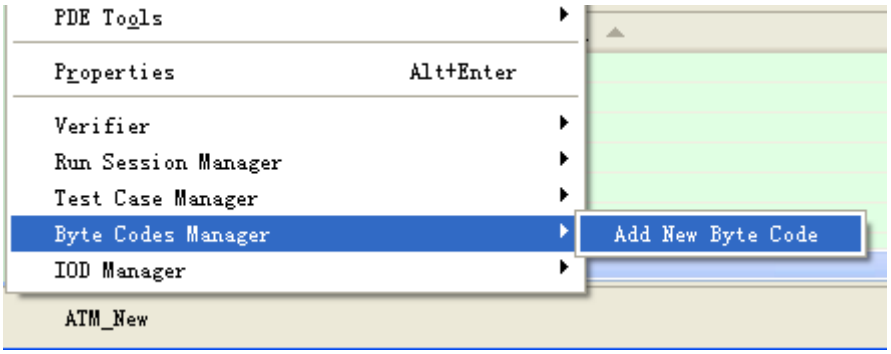


Click the finish button and a new project is created.

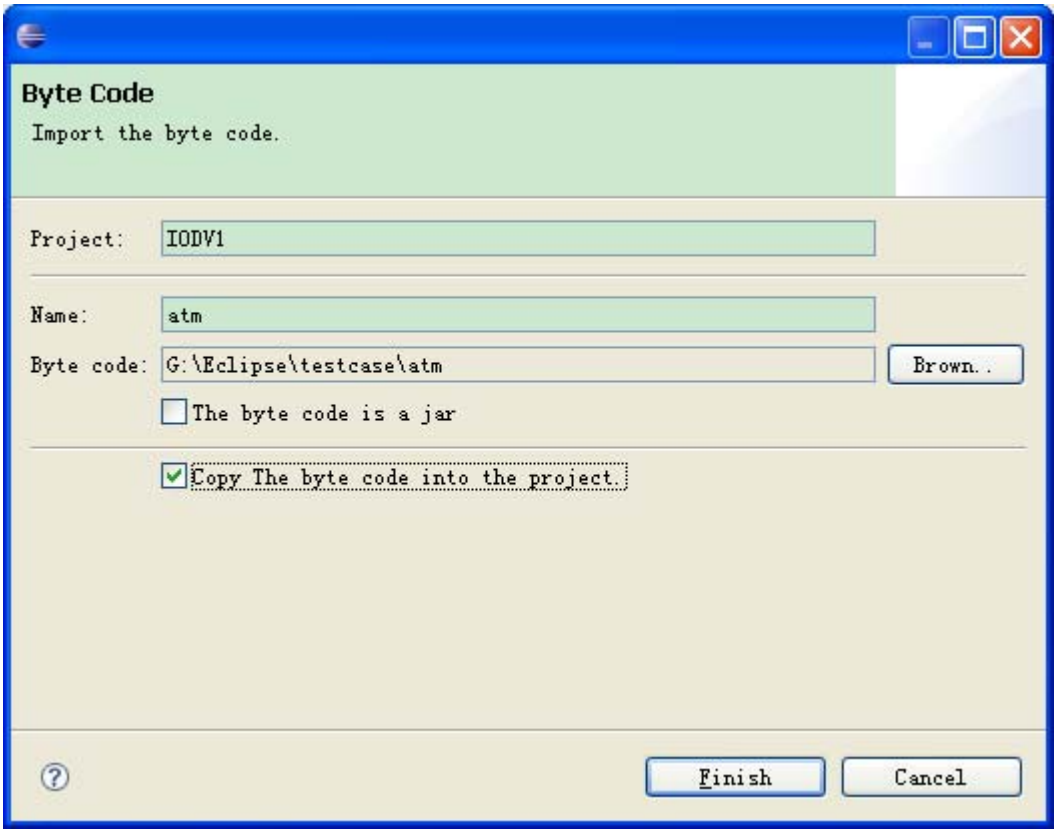


3.2 Import Byte Codes

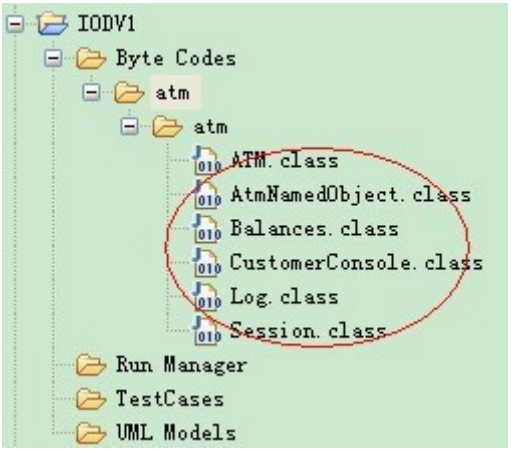
Choose the ‘Add new byte code’ item in the right click menu ‘Byte Codes Manager’.



In the popup window, specify the location of byte codes.

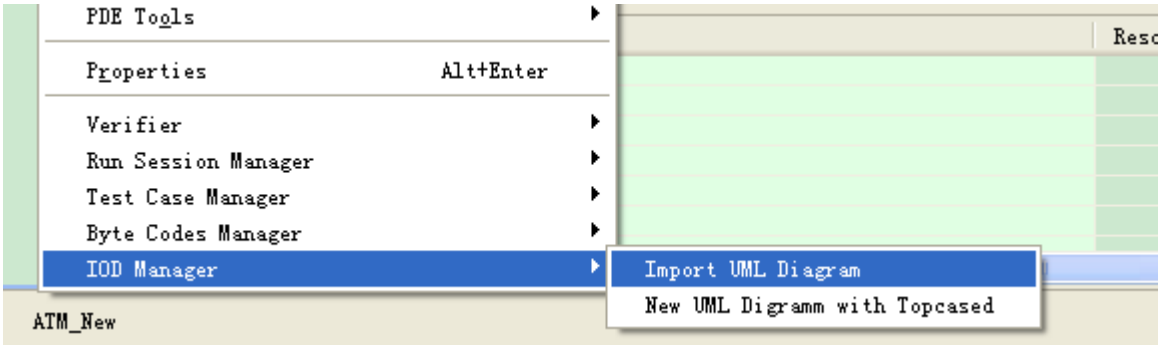


Click the finish, then the byte codes in imported into the project.

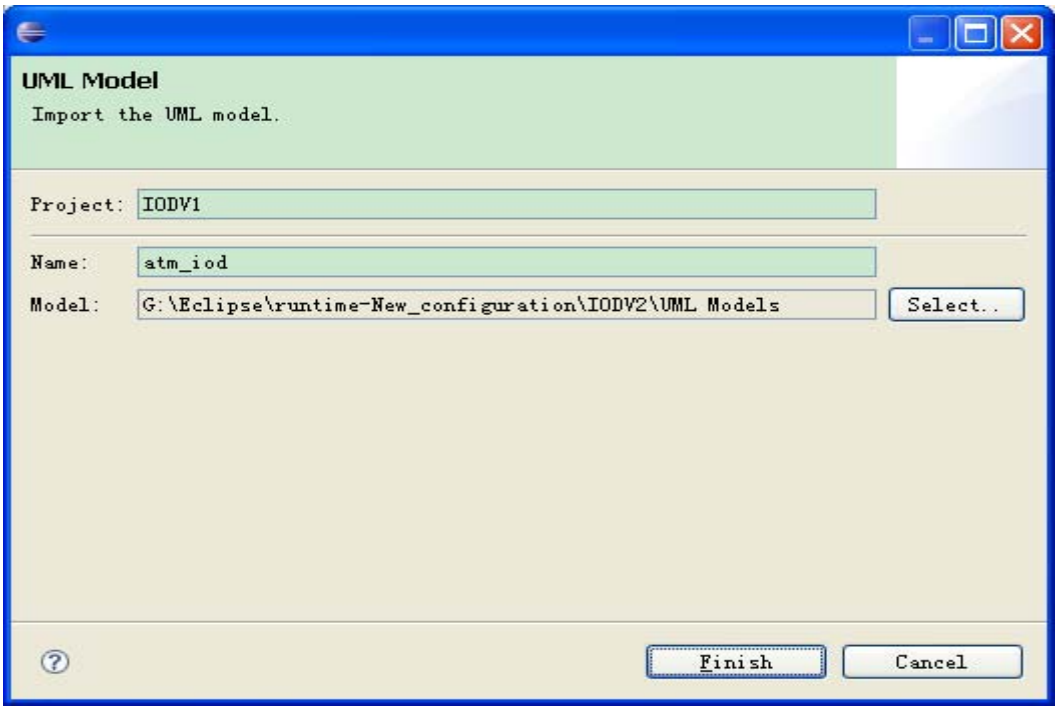


3.3 Import Interaction Overview Diagrams

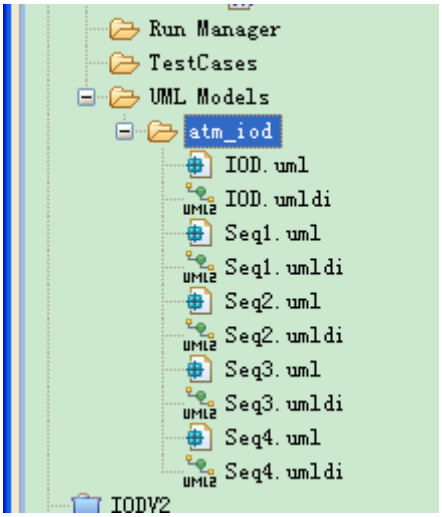
Click the 'Import UML Diagram' item in the right click menu 'IOD Manager'.



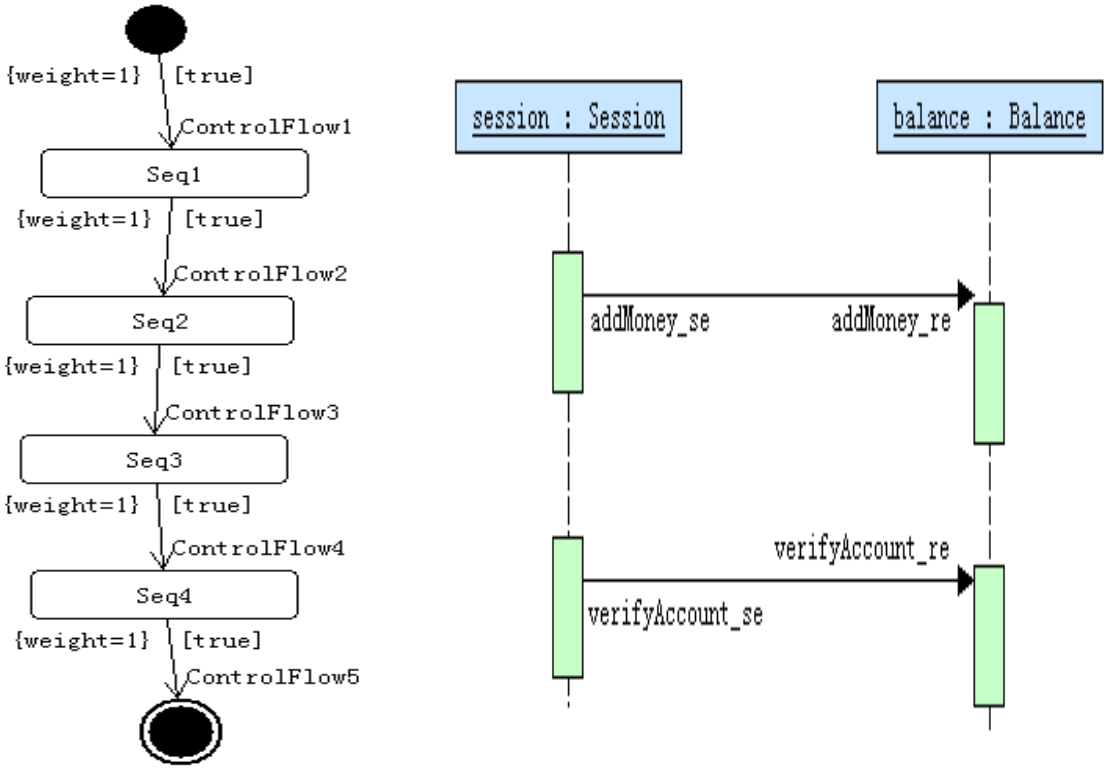
In the popup window, specify the location of diagrams or create a new diagram.



Click finish, then the interaction overview diagrams are imported into the project.



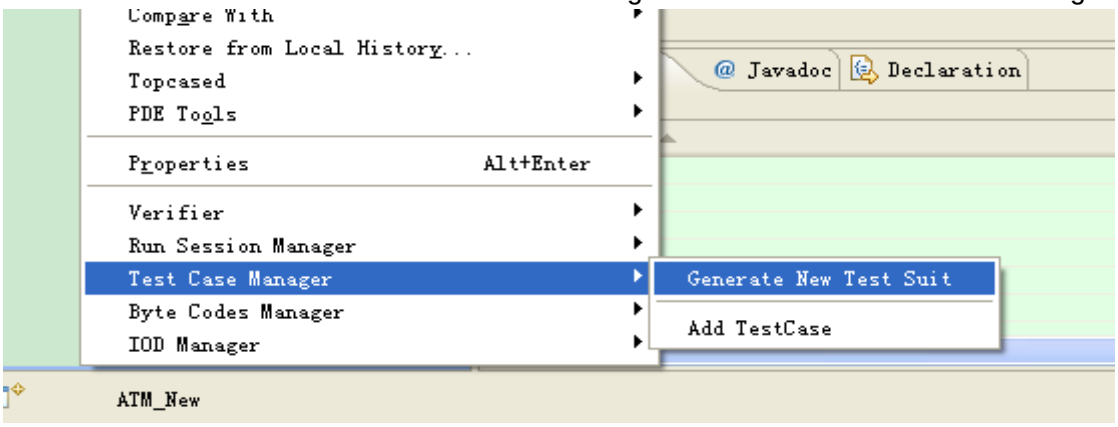
An Interaction Overview Diagram consists of diagrams at two levels. The first level is an activity diagram. The second level is a sequence diagram. One can jump from the activity diagram to the sequence diagram by double clicking one node.



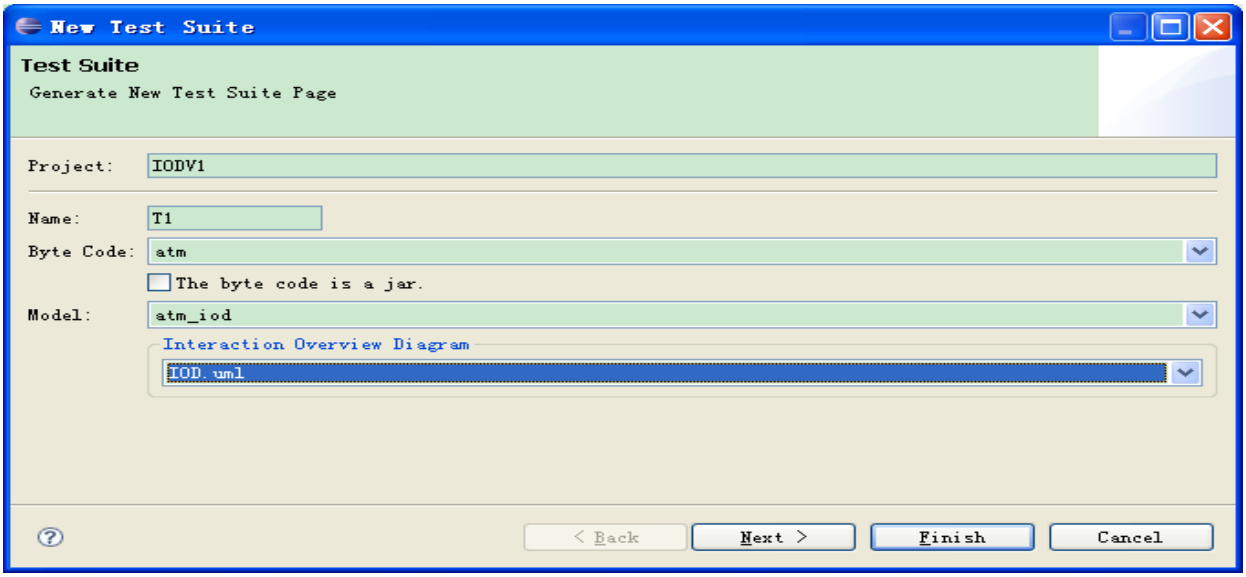
3.4 Create a Test Suite

For the ease of management, test cases that share the same byte codes, the same interaction overview diagrams and the same event-method matching are grouped in one test suite. Test cases belonging to the same test suite may differ from their execution

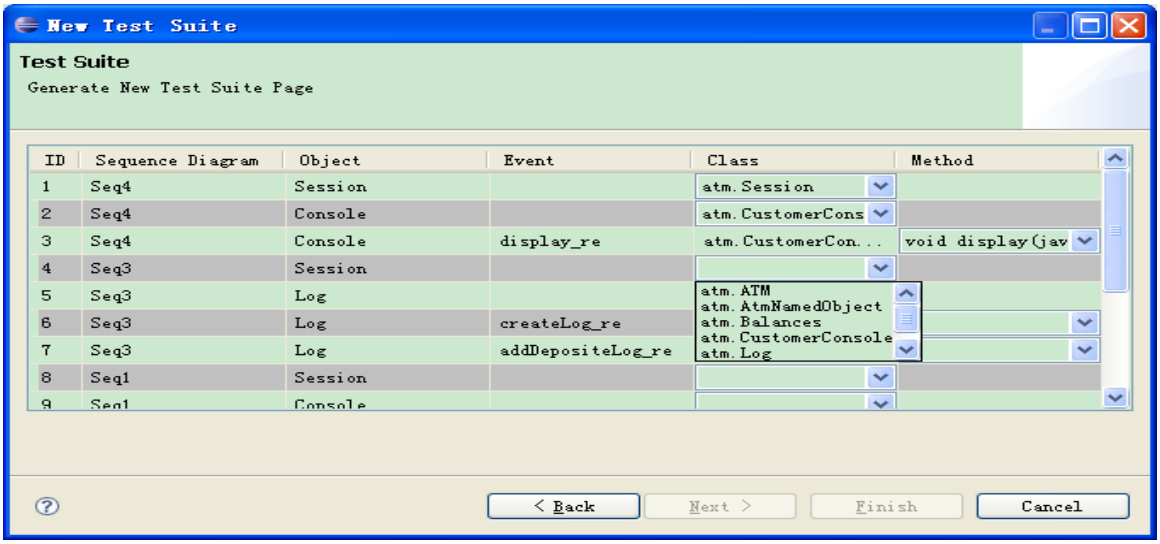
configuration. Thus, before creating any test case, a test suite must be set up first. Click the 'Generate New Test Suite' item in the right click menu 'Test Case Manager'.



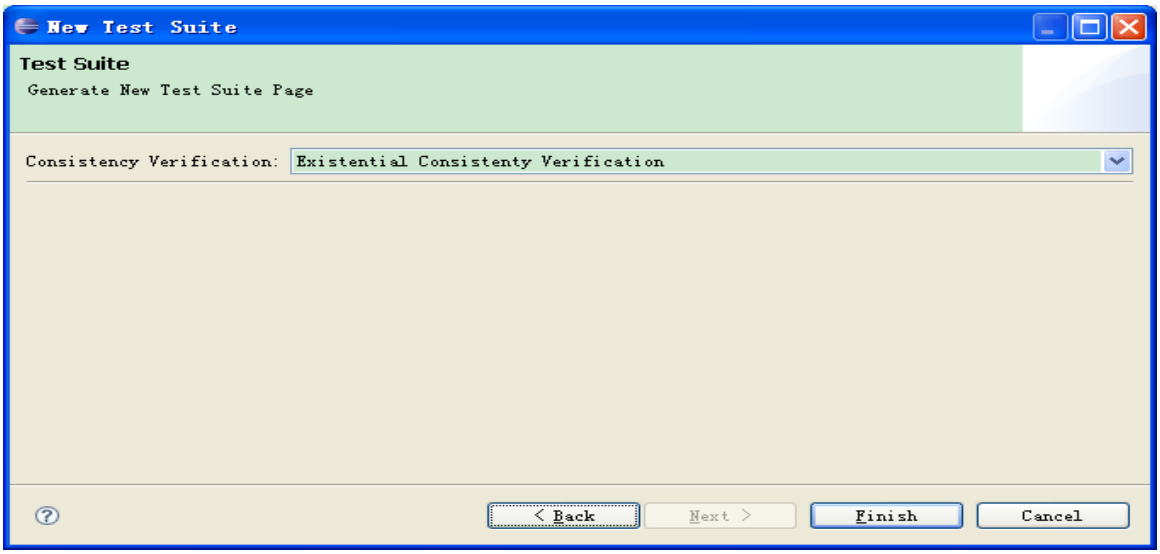
Select the Byte codes and interaction models used for verification.



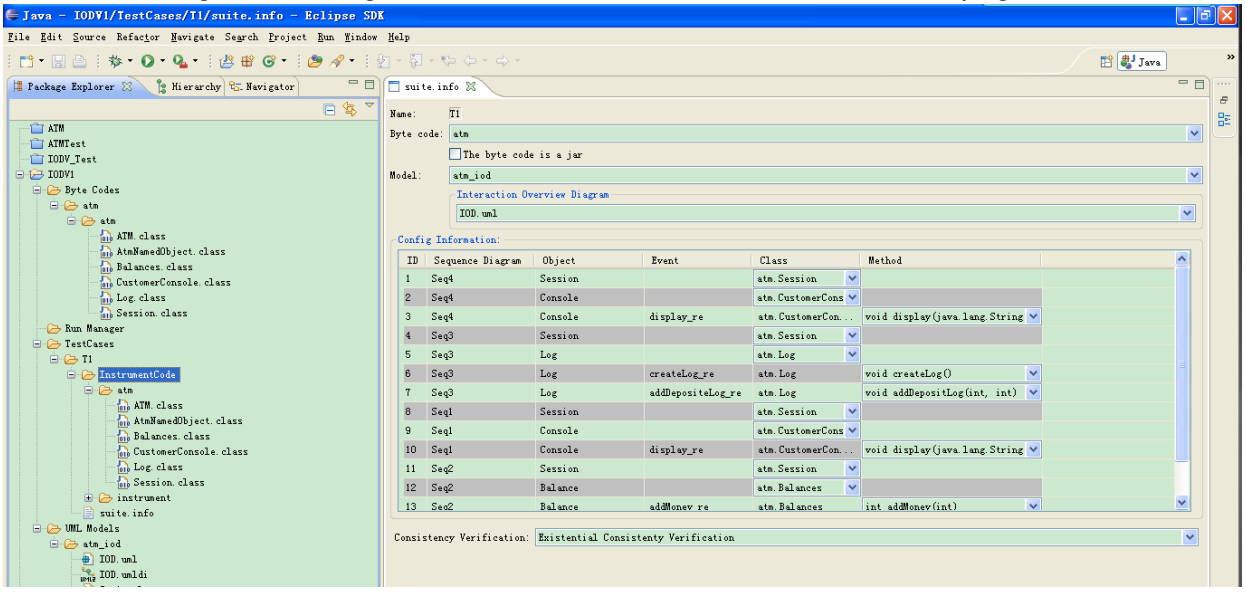
Click the next and match the event and method in the next page.



Specify the type of verification and click finish to complete the setup.

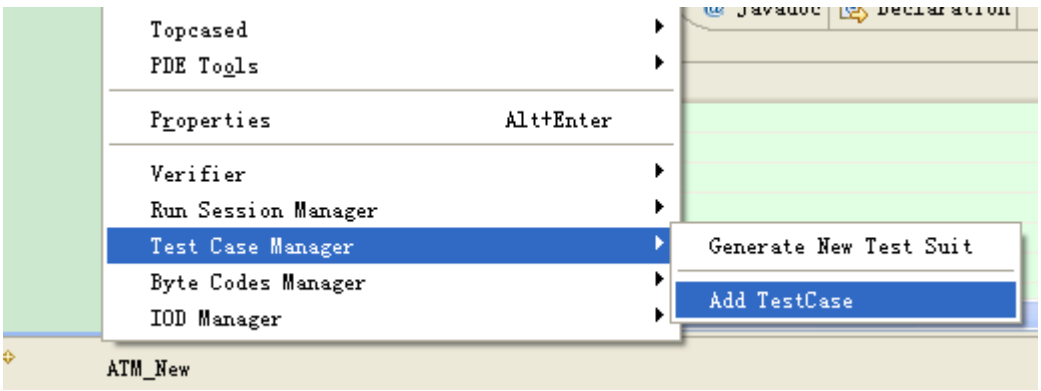


One can change the configuration of test suite in the test suite edit page.

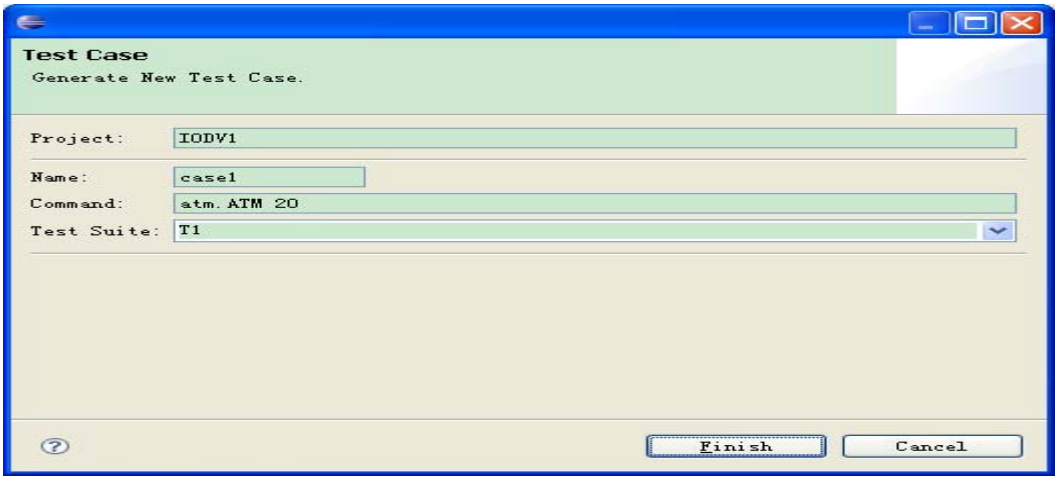


3.5 Create Test Cases

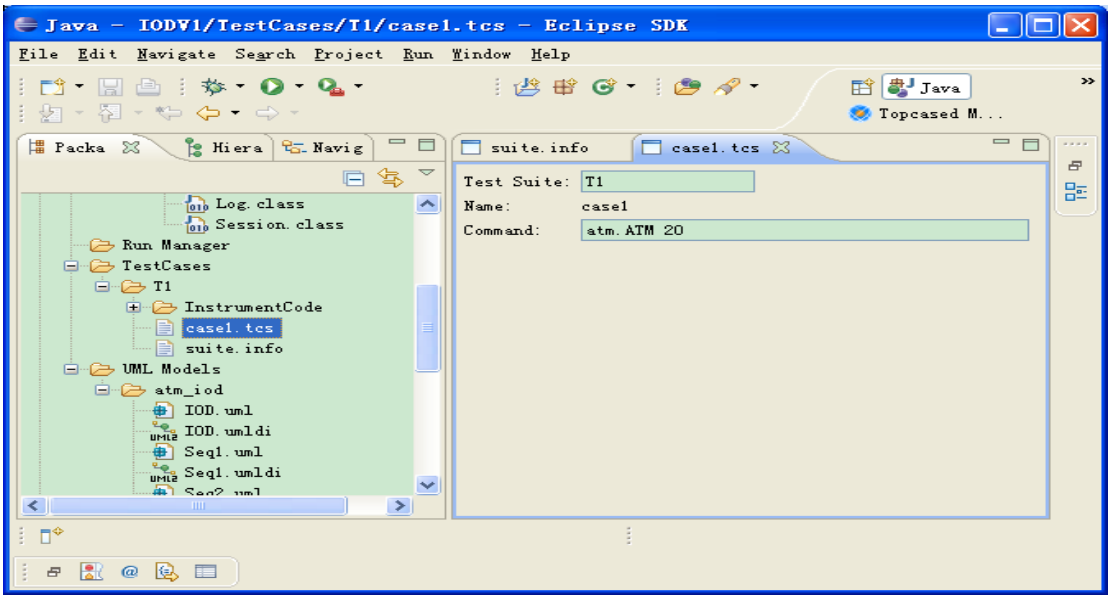
Test case manager help users to set the execution configuration of test cases. Right click the ‘Add testcase’ item in the right click menu’ Test Case Manager’.



In the popup windows, fill the command (including input arguments) used to start the execution of such test case.

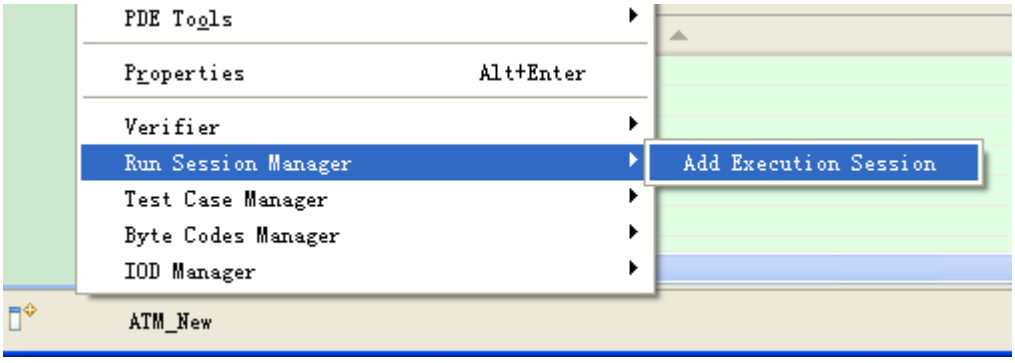


Test cases may be reconfigured later by the test case edit page.

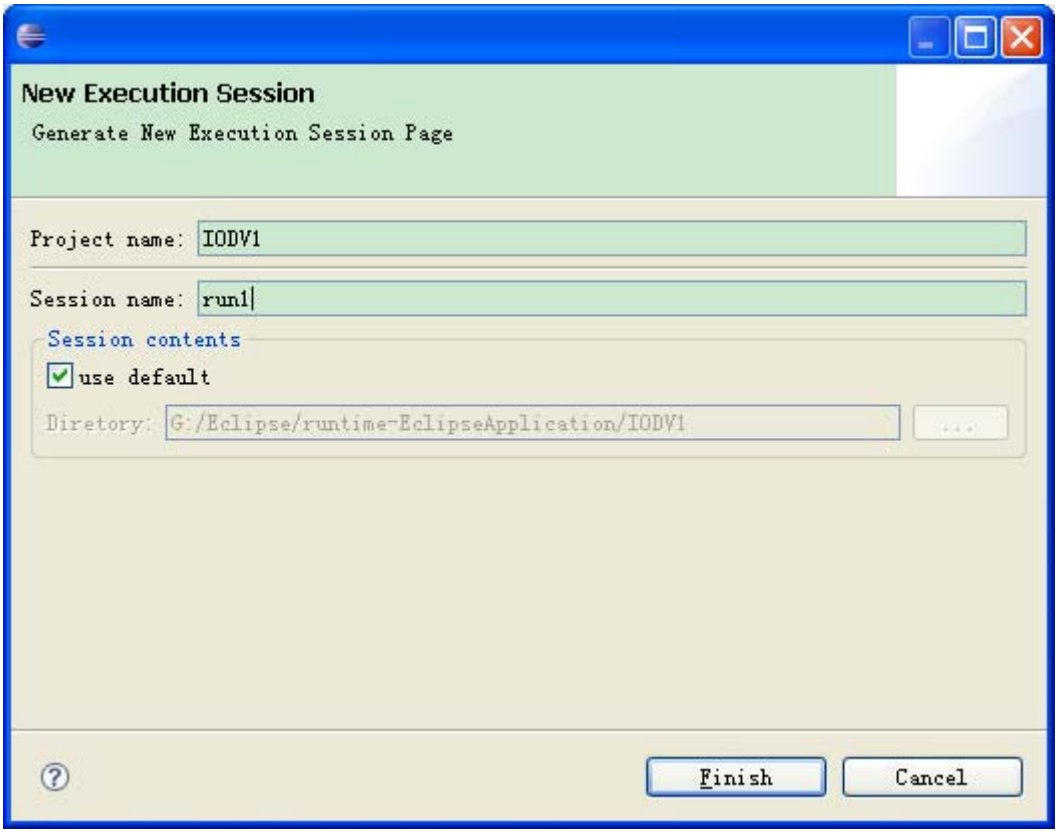


3.6 Execute Test Cases

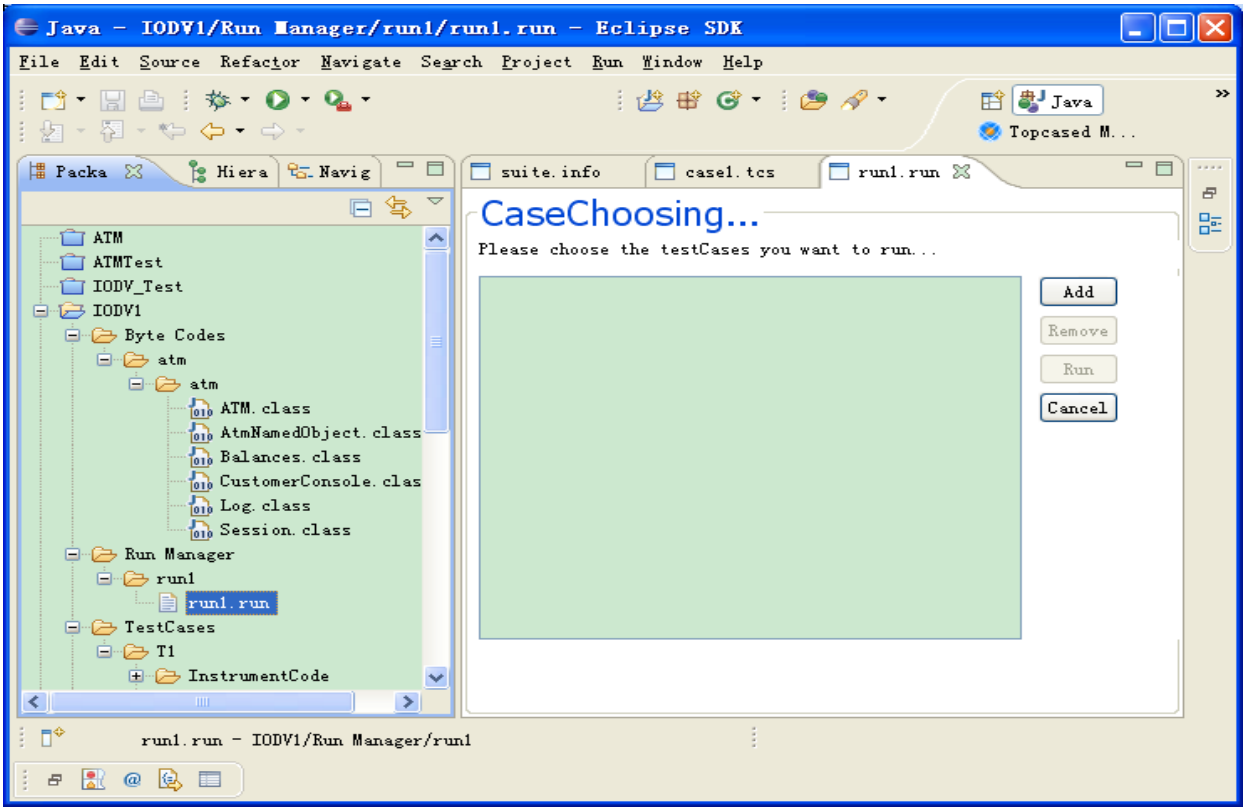
Test cases are grouped in an execution session before run-time execution.
Click the ‘Add Execution Session’ item in the right click menu ‘Run Session Manager’.



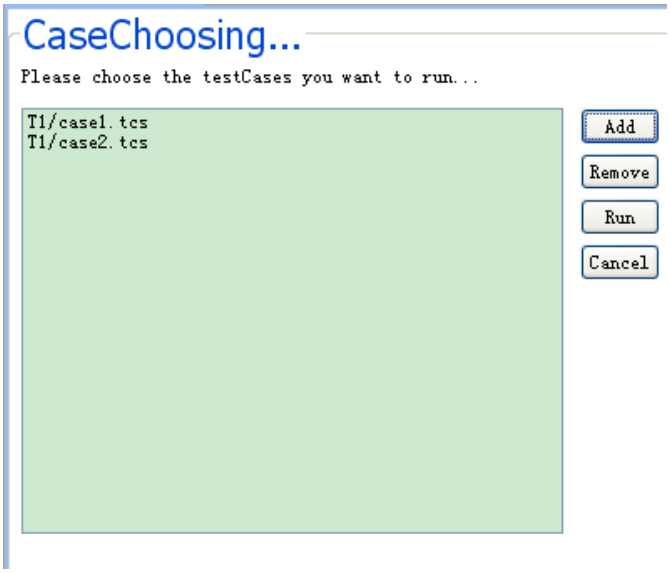
Type the name of execution session.



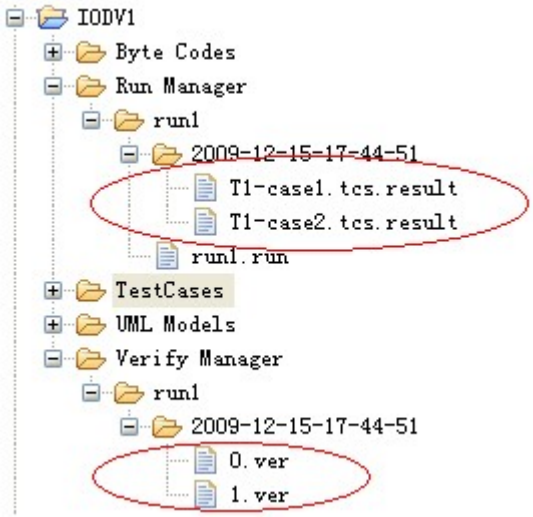
Add test cases into the session by the Case choosing page.



An execution may contain many test cases.

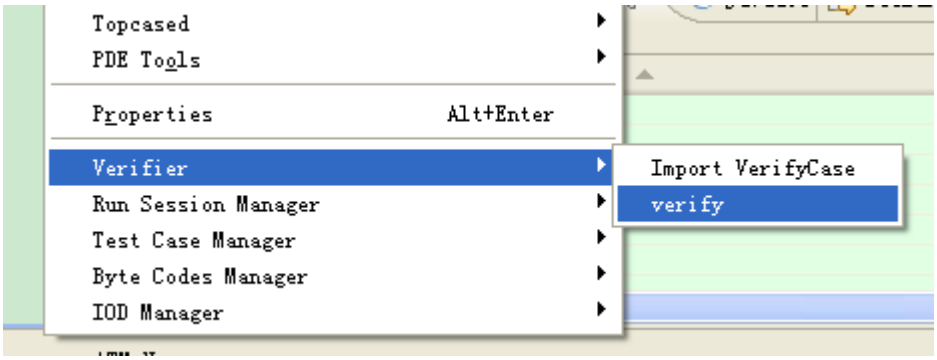


After execution, trace files are collected case by case. Meanwhile, coverage information is also generated in the verification folder.

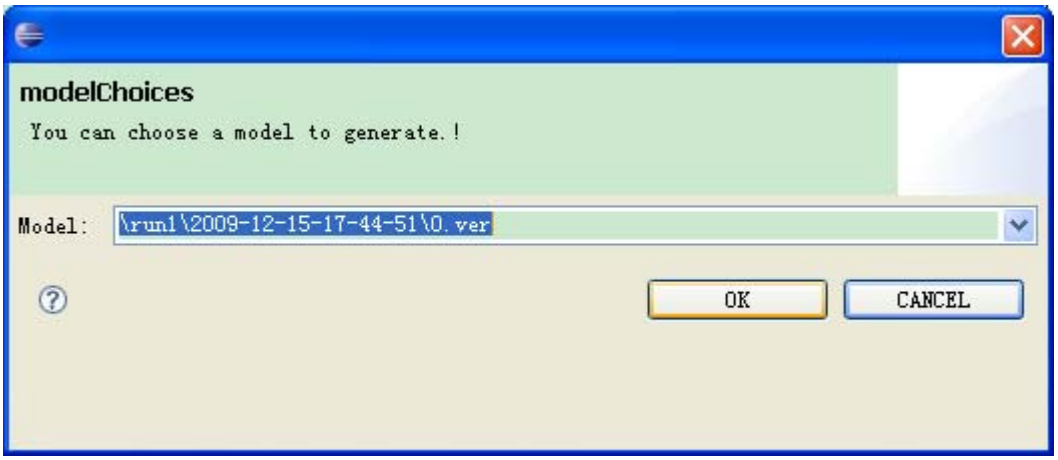


3.7 Generate Verification Results

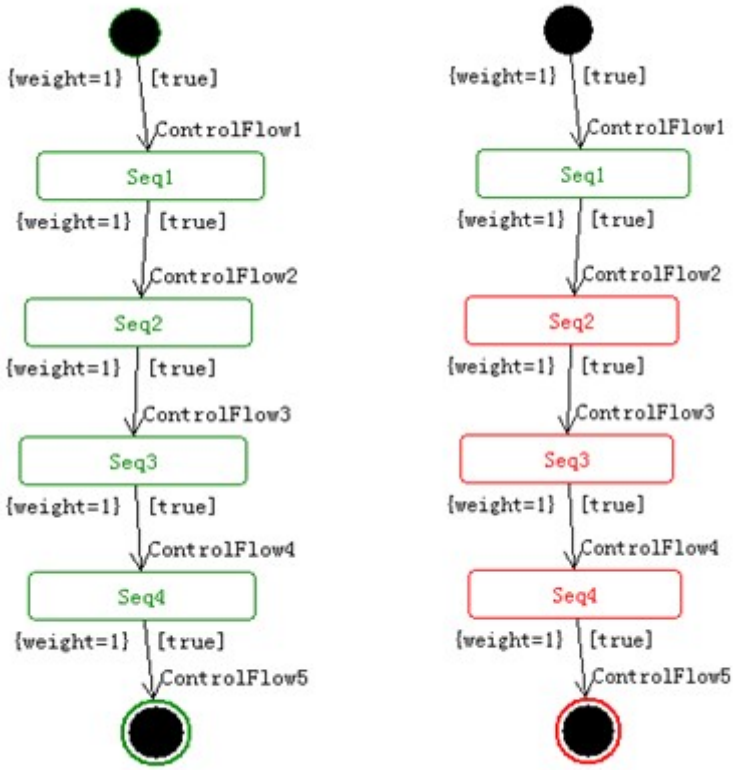
To see the verification result, click the ‘verify’ item in the ‘Verifier’.

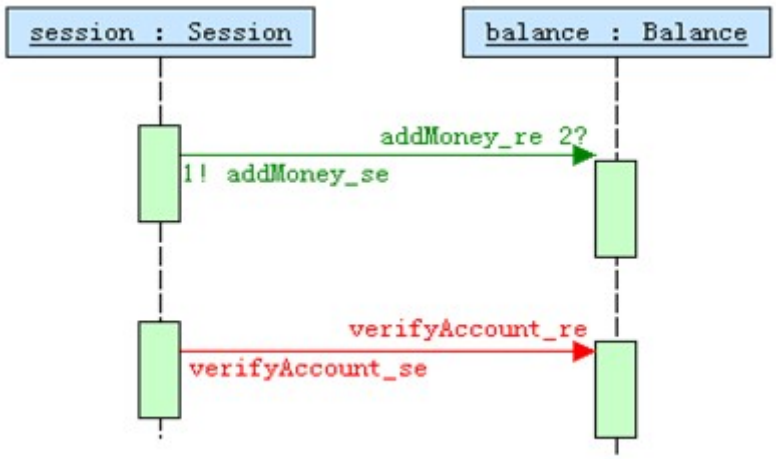


Choose the specification model.



Correct verifications are shown in green while failed verifications are shown in red both in the activity diagram and in the sequence diagram.





Appendix A.

The dependent plug-ins is available on the following URL.

emf-sdo-xsd-SDK-2.4.2
http://www.eclipse.org/downloads/download.php?file=/modeling/emf/emf/downloads/drops/2.4.2/R200902171115/emf-sdo-xsd-SDK-2.4.2.zip

emf-transaction-SDK-1.2.3
http://www.eclipse.org/modeling/download.php?file=/modeling/emf/transaction/downloads/drops/1.2.3/R200902102018/emf-transaction-SDK-1.2.3.zip

emf-validation-SDK-1.2.1
http://www.eclipse.org/modeling/download.php?file=/modeling/emf/validation/downloads/drops/1.2.1/R200808130931/emf-validation-SDK-1.2.1.zip

GEF-SDK-3.4.2
http://archive.eclipse.org/tools/gef/downloads/drops/R-3.4.2-200902171642/GEF-SDK-3.4.2.zip

gmf-sdk-2.1.3
http://www.eclipse.org/modeling/download.php?file=/modeling/gmf/gmf/downloads/drops/2.1.3/R200902181028/gmf-sdk-2.1.3.zip

mdt-ocl-SDK-1.2.2
http://www.eclipse.org/modeling/download.php?file=/modeling/mdt/ocl/downloads/drops/1.2.2/R200809160927/mdt-ocl-SDK-1.2.2.zip

mdt-uml2-SDK-2.2.2
http://www.eclipse.org/modeling/download.php?file=/modeling/mdt/uml2/downloads/drops/2.2.2/R200902101440/mdt-uml2-SDK-2.2.2.zip

org.topcased.sdk-2.5.0
http://gforge.enseeiht.fr/frs/download.php/2157/org.topcased.sdk-2.5.0.zip

epfc-richtext-1.5.0.2
http://archive.eclipse.org/technology/epf/composer/1.5/release/epfc-richtext-1.5.0.2-win32.zip

net.sourceforge.lpg.lpgjavaruntime_1.1.0.v200803061910.jar
http://repository.grepcode.com/java/eclipse.org/3.5/plugins/net.sourceforge.lpg.lpgjavaruntime_1.1.0.v200803061910.jar

dom4j-1.6.1.jar
http://www.java2s.com/Code/JarDownload/dom4j-1.6.1.jar.zip