Test Execution Document

DaCoPAn2

Helsinki, 7th May 2005

Software Engineering Project
UNIVERSITY OF HELSINKI
Department of Computer Science

Course

581260-4 Software Engineering Project (6 cr)

Project Group

Mikko Airaksinen
Tomi Korkki
Pauli Miettinen
Timo Tuominen
Mikko Väänänen

Customer

Markku Kojo

Project Masters

Juha Taina (Supervisor)
Marianne Korpela (Instructor)

Homepage

`http://www.cs.helsinki.fi/group/dacopan2`

Change Log

Version	Date	Modifications
1.0	27.05.2004	First version
1.1	29.05.2005	Initial version for DaCoPAn2
1.5	03.05.2005	Beta version for DaCoPAn2
2.0	06.05.2005	Finished version for DaCoPAn2

Contents

1 Introduction

This is the Test Execution Document of the DaCoPAn2 Software Engineering project at the Computer Science Department of the University of Helsinki. The DaCoPAn2 project is a follow-up project for the DaCoPAn project. Its main goal is to improve and expand the Animator subsystem produced by the DaCoPAn group.

This document presents the results of the execution of the different tests presented in the DaCoPAn2 Test plan document [1]. The document is organized as follows:

- Section 2 introduces the results of unit testing.
- Section 3 introduces the results of integration testing.
- Section 4 introduces the results of system testing.
- In section 5 the bugs found during test phase are listed.

The results of the test execution will be presented in subsection as follows:

- Id and name of the executed test.
- Description of the test, or collected results.
- Errors found if any, and maybe a reference to the bug and fixes report. If error was found on certain step of Test Case, the step number is given.

All test cases were first executed once and found bugs were fixed. Then all test cases were executed again and no further bugs were found.

Test were run on following environment: Intel® Pentium® 4 CPU 2.66GHz processor, Computer Science Linux #2.152, kernel version 2.6.11-1.14_FC3.csl and Java™ 2 Runtime Environment, Standard Edition (build 1.5.0-b64), Java HotSpot™ Client VM (build 1.5.0-b64, mixed mode, sharing). Program was also tested to work with Microsoft® Windows Server™ 2003 Standard Edition with same Java™ Runtime Environment.

2 Unit testing

Unit tests were created as JUnit tests. Not all classes were tested as there were lots of user interface classes. It was thought to be unnecessary to test those classes. The tested classes, the functions of classes and JUnit test are listed in table 1.

JUnit tests were executed using Eclipse 3.0.1 and JUnit version 3.8.1. Contrary to the Test plan, RITA was not used for running the unit tests, because the software was found to be unsuitable for this kind of user interface oriented project, where unit testing and other black box testing can never completely exhaust all the different

user input combinations and even statement coverage could not be sensibly achieved for all classes. In addition, since RITA was somewhat buggy, the labour required to configure it was seen as out of proportion compared to the benefit gained from it.

The JUnit tests were run once, discovered problems fixed and then run again to detect any new problems that might have appeared. No problems were found, as can be seen from Figure 1 at page 3.

Table 1: List of JUnit tests executed.

Class		
Package	File	Test case
contsig	AnimationTimeState.java	AnimationTimeStateTest.java
model	DataView.java	DataViewTestCase.java
	ENCTreeModel.java	EncTreeModelTestCase.java
	Layer.java	LayerTestCase.java
	NoteManager.java	NoteManagerTestCase.java
	Note.java	NoteTestCase.java
	ScenarioStepIterator.java	ScenarioStepIteratorTestCase.java
	TransferUnit.java	TransferUnitTestCase.java
pef	XMLProtocolEventsReader.java	XMLProtocolEventsReaderTestCase.java
scenario	ObjectSerializer.java	ObjectSerializerTestCase.java
	ScenarioFile.java	ScenarioFileTestCase.java
	Saveable.java	ScenarioSavingTestCase.java
settings	GeneralSettings.java	GeneralSettingsTest.java
ui	CalcYCoord.java	CalcYCoordTestCase.java
	ENCPanel.java	ENCPanelTest.java
	MainFrame.java	MainFrameTestCase.java
	SettingsMSC.java	SettingsMSCTest.java
ui.tsc	NoticeTrigger.java	NoticeTriggerTestCase.java
	SettingsTSC.java	SettingsTSCTest.java

While executing test `TrasferUnitTestCase.java` bug 5 was found and later fixed.
While executing test `MainFrameTestCase.java` bug 6 was found and later fixed.

3 Integration testing

In this section, the tests that test the integration of the Animator modules and integration between the Analyzer and Animator are presented.

Figure 1: Screenshot from Eclipses JUnit test panel.

Test case 1 – ProtocolEventsReader and Dataview

- DESCRIPTION
  This test was run to ensure that the Animator ProtocolEventsReader subsystem works correctly. JUnit test case `XMLProtocolEventsReaderTestCase.java` was run to ensure that.
- ERRORS FOUND
  No errors or failures were collected.

Test case 2 – Analyzer generated PEFs compliance to the Animator XML reader

- DESCRIPTION
  This test was run to test that the Animators XML reader can correctly read Analyzer generated PEFs. It was done by loading a file named `pef_4_2.pef` and verifying visually that the PEFs contents corresponds to the animation in both MSC and TSC modes.
- ERRORS FOUND
  Full correspondence was verified and no problems occurred during the reading phase of the PEF.

4 System testing

This section contains the largest part of testing the DaCoPAn Animator—the system testing. It contains all those tests that are needed to verify that all subsystems of the DaCoPAn Animator work correctly.

Test case 3 – Settings and user interface visualization in MSC

- DESCRIPTION
  The test case was conducted with `pef_4_2.pef`
- ERRORS FOUND
  Bug 1 was found during step 5. Otherwise the results were as expected.

Test case 4 – MSC buttons and encapsulation

- DESCRIPTION
  The test case was conducted using `pef_iota_zeta_1.pef`.
- ERRORS FOUND
  Bug 2 was found during step 9. No other errors were found.

Test case 5 – Settings and user interface visualization in TSC

- DESCRIPTION
  The test case was conducted using `pef_sack.pef`.
- ERRORS FOUND
  Bug 1 was found during step 5. Bug 3 was found during step 14. Bug 4 was found during step 17.

Test case 6 – TSC buttons and sliders

- DESCRIPTION
  The test case was conducted using `pef_sack.pef`.
- ERRORS FOUND
  No problems were discovered.

Test case 7 – Localization

- DESCRIPTION
  The test case was conducted using various PEF files.
- ERRORS FOUND
  When loading `pef_sack.pef`, bug 10 was found. In addition, some strings were not correctly localized. They are listed in bug 9.

Test case 8 – Help menu

- DESCRIPTION
  This test was to make sure that the "Help" menu works correctly and is possible to invoke before loading any PEF files.
- ERRORS FOUND
  The help menu was shown without any problems, but its contents had bugs that are listed in bug 7.

Test case 9 – Scenario and explore modes

- DESCRIPTION
  This test was to make sure that MSC's scenario and explore modes work correctly. The test was executed by loading a PEF file named `pef_4_2.pef` and creating and running the scenario.
- ERRORS FOUND
  Scenario mode worked as it should, but unlike the test case states, pressing "Encapsulation" at recording mode also inserts encapsulation at play list. This was, however, considered to be a feature, not a bug.

Test case 10 – TimePanel

- DESCRIPTION
  This test was to ensure that the Time panel works correctly in MSC mode. The PEF named `pef_3_1.pef` was loaded and played in MSC mode.
- ERRORS FOUND
  No problems were perceived.

Test case 11 – Note panel in MSC

- DESCRIPTION
  This test was to ensure that the "Note panel" in the MSC mode works correctly. The PEF named `pef_3_2.pef` was loaded and the test case was executed.
- ERRORS FOUND
  Adding, editing and deleting the notes worked fine, but bug number 8 was found.

Test case 12 – Note panel in TSC

- DESCRIPTION
  Used `pef_4_2.pef` in testing.
- ERRORS FOUND
  No errors were found.

Test case 13 – Visibility of packets in UFO

- DESCRIPTION
  Used `pef_4_2.pef` in testing.
- ERRORS FOUND
  No errors were found.

Test case 14 – Save all settings and restore defaults

- DESCRIPTION
  This test was run to make sure that saving and restoring settings work correctly. The PEF files used were `pef_5_2.pef` and `pef_3_2.pef`.
- ERRORS FOUND
  The test case was executed as it should and no errors were found.

Test case 15 – TSC and PEFs without TCP layer or sequence numbers

- DESCRIPTION
  Loaded `pef_2_1.pef` (no tcp data) and switched to TSC mode. A correct behavior was detected. Generated a copy of `pef_4_2.pef` and commented out a single row (sequence number from a single transfer unit). An exception was thrown and the TSC view did not show any data (was completely gray).
- ERRORS FOUND
  Bug number 11 was found.

Test case 16 – Text visibility

- DESCRIPTION
  This test was to ensure that all texts and their background have clearly distinguishable colors. The PEF file `pef_6_1.pef` was loaded and all animator components were scanned through.
- ERRORS FOUND
  All texts—especially the text "Message Sequence Chart"—were distinguishable from their background colors.

Test case 17 – Animation panel

- DESCRIPTION
  The test was conducted using the file `pef_4_2.pef`.
- ERRORS FOUND
  The results were exactly as expected.

Test case 18 – Opening animator files

- DESCRIPTION
  This test was to ensure that the "Open File" dialog shows all and only all supported file types by default. The following dummy files were created in the default directory:
  - `puppua.tex`
  - `puppua`
  - `puppua.java`
  - `.puppua`

  to execute the test case.
- ERRORS FOUND
  None of the created files were visible in the file chooser dialog by default. Instead all files that had `.sce` or `.pef` suffix were visible. Thus the test case was executed without any errors.

Test case 19 – Animation readability at the auditorium

- DESCRIPTION
  This test case was run to ensure that the Animator is usable for educational purposes, especially when used in auditorium context. The test was run at the Exactum building in auditorium CK112 using CSLinux and data-projector.
- ERRORS FOUND
  The default font size for the MSC mode was found to be too small, but enlarging the font size at settings helped as it should. In TSC mode some default colors were not very readable, but color chooser helped. Also dropped cross in the TSC mode was not very readable. This was considered as a feature, not as a bug.

5 Found and fixed bugs

In this section, the bugs we found during the first execution of test cases are listed. Also the fixes for bugs are listed. All bugs listed here are fixed.

BUG 1

Description: When protocol header fields were deselected in the MSC settings after pressing "Apply", these changes were reflected in the drawing area when the settings window was dragged on top of it, and remained even if the "Cancel" button was pressed.
In fact, after pressing "Apply" once, any changes made to layer-specific settings in the MSC as well as Graphical elements and Animation settings in the TSC had the same effect.
Fix: `MainFrame` was changed to use the copy constructor of the `GeneralSettings` class.

BUG 2

Description: When switching to the Application layer after loading the file `pef_iota_zeta_1.pef`, the following exception was thrown: "Exception in thread 'AWT-EventQueue-0' java.lang.IllegalArgumentException: timeScale cannot be negative or zero!" meaning that the `AnimationTimeState` object received an illegal `timeScale` value because the Application layer is empty for `pef_iota_zeta_1.pef`
Fix: Added test for non-positive time scale arguments in `MainFrame.java`. Should one occur, time scale is set to 1.

BUG 3

Description: The chosen notice delay was applied not for the unit which triggered the notice but for the next unit. It was also applied for each following unit even when they triggered no notices.
Fix: The notice delay is now applied only to units with notices.

BUG 4

Description: The unit variables shown in the Unit info panel were only updated after selecting a unit with the mouse or when a new unit was drawn during animation. The changed settings were not reflected for an already drawn and active unit.
Fix: The text display is now updated at every repaint.

BUG 5

Description: If a PEF file does not have any variables, `getPayloadSize()` in class `TransferUnit` throws null pointer exception error.
Fix: Added null test for all variables.

BUG 6

Description: If `TimePanel` has null as first parameter, it throws null pointer exception error.
Fix: Added null test for first parameter

BUG 7

Description: The contents of menu Help → About are not correct. General information in the panel with same name is out of date and the licence information in the respective panel wraps words ugly.
Fix: Added information about DaCoPAn2 project and added `setWrapStyleWord(true)` to `UserInterface::showAboutBox()`.

BUG 8

Description: Sometimes when in MSC mode and the progress line is after a note, when "Stop" is pressed the note text is disappears. Also when animation is stepped through with "Step forward" or "Step backward", note texts are not showed in synchron.
Fix: Method `stepTo()` in class `NotePanel` was fixed s.t. it uses the correct time to show note texts.

BUG 9

Description: The following strings were not localized:

- TSC Settings: “Displayed element”
- TSC Settings: “Highlight units”
- TSC Settings, Notice Dialog: “Value must be a number”
- TSC Notices: “disabled”
- TSC Error dialog header: “Message” (Appears when file contains no TCP data.
- TSC: Time axis scale unit (“ms” or “s”)
- MSC: “Variables” column header
- General settings: panel labels
- TSC Color palette: whole component
- File dialog: whole component

Fix: The strings were externalized. The Java components `JColorChooser` and `FileDialog` could not be localized.

BUG 10

Description: When switching to TSC mode with a valid pef file containing TCP information, the TSC incorrectly displayed the error dialog which claims that no TCP data is available. This was discovered with a `dacopan.properties` localization file containing dummy strings. Also, when trying to select the header fields for the TSC Unit info panel in the settings, the list of header fields was empty.
Fix: The TSC tried to look for the TCP protocol using its localized name. It was fixed to use the protocol's raw name.

BUG 11

Description: A TCP transfer unit without a sequence number throws a `java.lang.NumberFormatException: null` when trying to show it using TSC
Fix: A null check and error message dialog was added.

BUG 12

Description: While in play mode the TSC Unit Info panel does not update when a new unit is displayed
Fix: See fix in 4

BUG 13

Description: Notices for SACK are not triggered
Fix: The Exists test for SACK variables was implemented.

References

1 DaCoPAn2 Software Engineering Project, *Test plan*. Relase 2.0. University of Helsinki, March 2005.