

# **User Guide**

## **Baile V&V Prototype**

**Felipe M. Besson**  
**Pedro M. B. Leal**

Department of Computer Science  
Institute of Mathematics and Statistics  
University of São Paulo (USP)

`{besson,pedrombl}@ime.usp.br`

## 1. Overview

This software prototype is part of HP Baile Project<sup>1</sup>, more specifically, this prototype belongs to the research line “Verification & Validation of Choreographies”. More details about this it and the basis of our software can be found in this technical report [2]. Our prototype is composed by:

- a web service choreography developed on OpenKnowledge(OK)<sup>2</sup>;
- JUnit test cases developed for the coreography testing
- *Ad hoc* scripts for enacting and testing the choreography automated

In this user guide, we described in steps how to install and use the software.

## 2. Prerequisites

The following software elements must be installed and working:

- Java 6 [3]
- SQLite 3 [4]
- Ant [1]

## 3. Where to download ?

The prototype, web services used and the REST client can be downloaded at the download section of Baile page (research line: Verification and Validation of Choreographies). The source code of these softwares are also be available on the Baile-V-V GitHub Repository.

## 4. Software directory structure

Our prototype is structure in the following files and directories (just the most important are described bellow):

file/directory	Description
baile.sh	script to start the application
src/	source code
test/	test code
bin/	binary files
config/	OpenKnowledge configuration files
lcc/	OpenKnowledge interaction model
lib/	third-party dependencies
log/	log files
test-reports/	html test reports

---

<sup>1</sup>Baile Project: <http://ccsl.ime.usp.br/baile/>

<sup>2</sup>OpenKnowledge: <http://www.openk.org/>

## 5. How to use ?

In this section, we detailed the main steps to interact with the application.

1. Start the script **baile.sh**, and then, the application will start:

```
besson@metropolis:~/BookTripWSChor$ ./baile.sh
Baile V&V - dez/2010

Commands:
start_chore - start the choreography
stop_chore - stop the choreography
run_unit-tests - run the unit tests
run_integration-tests run the integration tests
run_acceptance-tests - run the acceptance tests
run_all-tests - run all tests
exit - stop the choreography if it is running and quit
help - show this message

Contacts:
Felipe Besson <besson@ime.usp.br>
Pedro Leal <pedrombl@ime.usp.br>

baile$ █
```

Figure 1. Welcome screen

2. In the prompt, start the choreography by typing **start\_chore** (as showed in Figure 2)

```
baile$ start_chore
Determining localhost's IP... 192.168.1.100
Stopping currently running instances... Done
Compiling components... Done
Publishing the web services... Done
Starting message trace queue... Done
Launching the Discovery Service... Done
Publishing the Interaction Model... Done
Launching the traveler... Done
Launching the travel agency... Done
Launching the airLine... Done
Launching the acquirer... Done
Setting up roles... Done
Choreography started.

baile$ █
```

Figure 2. Starting the choreography

3. Once all components of choreography (services, roles, and OpenKnowledge entities) have been started successfully (as showed in Figure 2), the tests can be executed. Test can be applied by using the following commands:

Command	Description
run_unit-tests	run the unit tests
run_integration-tests	run the integration tests
run_acceptance-tests	run the acceptance tests
run_all-tests	run all tests

4. After test execution, some JUnit html reports containing the test results are automatically generated. These reports are saved in the test-reports directory. The Figure 3 presented a html report generated after some unit tests execution.

Unit Test Results.						
						Designed for use with <a href="#">JUnit</a> and <a href="#">Ant</a> .
<b>Class br.usp.ime.test.unit.AcquirerWSTest</b>						
Name	Tests	Errors	Failures	Time(s)	Time Stamp	Host
AcquirerWSTest	6	1	0	21.899	2010-12-21T04:06:13	metropolis
<b>Tests</b>						
Name	Status	Type	Time(s)			
shouldAddNewAccount	Success		3.796			
shouldAddAndRetrieveTheAccount	Error	expected:<[1234]Jacob[]1000> but was:<[1234]Jacob[s]1000>	2.246			
shouldApproveCreditCard	Success		2.204			
shouldNotApproveCreditCard	Success		0.087			
shouldDiscountValueCorrectly	Success		4.390			
shouldNotDiscountValue	Success		4.219			

Figure 3. HTML Report for unit test results

## 6. How to add more test cases ?

The script compiles every test class of the test folder before any test execution. So, modifications on a existing test class, will be compiled before the next command test execution.

Any new test class needs to be added in its test folder (unit, integration, or acceptance). Once added, the script will recognised the new class automatically.

## 7. Known Issues

Our choreography does not support interactions interruptions properly. So, if any test was forced interrupted, in some cases, the choreography must be restarted. There are some crash but rare communication errors. When these erros happened, the processes that compose the choreography must be killed. It can be done executing the script `./script/stopChor.sh`. Please contact us about any new issue.

## 8. Acknowledgements

This research received funding from HP Brasil under the Baile Project.

## References

- [1] Apache Ant. Java library and command-line for automated building. Available on: <http://ant.apache.org/>, 2010.
- [2] F. M. Besson, P. M. B. Leal, and F. Kon. Towards verification and validation of choreographies. Technical Report RT-MAC-2011-01, Institute of Mathematics and Statistics - University of Sao Paulo (USP), 2011. Available on: <http://ccsl.ime.usp.br/baile/Downloads>.
- [3] Oracle. Java se downloads. Available on: <http://www.oracle.com/technetwork/java/javase>, 2010.
- [4] SQLite. Sql database engine. Available on: <http://www.sqlite.org>, 2010.