# Petri Nets Verification

## Prerequisites

- BPEL2PNML version 1.1: A tool for translating BPEL to Petri nets represented using the Petri Net Markup Language (PNML). (http://yawlfoundation.org/babeltools/),
- BPEL2OWFN version 2.0.3: BPEL2oWFN translates BPEL into an oWFN (open Workflow Net). This oWFN can be used to: check for deadlocks or any other Petri net property, or check any temporal logic formula with a variety of model checking tools. (https://www.gnu.org/software/bpel2owfn/),
- LoLA version 2.0: A Low Level Petri Net Analyzer.\\ (http://service-technology.org/lola/),
- WofBPEL version 1.0.2: A tool for formal analysis of BPEL processes. This tool takes as input Petri net and checks for unreachable BPEL activities and for concurrent activities that may enter in competition for the same inbound message.\\ (http://yawlfoundation.org/babeltools/),
- Graphviz version 2.38: Graph visualization tool (http://www.graphviz.org/).

## Verification steps

BPEL → BPEL2PNML → Petri Net in PNML format → WofBPEL → verification results

BPEL → BPEL2OWFN → Petri Net in oWFN format → LoLA + CTL formula → verification results

BPEL2OWFN ihas been used in two different modes (mode=choreography and mode=petrinet). Output files have postfix 'ch' or 'pn' to indicate used mode.

## Files (other samples also added in MODELCHECKING\petrinet)

| | |
|---|---|
| HandleCaseProcess.bat | Command file to run all transformations |
| HandleCaseProcess.bpel | Source BPEL (modified to be able to use the tooling) |
| HandleCaseProcess.bpel.org | Original BPEL file |
| HandleCaseProcess.wofbpel | Result WofBPEL verification |
| HandleCaseProcessBPEL2OWFN_CH.info | Additional information about the transformation process |
| HandleCaseProcessBPEL2OWFN_PN.info | Additional information about the transformation process |
| HandleCaseProcessBPEL2PNML.info | Additional information about the transformation process |
| HandleCaseProcessBPEL2PNML.xml | Result of BPEL to PNML transformation |
| HandleCaseProcessverify.bat | Run LoLA + CTL formula verification |
| HandleCaseProcessverify.output | LoLA + CTL formula verification results |
| HandleCaseProcess_ch.bmp | Visualization of generated Petri Net created with Graphviz |
| HandleCaseProcess_ch.dot | Graphical format generated with BPEL2OWFN. Graphviz input |
| HandleCaseProcess_ch.info | Additional information about generated Petri Net |
| HandleCaseProcess_ch.lola | Result of BPEL to Petri Net transformation |
| HandleCaseProcess_ch.task | Additional information about generated Petri Net |
| HandleCaseProcess_ch.xml | Additional information about generated Petri Net |
| HandleCaseProcess_pn.info | Additional information about generated Petri Net |
| HandleCaseProcess_pn.lola | Result of BPEL to Petri Net transformation |
| HandleCaseProcess_pn.task | Additional information about generated Petri Net |
| HandleCaseProcess_pn.xml | Additional information about generated Petri Net |

```
lola:   CTL formula contains 2 significant temporal operators and needs 9 bytes of payload
lola: RESULT
lola:   result: yes
lola:   The net satisfies the given formula.
lola:   341 markings, 1032 edges

c:\ws\tst>lola --formula="(AG EF (p975 > 0))" HandleCaseProcess_pn.lola
lola: NET
lola:   reading net from HandleCaseProcess_pn.lola
lola:   finished parsing
lola:   closed net file HandleCaseProcess_pn.lola
lola:   2473/65536 symbol table entries, 6 collisions
lola:   preprocessing...
lola:   finding significant places
lola:   1038 places, 1435 transitions, 899 significant places
lola:   computing forward-conflicting sets
lola:   computing back-conflicting sets
lola:   1494 transition conflict sets
lola: TASK
lola:   read: AG (EF (p975 > 0))
lola:   formula length: 18
lola:   checking liveness
lola:   liveness not yet implemented, converting to CTL...
lola:   processed formula: !(E(TRUE U !(E(TRUE U p975 > 0))))
lola:   processed formula length: 34
lola:   3 rewrites
lola:   formula mentions 1 of 1038 places; total mentions: 1
lola: STORE
lola:   using a bit-perfect encoder (--encoder=bit)
lola:   using 3596 bytes per marking, with 0 unused bits
lola:   using a prefix tree store (--store=prefix)
lola: RUNNING
lola:   CTL formula contains 2 significant temporal operators and needs 9 bytes of payload
lola: RESULT
lola:   result: yes
lola:   The net satisfies the given formula.
lola:   341 markings, 1032 edges

c:\ws\tst>lola --formula="(p10 > 0) -> (ALLPATH ALWAYS NOT DEADLOCK)" HandleCaseProcess_pn.lola -s -p
```