Tips & Tricks for OMNeT++

Rudolf Hornig

OMNeT++ Workshop
March 21, 2010
Barcelona, Spain

Using Valgrind in IDE

Some memory corruption issues are very hard to track down using a simple debugger. Valgrind on Linux can help you to nail down these bugs.

In OMNeT++ 4.2b1 you can start the simulation directly with Valgrind. After running the simulation you will get a detailed log about potential issues.

Demo: Finding a mysterious bug in TictocValgrind project. After running the simulation for a while it crashes or a spurious error message appears:

"The dup() method, declared in class cObject, is not redefined in class cObject"

The simulation command line

How to start a simulation from the command line?

The IDE launcher helps you start your simulations easily by building the command line and the environment for your program automatically. Sometimes it would be great to start the simulation from outside the IDE. How do you know what command line should be used?

Tip: Run your, simulation from the IDE, go to 'Debug view', right click on your program and select 'Properties'. The IDE will show you the currently used command line, working directory and path variable.

Measuring channel throughput

Sometimes you need to measure the throughput and other statistics of a link (Datarate channel). Previously you had to insert a ThruputMeter module in the path, or let a ChannelInstaller replace channel objects with ThruputMetering channel.

Starting with OMNeT++ 4.1, this is no longer needed. The built-in DatarateChannel provides throughput statistics although they are disabled by default.

To enable throughput statistics in your result files, put in your INI file:

```
**.channel.throughput.result-recording-modes = all
```

Busy, utilization, packets, packetBytes, packetsDiscarded are also available. Tip: try 'opp_run -h neddecls'

Unreproducible results

The simulation must generate exactly the same results independent of platform, architecture or the currently used runtime environment. If you do not get consistent results for the same run, you should check for:

- Uninitialized variables (use Valgrind)
- Undefined iteration order on maps caused by using pointers as keys.
- Overflow of architecture specific values (int, long etc.)
- Unintended use of ev.isGui() which modifies the behavior of the simulation depending on the currently used runtime environment.

Unreproducible results 2

These errors are very hard to catch because they do not cause direct crashes and they may surface only occasionally on specific platforms or only after some time of running.

A debugger can be a great help, but first we have to find the event where the two runs start to diverge. Event log recording can help here. Record both runs by starting the simulation with: `--record-eventlog=true`

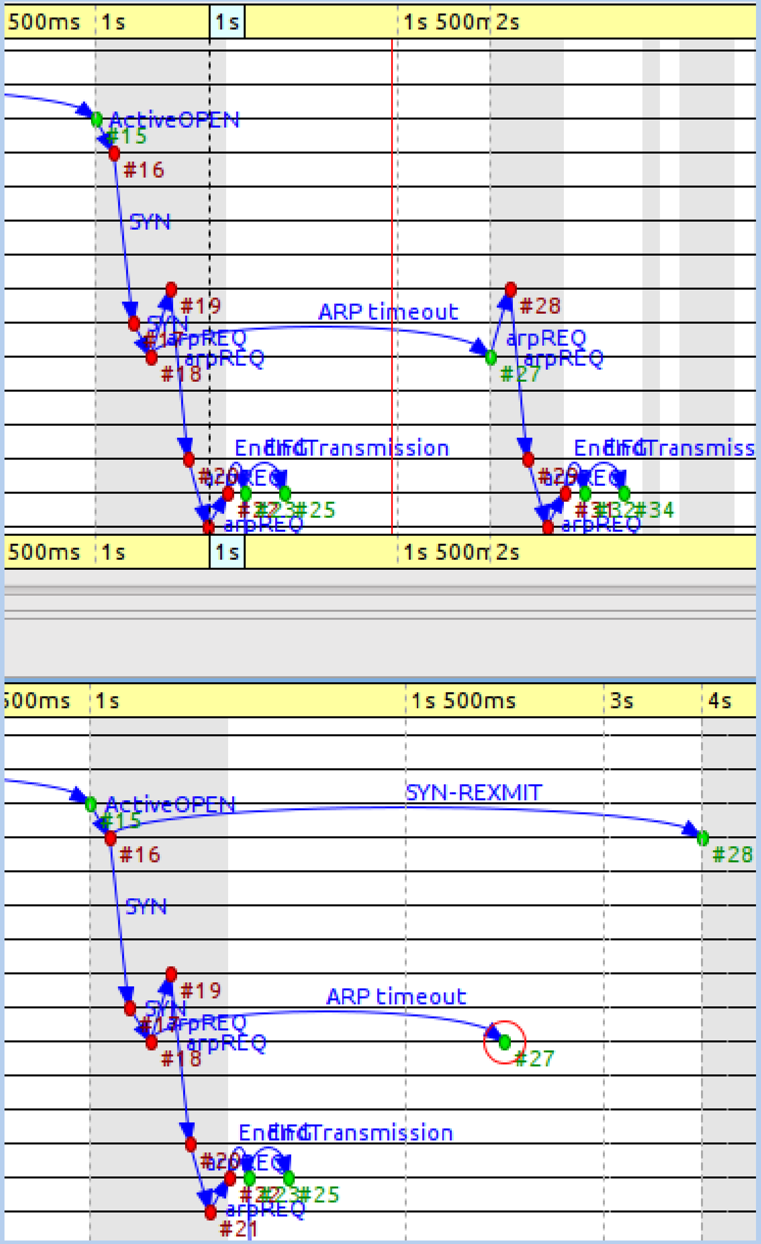
This will generate two .elog files that can be compared. Of course a visual diff tool is useful here, but you can also open both log files in the sequence chart editor and place them side by side and look around where the difference is.

Unreproducible results 3

We have to look for the first event where the two simulation starts to differ. Let's take a look at the provided example where the ARP example is behaving differently in Tkenv and Cmdenv.

Notice that the first event occurring at different time is #28. It is directly caused by #27 (an ARP timeout)

Debug that code and see what happens in those events.

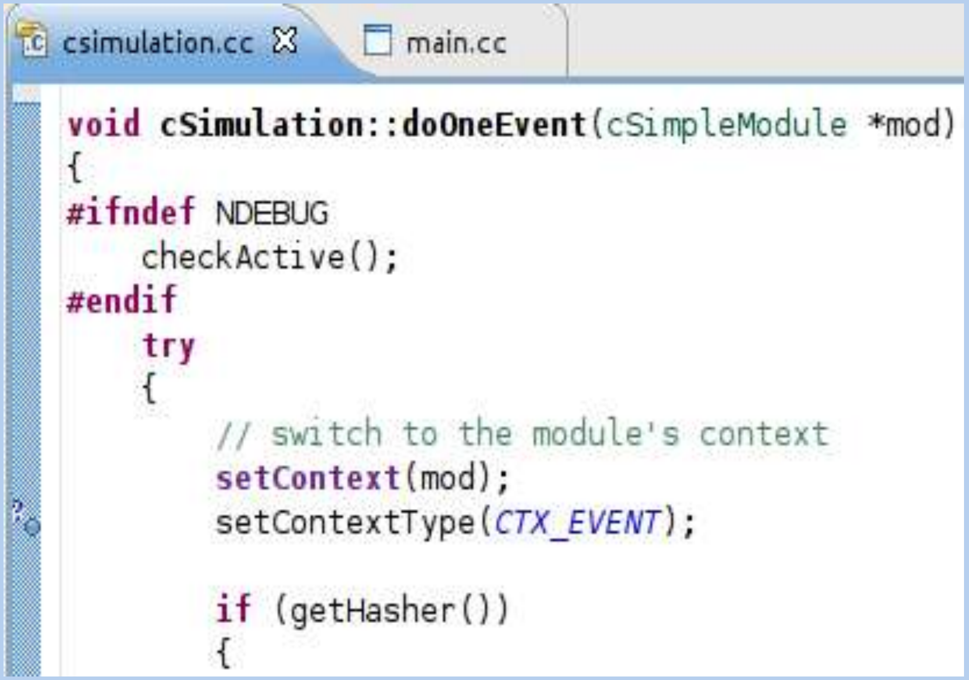


Unreproducible results 4

Let's start the debugger and place a breakpoint in cSimulation::doOneEvent(...)

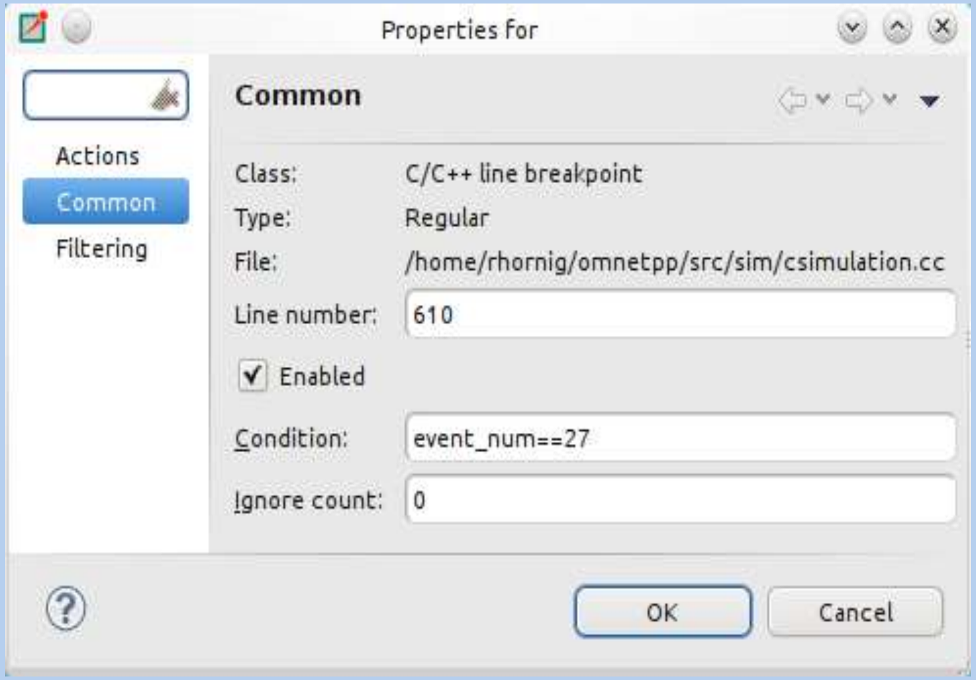
This is the place where each event is processed and passed to the model.

We want to check event #27, so we have to add the following condition to the breakpoint: event_num==27



```
void cSimulation::doOneEvent(cSimpleModule *mod)
{
#ifndef NDEBUG
    checkActive();
#endif
    try
    {
        // switch to the module's context
        setContext(mod);
        setContextType(CTX_EVENT);

        if (getHasher())
        {
```

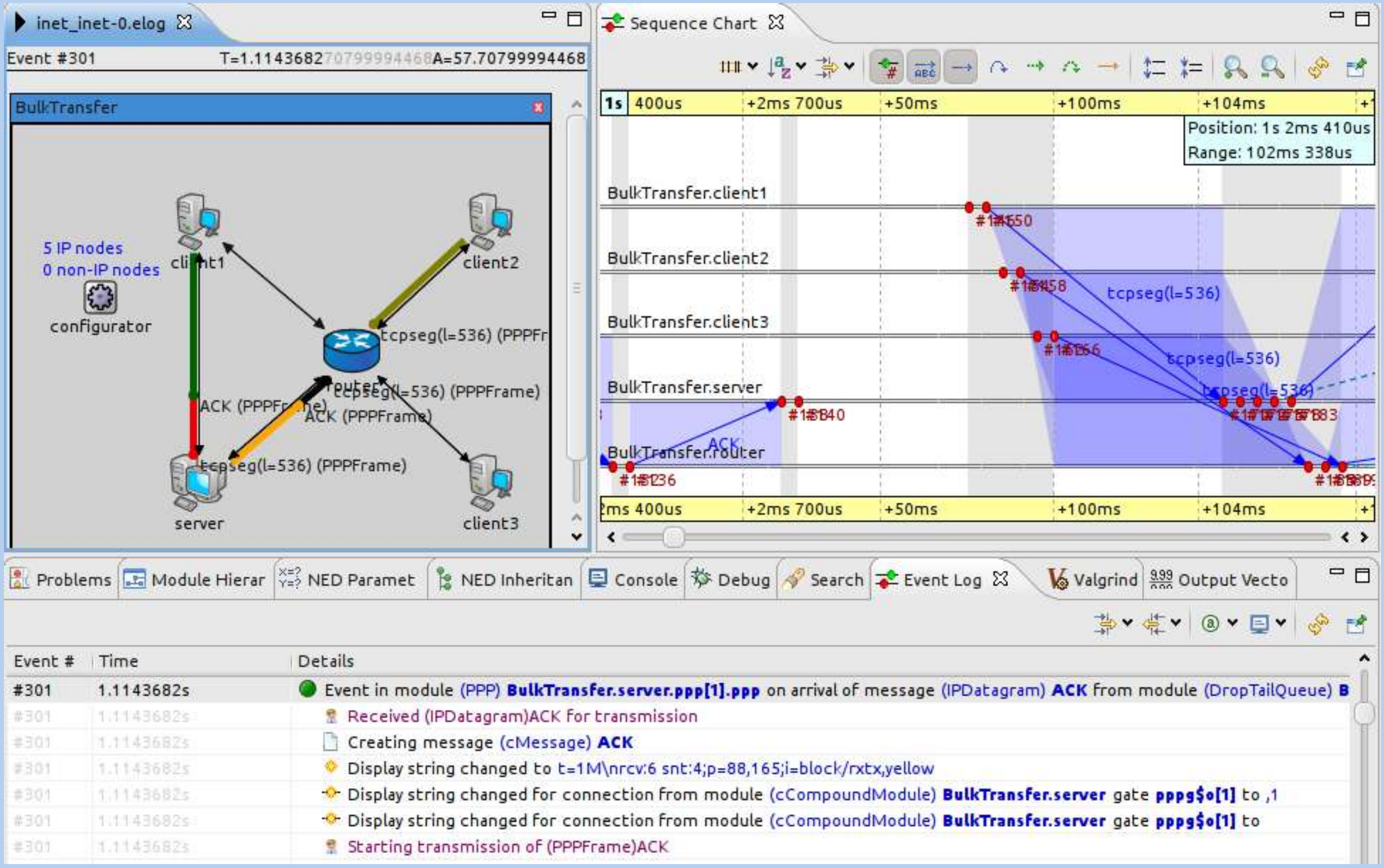


Common

Class:	C/C++ line breakpoint
Type:	Regular
File:	/home/rhornig/omnetpp/src/sim/csimulation.cc
Line number:	610
[✓] Enabled	
Condition:	event_num==27
Ignore count:	0

Using the Animation Player

OMNeT++ 4.2b1 now supports the animation playback of .elog files. It is possible to display the same log file in Animator, Sequence Chart and the Event Log View at the same time:

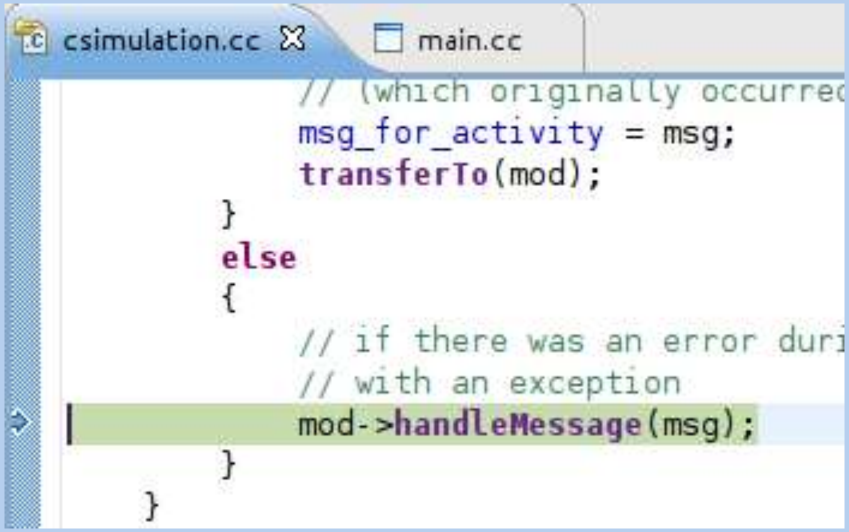


```
Event #  Time         Details
#301     1.1143682s   Event in module (PPP) BulkTransfer.server.ppp[1].ppp on arrival of message (IPDatagram) ACK from module (DropTailQueue) B
#301     1.1143682s     Received (IPDatagram)ACK for transmission
#301     1.1143682s     Creating message (cMessage) ACK
#301     1.1143682s     Display string changed to t=1M\nrcv:6 snt:4;p=88,165;i=block/rxtx,yellow
#301     1.1143682s     Display string changed for connection from module (cCompoundModule) BulkTransfer.server gate pppg$o[1] to ,1
#301     1.1143682s     Display string changed for connection from module (cCompoundModule) BulkTransfer.server gate pppg$o[1] to
#301     1.1143682s     Starting transmission of (PPPFrame)ACK
```

Unreproducible results 5

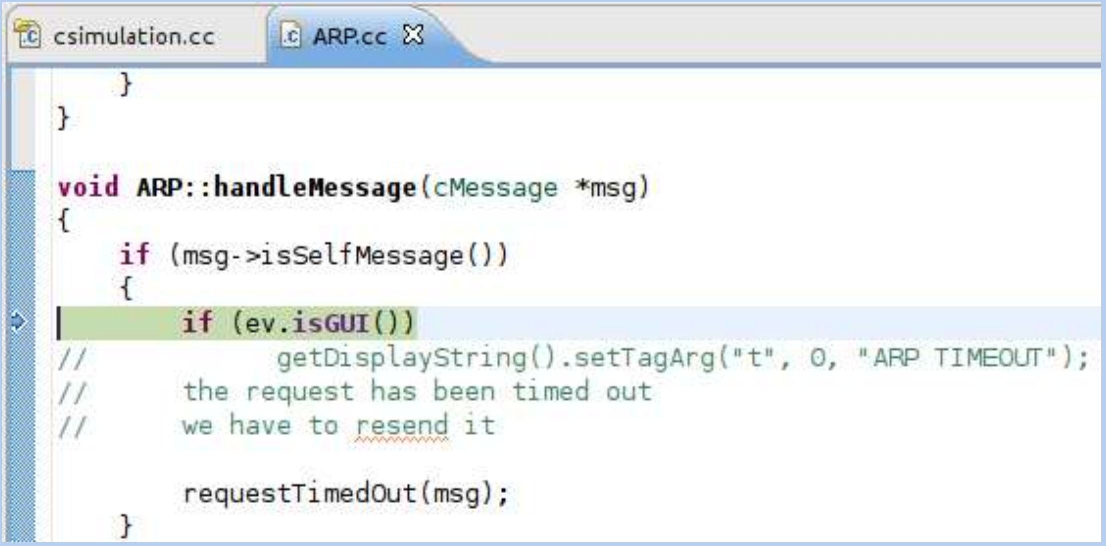
Step through the doOneEvent method and enter into the model, which will be the ARP module's handleMessage().

We will quickly realize that by mistake we have commented out the getDisplayString().setTagArg() line, but not the previous ev.isGUI() test which disabled the timeout processing in Cmdenv, but not in Tkenv. This caused the simulation to run differently...



csimulation.cc main.cc

```
                // (which originally occurre
                msg_for_activity = msg;
                transferTo(mod);
            }
            else
            {
                // if there was an error duri
                // with an exception
                mod->handleMessage(msg);
            }
        }
```



csimulation.cc ARP.cc

```
    }
}

void ARP::handleMessage(cMessage *msg)
{
    if (msg->isSelfMessage())
    {
        if (ev.isGUI())
//              getDisplayString().setTagArg("t", 0, "ARP TIMEOUT");
//          the request has been timed out
//          we have to resend it

        requestTimedOut(msg);
    }
```