Project acronym: MODELPLEX

Project full title: MODELling solution for comPLEX software systems

Contract n° 034081

Workpackage 4: Verification and Validation

Deliverable D4.3.c: Verification Tools (EXTRACT)

Project co-funded by the European Community under the "Information Society Technology" Programme

Behavioural Consistency Checker (BCC)

This section presents the Behavioral Consistency Checker User's Guide.

1.1. Installation

It should be noted that for this first prototype version, packaging is incomplete as the full SVT (Simulation, Verification and Test) workbench (T4.1) is not yet operational. A version packaged as a ModelBus service is registered in the ModelBus service registry maintained by Fraunhofer Fokus based at:

http://monarch.fokus.fraunhofer.de:8080/WebRegistryClient/

This version will serve as basis for integration in the SVT workbench in further deliverables, so the tool will be available as part of an Eclipse plugin.

1.1.1. Prerequisites

The standalone java application requires a JRE >= 1.5, and packages its dependencies to some standard libraries (org.eclipse.emf, org.eclipse.uml2) so there are no additional requirements.

Optional setup of the verification tool suite server CPN-AMI 3.2 requires a machine running Linux or MacOSX. This can be a different machine than the one running the actual client tool.

The input models need to be available in standard EMF compliant UML 2.1 format, usually bearing the extension ".uml". To know whether this is the case, try opening the model using the Eclipse built-in UML model editor, if the tree view does not open correctly, the BCC tool will also have parsing issues with the model.

1.1.2. Distribution Package

The BCC package is available from:

https://www.modelplex.org/svn/work_area/WP4/T4.3 Model based verification/LIP6-BCC/BCC-1.0.zip.

The standalone version is packaged as a zipped archive containing an executable jar file, the required libraries, and some examples taken from WP1 case study material.

By default BCC is configured to access the publicly available test CPN-AMI server hosted at LIP6, but the user can follow these instructions to download and install their own server if necessary.

OPTIONAL:

To install CPN-AMI on another server (Linux or MacOS X), download the package at:

- For Linux: http://www.lip6.fr/cpn-ami-download/CPN-AMI-3.2.tar.gz
- For Mac PPC: http://www.lip6.fr/cpn-ami-download/CPN-AMI-PPC-3.2.dmg
- For Mac Intel: http://www.lip6.fr/cpn-ami-download/CPN-AMI-X86-3.2.dmg

Then decompress the archive and follow the instructions of the README file to compile and run the server. The homepage at http://move.lip6.fr/software/CPNAMI/ provides additional information on the contents of this distribution.

1.1.3. Installation Instructions

Simply unzip the archive BCC-1.0.zip in a folder. Since this is a standalone java program, no additional installation is required.

To execute simply run "java –jar bcc-1.0.jar" in the root folder of the unzipped archive or double-click this file according to your platform. A simple Swing GUI should open.

The default configuration is set up to access a publicly available CPN-AMI instance, running on host hephaistos.lip6.fr on port 7001. The default user is "modelplex" and the associated password is "123456". These settings can be changed through the "Options->CPNAMI Settings". The menu entry "Options->Save Settings" saves both these connection settings and the default model chosen at startup.

1.1.4. Validating Installation

To check that the installation is operational, try running the tool on one of the example models available included in the "models/" folder of the distribution. The tool should point at a model taken from SAP examples for WP1 included in the distribution. Simply hit the "Transform button" to analyze the UML model provided and generate the underlying Petri net representation that is used for verification.

For each activity diagram or state machine present in the source file, a corresponding net is created. Then left-click the diagram you wish to analyze in the transformation products panel, and hit the "Check Consistency" button.

The service may take some time to run especially if the model is very large, be patient. When run in console mode, the tool displays some traces and indicates progress by producing a line of dots ".".

The analysis results are then presented at the bottom of the GUI, with indication of the severity, the source of the error, and a text describing the error. This particular model of a Sales Order Process is full of errors, it displays unbounded behavior that affects practically the whole activity diagram. The tool should produce a lot of red error messages if everything is installed correctly.

1.2. BCC User's Guide

The tool in its present form requires very little user expertise to run, as we have limited configuration to the essential: the input file name.

Simply choose your model file, click the transform button, then select the diagram you wish to analyze and hit the "Check Consistency" button. Future versions of the tool will produce more transformation products illustrating a composition of related diagrams of the source specification (e.g. an interaction between two state machines), but configuration will remain minimal as the tool will auto select incrementally more complex scenarios (e.g. isolated diagrams with uncontrolled environment, if validated composition of diagrams etc…). Additional consistency rules will also be defined introducing new kinds of errors and warnings, but this does not require user intervention.

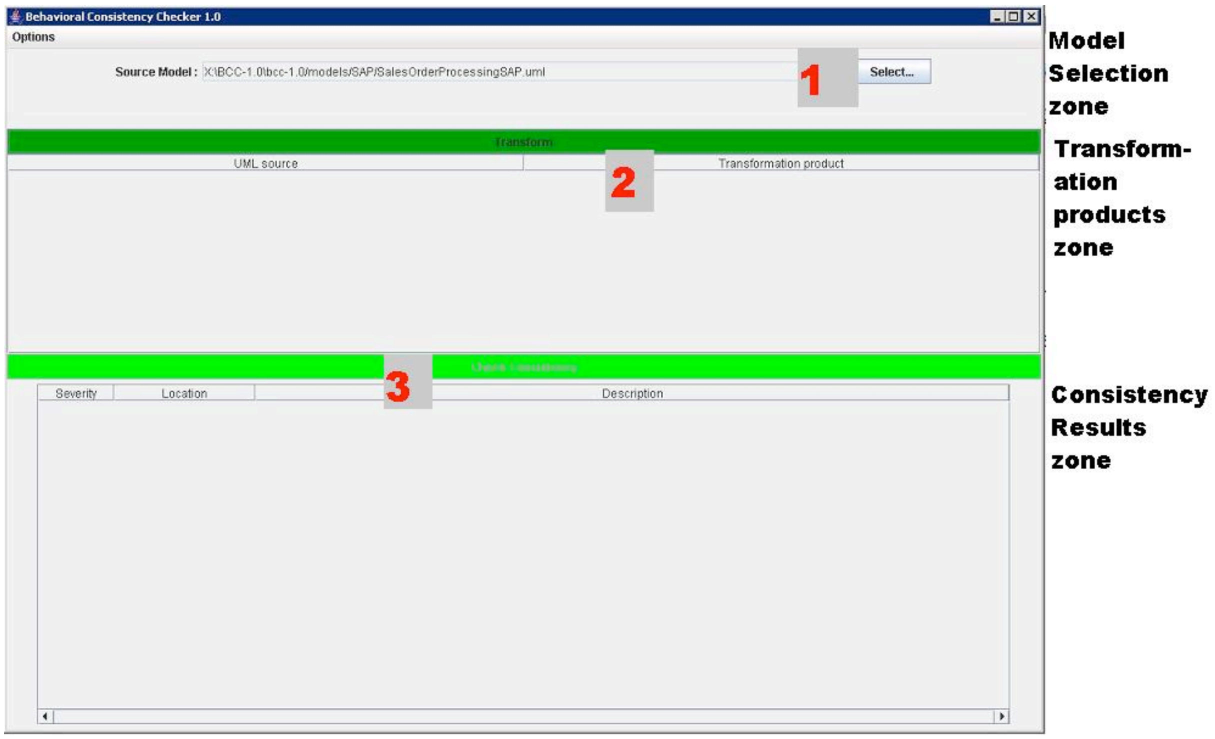
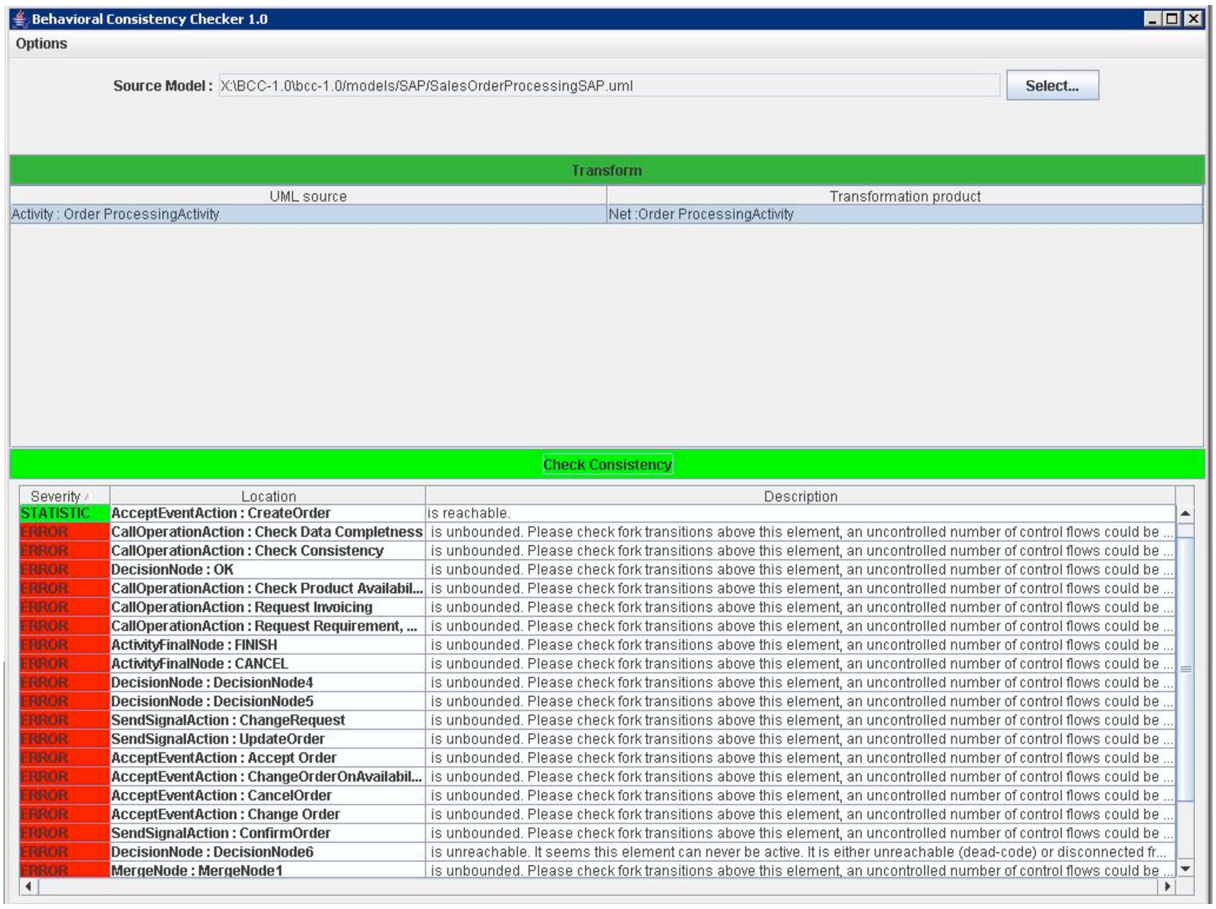

Source Model : X:\BCC-1.0\bcc-1.0/models/SAP/SalesOrderProcessingSAP.uml

Figure 1 BCC Graphical User Interface

The steps to run the tool are:

1. Select the model.
2. Transform the model to obtain a Petri net for each diagram.
3. Run the analysis.
4. If you have errors try to correct them and then iterate.

The error description are hopefully self-explanatory, but a full index of potentially raised errors and some tips on how to correct them will be elaborated and available with the full user documentation of the next prototype release.



Source Model : X:\BCC-1.0\bcc-1.0/models/SAP/SalesOrderProcessingSAP.uml

UML source	Transformation product
Activity : Order ProcessingActivity	Net :Order ProcessingActivity

Severity	Location	Description
STATISTIC	AcceptEventAction : CreateOrder	is reachable.
ERROR	CallOperationAction : Check Data Completness	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	CallOperationAction : Check Consistency	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	DecisionNode : OK	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	CallOperationAction : Check Product Availabil...	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	CallOperationAction : Request Invoicing	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	CallOperationAction : Request Requirement, ...	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	ActivityFinalNode : FINISH	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	ActivityFinalNode : CANCEL	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	DecisionNode : DecisionNode4	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	DecisionNode : DecisionNode5	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	SendSignalAction : ChangeRequest	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	SendSignalAction : UpdateOrder	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	AcceptEventAction : Accept Order	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	AcceptEventAction : ChangeOrderOnAvailabil...	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	AcceptEventAction : CancelOrder	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	AcceptEventAction : Change Order	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	SendSignalAction : ConfirmOrder	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...
ERROR	DecisionNode : DecisionNode6	is unreachable. It seems this element can never be active. It is either unreachable (dead-code) or disconnected fr...
ERROR	MergeNode : MergeNode1	is unbounded. Please check fork transitions above this element, an uncontrolled number of control flows could be ...

Figure 2 Error display. You can sort the results by clicking the column header, here we sort by severity.