BPEL2oWFN

Translating BPEL Processes into Open Workflow Nets
BPEL2oWFN Version 1.2, 6 April 2006
Installation Manual

Niels Lohmann

About this document:

This manual describes the installation of BPEL2oWFN, version 1.2, a tool translating a web service described in BPEL into open workflow net (oWFN), last updated 6 April 2006. Once BPEL2oWFN is installed please read the User's Manual which is part of the distribution or can be downloaded from the website of BPEL2oWFN (http://www.informatik.hu-berlin.de/top/tools4bpel/bpel2owfn).

BPEL2oWFN is part of the Tools4BPEL project funded by the Bundesministerium für Bildung und Forschung. See http://www.informatik.hu-berlin.de/top/tools4bpel for details.

Table of Contents

1 Contents of this Distribution

1.1 Files

`AUTHORS` A list of all authors of BPEL2oWFN as well as acknowledgements.

`ChangeLog`
The change log listing all changes made during the distribution versions of BPEL2oWFN.

`COPYING` The GNU General Public License (GPL) under which BPEL2oWFN is distributed. The GPL is also a chapter of the User's Manual.

`INSTALL` Setup and Installation instructions and all information needed to bootstrap BPEL2oWFN.

`NEWS` News concerning BPEL2oWFN or the Tools4BPEL project.

`README` A file describing the contents of the distribution.

Moreover, the directory contains several files controlling the setup, build and installation process. These files were generated by the GNU Autotools. For more information on the installation process see Chapter 2 [Setup and Installation], page 2.

1.2 Directories

`src` The sources of BPEL2oWFN. Each file contains a short documentation of what it does as well as Doxygen-styled comments (see `http://www.stack.nl/~dimitri/doxygen` for details).

`doc` The documentation of BPEL2oWFN consisting of a manual and this installation manual. The documentation is in Texinfo format and can be converted into many file formats using '`make` *`format`*' where '*`format`*' is a file format out of '`dvi`', '`html`', '`pdf`' or '`ps`'.

`tests` Several test processes to self-test BPEL2oWFN. The processes '`loan.bpel`', '`purchase.bpel`' and '`shipping.bpel`' were taken from the BPEL4WS 1.1 specification.

2 Setup and Installation

The distribution environment of BPEL2oWFN was created using the GNU Autotools. If you are familiar with the setup and installation procedure of any GNU tool you may skip this section.

2.1 Setup

To setup the installation environment and to compile BPEL2oWFN simply follow these steps:

1. Unpack the source tarball by entering[1]
   ```
   gunzip bpel2owfn-1.2.tar.gz
   tar xf bpel2owfn-1.2.tar
   ```
2. Enter that directory:
   ```
   cd bpel2owfn-1.2
   ```
3. Run the configure shell script which attempts to guess correct values for various system-dependent variables used during compilation. It uses those values to create a '`Makefile`' in each directory of the package. Entering
   ```
   ./configure
   ```
   should lead to an output like:

```
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for g++... g++
checking for C++ compiler default output file name... a.exe
checking whether the C++ compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables... .exe
checking for suffix of object files... o
checking whether we are using the GNU C++ compiler... yes
checking whether g++ accepts -g... yes
checking for style of include used by make... GNU
checking dependency style of g++... gcc3
checking for gcc... gcc
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ANSI C... none needed
checking dependency style of gcc... gcc3
checking for a BSD-compatible install... /usr/bin/install -c
checking for flex... flex
checking for yywrap in -lfl... yes
checking lex output file root... lex.yy
checking whether yytext is a pointer... no
checking for bison... bison
checking for kc++... kc++
configure: creating ./config.status
config.status: creating Makefile
config.status: creating doc/Makefile
config.status: creating src/Makefile
config.status: creating tests/Makefile
config.status: creating src/bpel2owfn.h
config.status: executing depfiles commands
```

   In some cases the configure script complains (i.e. warns) about tools which could not be found in your system path. In this case please check Chapter 3 [Needed Programs], page 5. BPEL2oWFN can be compiled without these tools. However, you cannot make changes to

[1] Cygwin users should open a `tcsh`-shell before.

large parts of the programm as you will not be able to generate source files without the tools.

4. After all Makefiles are created, run

   ```
   make
   ```

   to compile the executable binary as well as a documentation file.

5. Alternatively, you can run

   ```
   make check
   ```

   to self-test the created binary. After many lines of output `make check` should announce

   ```
   ==================
   All 4 tests passed
   ==================
   ```

   or, if anything went wrong, prints error message and requests to report the encountered bug to nlohmann@informatik.hu-berlin.de.

2.2 Installation

After successful compilation and self tests an executable file '`bpel2owfn`' (or '`bpel2owfn.exe`' on systems running Microsoft Windows) is created in the directory '`bpel2owfn-1.2/src`' of the distribution. This file has no dependencies to other files of the distribution and can be copied to any directory.

Alternatively you can install BPEL2oWFN properly by executing[2]

```
make install
```

to copy the binary to the directory '`/usr/local/bin`' and to make the documentation accessible (i.e. copy it to '`/usr/local/info`') for

```
info bpel2owfn
```

You can undo the installation by entering

```
make uninstall
```

The setup and installation procedure can be customized by running `./configure` with several command-line options. Type

```
./configure --help
```

for more information.

2.3 Problems during Setup or Installation

Any warning during an in other respects successful compilation can be ignored — these warnings mostly occur in generated files from Flex, Bison or Kimwitu++ and are outside of our scope.

The compilation of BPEL2oWFN was tested on several operating systems such as Microsoft Windows (running Cygwin), GNU/Linux, Apple Mac OS X and Sun Solaris. Anyway, if the compilation fails please make sure you use the most recent (stable) versions of the GNU Compiler Collection, Flex, Bison and Kimwitu++. If this does not help please send us an electronic mail to nlohmann@informatik.hu-berlin.de. Include the version number of your distribution (version 1.2 of 6 April 2006). Also include in your message the output that the make process produced. We will try to answer your mail within a week.

If you are not interested in compiling BPEL2oWFN yourself you can alternatively download executable binaries for several operating systems on the website of BPEL2oWFN (http://www.informatik.hu-berlin.de/top/tools4bpel/bpel2owfn).

[2] Usually, '`make install`' and '`make uninstall`' can only be executed by the root user.

2.4 Makefile Targets

The generated Makefiles serve different targets:

`all` Compile BPEL2oWFN and create an executable binary '`bpel2owfn`' (or '`bpel2owfn.exe`' on systems running Microsoft Windows) in the directory '`bpel2owfn-1.2/src`'.

`dvi`
`html`
`pdf`
`ps` Generate documentation files in the given format which can be DVI, HTML, PDF or PostScript. The documentation is generated in the directory '`bpel2owfn-1.2/doc`' and is named '`bpel2owfn.`*`format`*' (User's Manual).

`clean` Clean all temporary files created during the compilation. Executables as well as generated documentation are preserved.

`distclean`
Clean also source files generated by the compiler generator tools (Flex, Bison and Kimwitu++). Executables as well as created documentation files are also removed. To re-compile you have to run '`./configure`' again. Please note that afer running '`make distclean`' you need the tools described in Chapter 3 [Needed Programs], page 5 to compile.

`check` Perform a number of self-tests: the directory '`tests`' contains several BPEL processes which are read and compiled.

`pure-windows-executable`
Create an executable binary which is independent of the Cygwin libraries. This file can be executed on any system running Microsoft Windows, independently of a Cygwin environment.

3 Needed Programs

3.1 Required Programs

To compile BPEL2oWFN you do not need any specific program other than the GNU Compiler Collection. If you however want to change the program (e.g. add new Petri net patterns) you need the following programs to generate source code. These programs are (as BPEL2oWFN itself) free software distributed under the GNU General Public License.

Flex, A fast scanner generator
: Flex is needed to generate the lexer of BPEL2oWFN. Flex was written by Vern Paxon. Copyright © 1990 The Regents of the University of California. All rights reserved. Flex is available under http://www.gnu.org/software/flex/flex.html.

Bison, The YACC-compatible Parser Generator
: Bison is needed to generate the parser of BPEL2oWFN. Bison was written by Robert Corbett und Richard Stallman. Copyright © 1992, 1998, 1999, 2003, 2004, 2005 Free Software Foundation, Inc. Bison is available under http://www.gnu.org/software/bison/bison.html.

Kimwitu++, The Term Processor
: Kimwitu++ is needed to generate all files managing the AST (abstract syntax tree). Kimwitu++ was written by Toby Neumann and Michael Piefel. Copyright © 1988-1997 Axel Belinfante, University of Twente. Copyright © 1997-2000 Michael Piefel, Humboldt-University Berlin. Kimwitu++ is available under http://site.informatik.hu-berlin.de/kimwitu++.

3.2 Optional Programs

BPEL2oWFN is a compiler and thus creating output files which can be read by several other programs. These programs are not needed to compile BPEL2oWFN yet without these programs BPEL2oWFN makes not much sense.

LoLA, a Low Level Petri net Analyzer
: LoLA is an explicit model checker implementing a broad variety of reduction techniques such as partial order reduction, symmetries or the sweep line method. LoLA was written by Karsten Schmidt and is available under http://www.informatik.hu-berlin.de/~kschmidt/lola.html.

The Model-Checking Kit
: The Model-Checking Kit is a collection of programs which allow to model a finite-state system using a variety of modelling languages, and verify it using a variety of checkers, including deadlock-checkers, reachability-checkers, and model-checkers for the temporal logics CTL and LTL. The Model-Checking Kit is maintained by Javier Esparza, Claus Schrter and Stefan Schwoon at the University of Stuttgart and is available under http://www.fmi.uni-stuttgart.de/szs/tools/mckit.

Graphviz, the Graph Visualization Software
: Graphviz is a graph visualization software with several graph layout programs including dot. Graphviz is maintained by AT&T Labs and is available under http://www.graphviz.org.

3.3 Reference System

As of April 2006, we use the following system developing and building BPEL2oWFN:

- Intel Pentium M processor, 1.60 GHz
- 1 GB RAM
- Microsoft Windows XP Home Edition Service Pack 2
- Cygwin with cygwin.dll version 1.5.19-4
- GNU Compiler Collection 3.4.4
- Flex version 2.5.4
- GNU Bison version 2.1
- Kimwitu++ version 2.3.8

The version numbers are not to be understood as system requirements, but as a help if you fail to compile the sources.

4 Change Log

Most recent version at the end of this section.

Version 1.0 (16 January 2006)

- input:
  - BPEL process compliant to the specificiation BPEL4WS version 1.1, May 2003.
- output:
  - LoLA Petri net
  - open workflow net
  - Dot representation
- implemented patterns:
  - Christian Stahl. *A Petri Net Semantics for BPEL*. Informatik-Berichte 188, Humboldt-Universität zu Berlin, July 2005.

Version 1.1 (24 February 2006)

- new output formats:
  - APNN (Abstract Petri Net Notation) Petri net
  - PNML (Petri Net Markup Language) Petri net
  - low-level PEP (Programming Environment based on Petri Nets) Petri net
- overworked patterns:
  - the Stahl-semantics (see version 1.0) has been enhanced and can be parametrized:
    - original semantics
    - models without BPEL's standard faults
    - acyclic models

    For more information please refer to the User's Manual.
- static analysis:
  - To reduce the size of the Petri net model static analysis is used to only generate nodes for the DPE (dead path elimination) when necessary.
- miscellaneous:
  - The compiler generator tools Flex, Bison and Kimwitu++ are not any more necessary to compile BPEL2oWFN: the generated sources are now part of the distrubution.
  - The oWFN format was overworked and allows final markings.
  - GNU getopt allows more flexible command-line parameter processing allowing to create several output file formats in a single run.
  - Debug output can be written into a log file.
  - The classes organizing the Petri nets are overworked for performance and extensibility.
- fixed bugs:
  - The generation of all possible types of the `<invoke>` activity (synchronous, asynchronous, with and without implicit scope) is overworked.

Version 1.2 (6 April 2006)

- overworked patterns:
  - more parameterized versions of the Stahl-semantics (see version 1.1):
    - acyclic models (also acyclic event handlers)

    - models without variables
    - models without standard faults occuring while handling a fault

  For more information please refer to the User's Manual.
- static analysis:
  - A control flow graph can be built and used to detect the use of uninitialized variables.
- miscellaneous:
  - The command-line parameter '`-p finishloop`' is renamed to '`-p finalloop`'.
  - Structural reduction rules remove all structural dead nodes of the generated Petri net model.
- fixed bugs:
  - Pattern of asynchronous `<invoke>` activity does no longer receive messages.
  - Structural reduction rules no longer change the semantics of the `<switch>` activity.
  - Structural reduction rules no longer remove the loop transition for deadlock search.

5 Authors and Acknowledgements

5.1 Authors

BPEL2oWFN was written by:

- Niels Lohmann (Humboldt-Universität zu Berlin)
- Christian Gierds (Humboldt-Universität zu Berlin)
- Dennis Reinert (Humboldt-Universität zu Berlin)

The Petri net patterns were created by:

- Christian Stahl (Humboldt-Universität zu Berlin)

5.2 Acknowledgements

We would like to thank:

- Simon Moser (IBM) for valuable advice concerning BPEL
- Michael Piefel (Humboldt-Universität zu Berlin) for help with Kimwitu++
- Oliver Kopp (Universität Stuttgart) for challenging BPEL processes

BPEL2oWFN is part of the Tools4BPEL project funded by the German Bundesministerium für Bildung und Forschung.

5.3 Contact Person

Niels Lohmann

Humboldt-Universität zu Berlin
Institut für Informatik
Unter den Linden 6
10099 Berlin, Germany

Homepage	http://www.informatik.hu-berlin.de/top/mitarbeiter/lohmann
E-mail	nlohmann@informatik.hu-berlin.de
Phone	(+49) (30) 2093-3070
Fax	(+49) (30) 2093-3067