Migrating OMNeT++ simulations from version 3.x to 4.0

Overview

This migration document provides information that needs to be considered in order to make an OMNeT++ 3.x simulation work under OMNeT++ 4.0. There are several other changes in various parts of the system which is not required during migration due to either being backward compatible or optional. For example, the ini file might be specified without the '-f' option directly as the first parameter on the simulation command line.

The reader should be already familiar with the OMNeT++ 3.x and 4.0 specifications before doing the migration.

The migration process

Since migrating a complex simulation model might be a non trivial and error prone task to do it is best done in two separate phases.

1. First migrate all files using as few of the new features as possible and try to reproduce the same simulation results either exactly or statistically. For example, stick to use double simulation time so that the new 64 bits exact simulation time representation does not affect the simulation model.

2. When the simulation results are satisfying switch to the new 64 bits exact simulation time representation and start using other new features of OMNeT++ 4.0.

Note that using double simulation time requires recompilation of OMNeT++ 4.0 because the prebuilt binaries are compiled with the 64 bits exact representation. Specifying the compiler switch -DUSE_DOUBLE_SIMTIME in CFLAGS and recompiling OMNeT++ provides the required binaries.

What needs to be migrated?

Migration is done by modifying the contents of various files in the simulation model. Some of them

1. environment variable scripts

2. ned files

3. msg files

4. C++ code and header files

5. ini files

6. makefiles
7. simulation batch execution scripts
8. .tkenvrc files
9. result post processing scripts
10. various file generator scripts

What doesn't need to be migrated?

What doesn't need to be migrated?

1. vector files?

QUESTION: problems with indexer, generates unreasonably big index files?

1. scalar files

Migration tools

1. tools in the migrate/ subdirectory.
2. opp_makemake
3. nedtool
4. scavetool
5. initool???
6. eclipse

Migration rules

Environment variables

Environment variables

1. The environment variable OMNETPP_BITMAP_PATH has been renamed to OMNETPP_IMAGE_PATH. The system will check this at runtime and print a warning if the old variable is still present.

ned files

1. The keywords endsimple, endmodule, endnetwork, endchannel are obsolete and the ned syntax has been changed to use a pair of curly braces instead.
2. The numeric parameter type is obsolete and must be replaced with int or double depending on the parameter usage.

3. The keyword const is obsolete and a new keyword volatile has been introduced. In 3.x an unqualified parameter was volatile while it is const in 4.0.

4. The display string has been turned into a property with the syntax @display(…).

5. Introduced a new variable property @unit(…) to specify physical units. For parameters where unit is specified all the values in ini files and ned files must also specify convertible units otherwise an error will be signaled.

6. The parameter prompt string also became a property as @prompt(…).

7. Conditional parameters are obsolete.

8. The gatesizes section in compound modules has been renamed to gates.

9. !! Parameter syntax swapped

10. The submodules section also uses a pair of curly braces for each submodule it defines.

11. !! connection syntax uses curly braces

12. !! for syntax changed

13. Ancestor parameters are obsolete.

14. The ref keyword is obsolete because parameters are now always passed by reference.

15. There is no more implicit conversion between bool and long/double

16. The import declarations now refer to fully qualified package or type names instead of files.

msg files

1. The field property syntax has been changed to be same as for ned files.

cc/h files

1. The 4.0 version introduces a new C++ class called SimTime for 64 bits exact simulation time representation. This is the default setting which might break existing simulation models. It is strongly recommended to use double simulation time representation during the migration and validation process and switch to the new class in a separate step. See the section The migration process for how to do this.

2. To simultaneously support double and SimTime simulation time representations one must use macros to convert simtime_t from/to double and const char *. The macros are SIMTIME_STR(t), SIMTIME_DBL(t), STR_SIMTIME(s),

SIMTIME_TTOA(buf, t). The constant MAXTIME is also defined correctly for both types.

3. The new SimTime class does not provide implicit conversion to double because it would silently lose precision (not even speaking of ambiguities). Unfortunately many simulation models use double variable types directly instead of simtime_t which will produce compilation errors at various places. Those definitions must be changed to specify simtime_t instead.

4. omnetpp/include/ChangeLog: head/tail to back/front

ini files

1. The sections [Cmdenv] and [Tkenv] are obsolete and the entries from these sections have been prefixed with cmdenv- and tkenv- respectively and can be written in any section. Practically most of the time they should be copied under the [General] section.

2. The sections [Run 1], [Run 2], … are obsolete and should be renamed to [Config …], [Config …], … Note that run numbers no longer refer to configuration sections but iteration numbers.

3. The entry cmdenv-express-mode (which was express-mode under [Cmdenv]) defaults to true instead of false.

4. Most of the entries in the section [Tkenv] are obsolete except the following ones which are still supported but prefixed as: tkenv-default-run, tkenv-image-path, tkenv-plugin-path.

5. The entry tkenv-default-run (which was default-run under [Tkenv]) was referring to a section with multiple entries but now it refers to an iteration number. This is an incompatible change and thus requires careful manual investigation.

6. There is a new entry cmdenv-interactive defaulting to false which causes cmdenv to never read stdin and abort on missing configuration entries. In 3.x the default behavior was to read a value from stdin.

7. The entry preload-ned-files is obsolete because in 4.0 ned files are loaded based on their packages and the entry ned-path which specifies a list of directories separated by semicolons. In a single directory simulation model the default value '.' for ned-path should be sufficient.

8. In 3.0 the entry network was referring to a ned type loaded from one of the files specified in the entry preload-ned-files. In 4.0 it specifies a qualified name referring to a ned type which must be available under the directories specified in the entry ned-path. In a single directory simulation model the network entry should work unmodified.

9. The entries **.use-default, **.interval has been renamed to **.apply-defaults, **.record-interval respectively.

Makefiles

The makefile generation and the make process has been rewritten. It is practically impossible to convert a modified 3.x Makefile to 4.0 but can be replaced by a newly generated one.

simulation launch files

simulation launch files

1. The run number command line parameter (specified with the command line switch –r) does not refer to an ini file configuration section any more. It specifies the TODO

.tkenvrc files

1. Some obsolete entries will be deleted

Automatic migration

The automatic command line migration tools are available under the migrate directory in the OMNeT++ 4.0 installation.

ned files

The tool migratened recursively migrates all ned files under the current directory by doing the following:

1. Converts all simple, module, network, channel type declarations to use the new curly brace format.

2. Converts all submodule declarations to use the new curly brace format.

3. Removes const qualifiers and adds volatile qualifier to non const parameter definitions.

Additional optional manual processing suggested:

1. Due to making the const qualifier obsolete and people being lazy to write out const explicitly some superfluous volatile qualifiers might pop up for parameters. It is safe to delete the volatile qualifier from parameters which are expected to be constant over the simulation. For example, if the parameter is only used by calling the C++ function par once inside initialize.

2. For safety reasons the automatic migration converts numeric parameter types to double. The parameters must be manually checked if the type int would be sufficient and change accordingly.

3. Check expressions and literal values for parameters and add units where required. For example, delay and datarate for channels.

4. Change import declarations to refer to fully qualified packages or type names. The simplest way is to use the organize imports feature from the OMNeT++ IDE by pressing Ctrl+Shift+O in the ned file text editor.

msg files

The tool migratemsg recursively migrates all msg files under the current directory by doing the following:

1. Converts all properties to the new format.

Additional manual migration is not required.

ini files

The tool migrateini recursively migrates all ini files under the current directory by doing the following:

1. Copies the entries from the sections [Parameters], [Cmdenv], [Tkenv], [OutVectors], [Partitioning] to the section [General].

2. Merges the entries from multiple occurrences of the section [General] into one.

3. Prefixes the entries in the sections [Cmdenv], [Tkenv] with cmdenv- and tkenv- respectively (unless the entry already begins with that).

4. Renames the sections [Run 1], [Run 2]... to [Config config1], [Config config2]...

5. Renames the output vector configuration entries **.interval to **.record-interval.

6. Renames the entries **.use-default to **.apply-default.

Makefiles

The old generated makefiles must be replaced with new makefiles generated by opp_makemake.

Manual migration

Despite the automatic migration support there are various cases where it is insufficient and manual migration must be done. This requires deep understanding of the simulation model, the OMNeT++ 3.x specification and the OMNeT++ 4.0 specification.

The OMNeT++ specifications are available at:

http://www.omnetpp.org/doc/manual/usman.html [http://www.omnetpp.org/doc/manual/usman.html]