CERTI Install Documentation

3.3.2cvs

Generated by Doxygen 1.5.5

Mon Nov 24 12:43:06 2008

Contents

1 index 1

2 Introduction 1

3 Building CERTI 2

4 Installing CERTI 8

5 Executing HLA simulation 8

6 Connecting to RTIG via a HTTP tunnel 12

7 Module Documentation 14

1 index

Installation Documentation

This documentation describes CERTI installation/configuration and simulation execution. If you want to develop own HLA simulations using CERTI, please refer the User documentation.

This documentation is divided into several parts:

- Introduction
- Building CERTI
- Installing CERTI
- Executing HLA simulation
  - Connecting to RTIG via a HTTP tunnel

2 Introduction

CERTI is an Open Source HLA compliant RunTime Infrastructure (RTI).

You'll find hereafter the documentation for building and installing CERTI. CERTI is primarily developed and maintained by the Toulouse research center of ONERA [http://www.onera.fr], the French Aerospace Labs. The primary goal of CERTI is to be used in research activities but CERTI has a growing number of users and contributors among the CERTI Open Source community.

People interested in CERTI may join the CERTI Open Source community at https://savannah.nongnu.org/projects/certi and/or by using the mailing list http://lists.nongnu.org/mailman/listinfo/certi-devel for discussion regarding CERTI usage.

3 Building CERTI

CERTI comes as either as an installer (binary) or compressed tar source archives.

This section describes how to build CERTI executables from the source code.

The primary distribution format is gzip compressed tar source archive (.tar.gz) or ZIP archive (.zip) and may be found on Savannah CERTI download area: http://download.savannah.nongnu.org/releases/certi/ .

CERTI build system uses CMake, http://www.cmake.org/ which is a cross-platform build system generator. CMake should be used to compile CERTI SDK on a variety of platform/compiler combination like:

- Linux x86 / gcc
- Linux x86_64 / gcc
- Solaris Sparc / Sun Studio
- Windows / Visual Studio
- Windows / Code::Blocks+MinGW
- ...

You may follow generic CMake usage instruction for building CERTI on various platforms: Running CMake, http://www.cmake.org/cmake/help/runningcmake.html://www.cmake.org/ or you may try to follow the below CERTI CMake usage.

3.1 Prerequisites

CERTI compilation requires a working C++ compiler and some development tools. You may have them already installed on your system or you may download and install the missing prerequisites. All used tools are free software.

CMake

- Windows installer
  - http://www.cmake.org/HTML/Download.html
- Fedora Linux

```
yum install cmake
```

- Debian/Ubuntu Linux

```
apt-get install cmake
```

Flex, Bison and m4

- Windows installer
  - http://gnuwin32.sourceforge.net/packages/flex.htm
  - http://gnuwin32.sourceforge.net/packages/bison.htm

  - http://gnuwin32.sourceforge.net/packages/m4.htm

- Fedora Linux

```
yum install flex bison
```

- Debian/Ubuntu Linux

```
apt-get install flex bison
```

NSIS (optionally, for building a Windows installer)

- Windows installer
  - http://nsis.sourceforge.net/Main_Page

libxml2 (optionally, to enable federation save and restore)

- Windows installer
  - http://www.zlatkovic.com/libxml.en.html
- Fedora Linux

```
yum install libxml2-devel
```

- Debian/Ubuntu Linux

```
apt-get install libxml2-devel
```

3.2 Building CERTI on Unix with Makefile generator

If you get a tarball source CERTI distribution such as you may found in the download section of the Savannah project http://download.savannah.nongnu.org/releases/certi/, you should follow these steps:

1. untar the archive: tar zxvf certi-<version>-Source.tar.gz this should create a certi-<version>-Source directory

```
tar zxvf certi-3.2.4-Source.tar.gz
... wait for tar ending ...
```

2. Prepare separate build directory and run CMake

```
mkdir build_certi
cd build_certi
cmake -DCMAKE_INSTALL_PREFIX=/path/to/install /path/to/certi-\<version\>-Source
... wait for cmake run  ending ...
```

3. compile your certi

```
make
... wait the compilation end ...
```

4. Then you may either install CERTI or build a binary package that will be usable for installation

- install CERTI

```
make install
... wait for make install end ...
```

- build your binary package

```
make package
```

After that you will have a compiled and usable CERTI package.

3.3 Building CERTI on Unix (GUI)

CMake 2.6.0 and up comes with a nice and handy graphical user interface which may be invoked with the cmake-gui command.

3.4 Building CERTI on Windows

Enter the Start menu and launch the CMake application. Enter the folder for the source code (e.g. `\certi`, created during the previous step). Enter the folder for building the programs (e.g. `\tempo`).

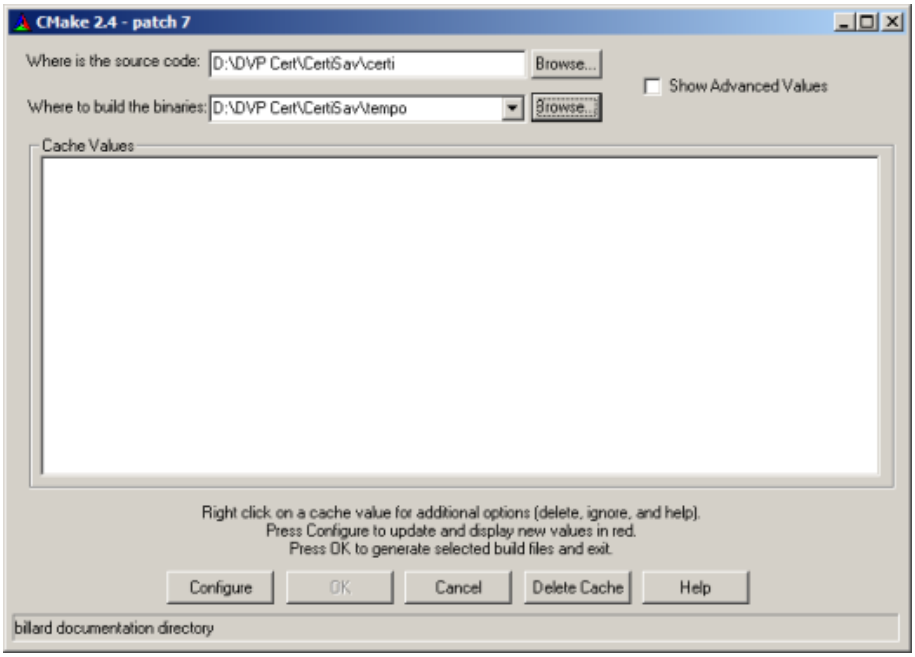

Figure 1: Launch CMake

Hit the `Configure` button.

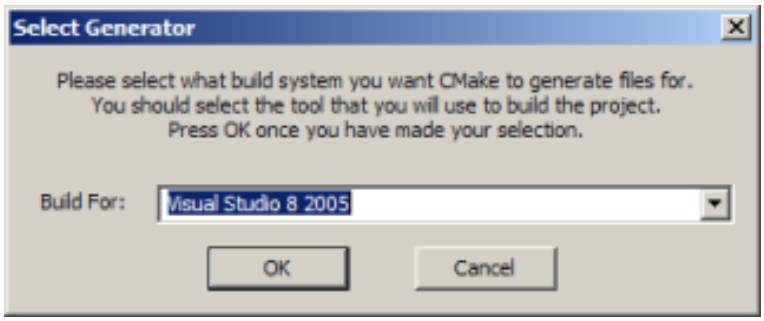

Figure 2: Configure

Choose the appropriate Generator.

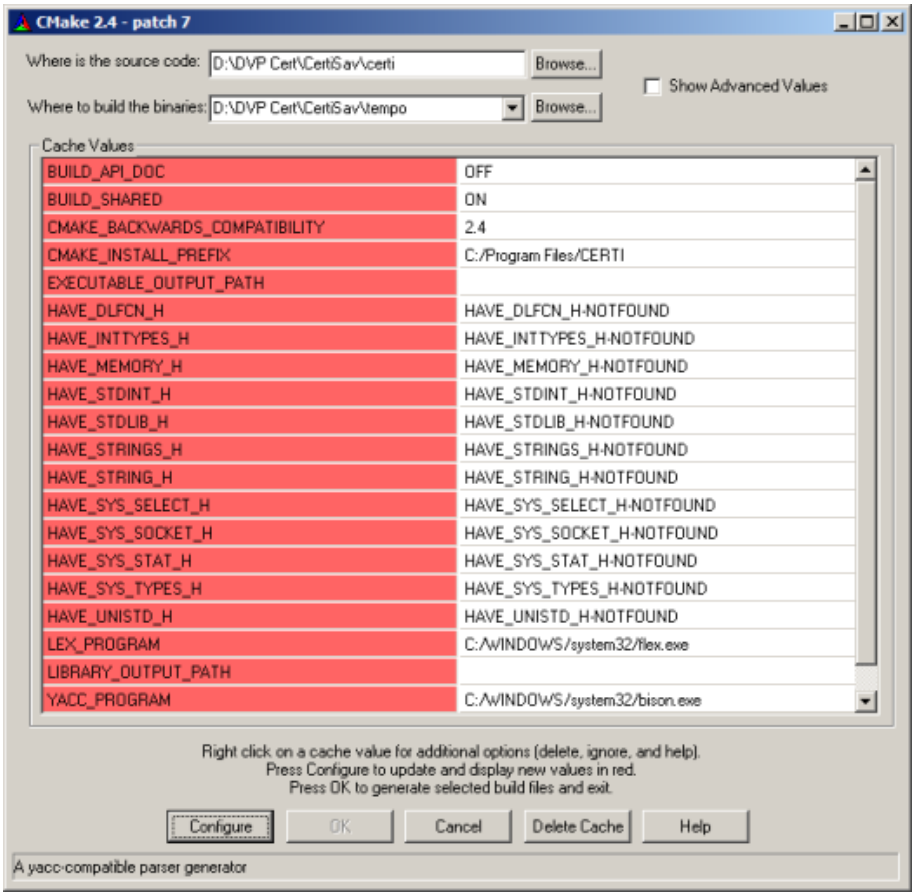

Figure 3: Choose Generator

Verify the the paths for the different tools (flex, bison) before hitting `Configure` again.

Repeat `Configure` until you get an enabled `OK` button.

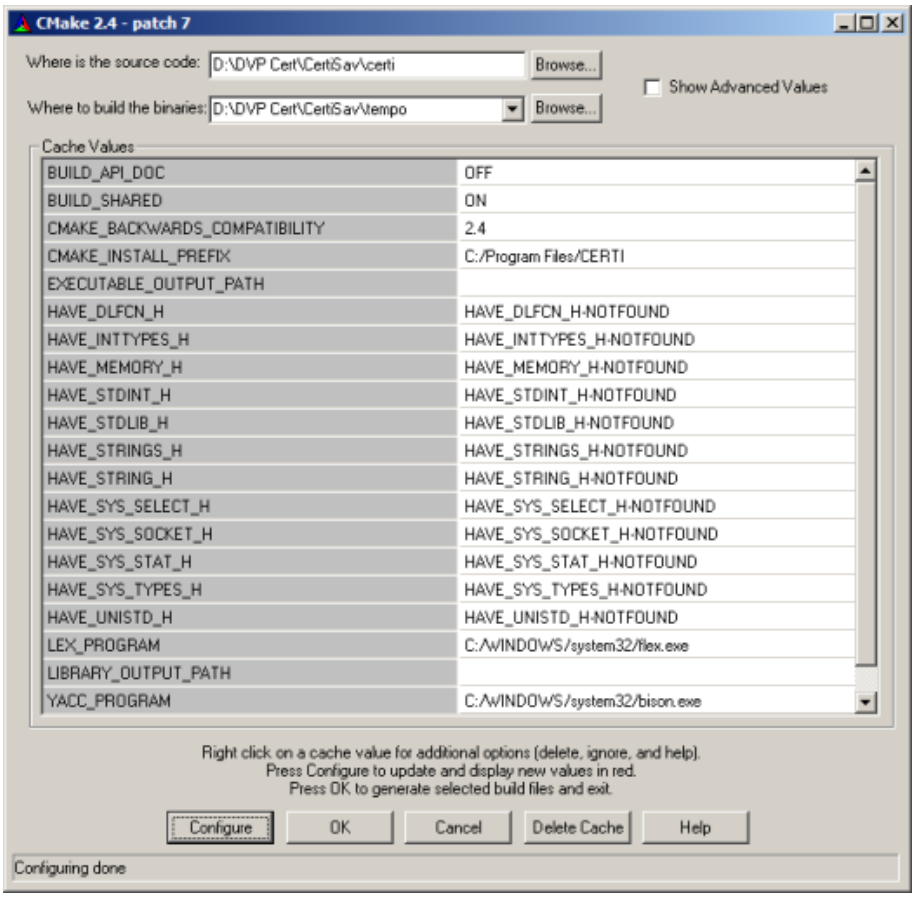

Figure 4: Generate project

You should get a `build` folder (e.g. \tempo) which looks like this:

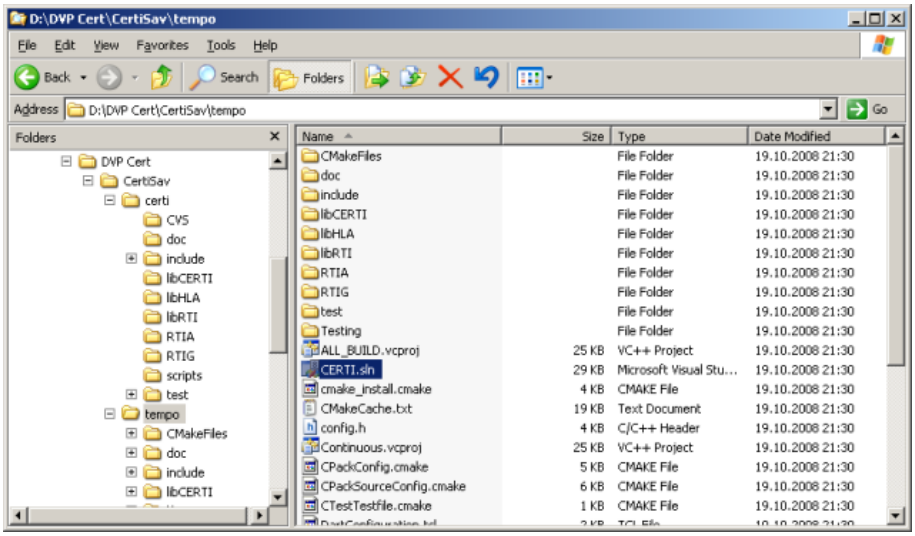

Figure 5: Visual Studio files

Launch the `CERTI.sln` (e.g. using Microsoft Visual C++ 2005).

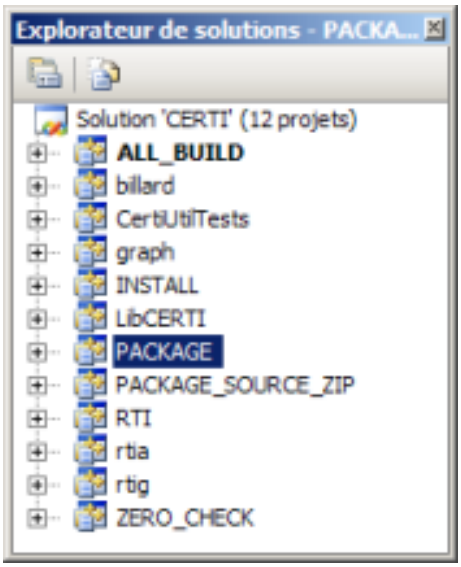

Figure 6: CMake VS Project

Select desired configuration (Debug or Release) and build the project (using Build All). Take a look at your folder:

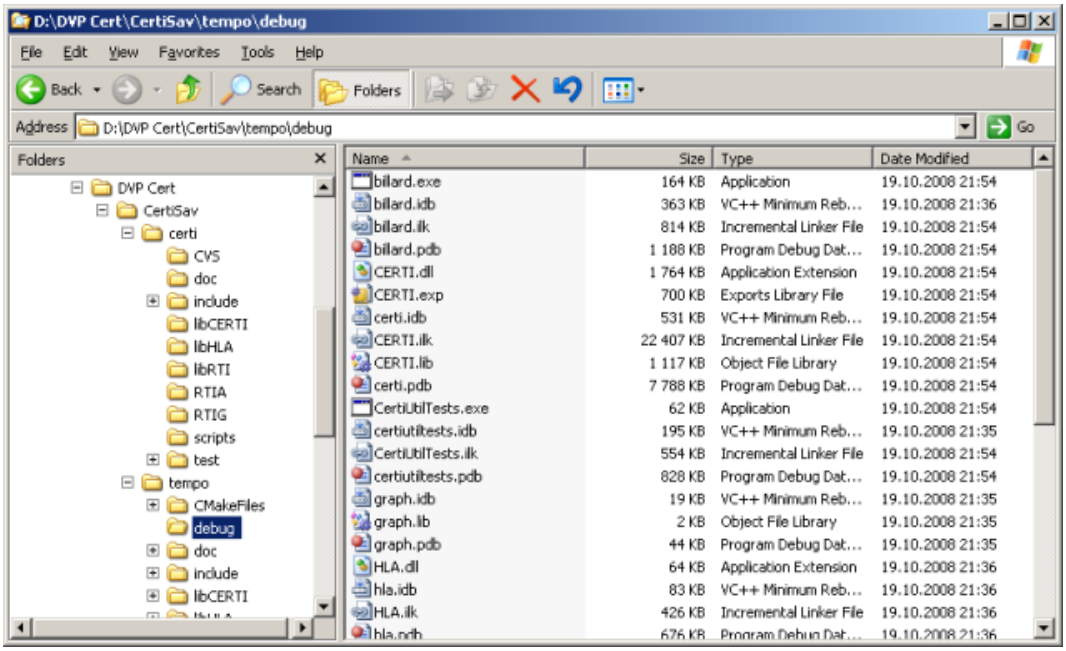

Figure 7: Build Tree

4 Installing CERTI

CERTI comes as a set of libraries and binaries executables, which may be used to build an HLA compliant simulation.

4.1 Install the CERTI software

4.1.1 CERTI installation layout

CERTI components are installed in a user-chosen CERTI_HOME directory. Below this root directory the layout is as described below:

TODO: How to set $PATH, LD_LIBRARY_PATH TODO: Where to store .fed files, what are the .fed files?

4.1.2 Windows installer

TODO: How to use the Windows installer.

4.1.3 RPM package

TODO: How to install CERTI from a .RPM file.

5 Executing HLA simulation

5.1 CERTI executables

CERTI comes with two main executables: RTIA and RTIG.

5.1.1 certi_user_execute

If ones want to properly execute an HLA simulation using CERTI one must: (FIXME more detail to come).

1. configure PATH
2. store .fed (or .xml) FOM file in the search path of the rtig

   See also:

   certi_FOM_FileSearch
3. run rtig,

   See also:

   RTIG
4. configure HOST/PORT/PROXY,
5. run federations, rtia is started automatically.

5.1.2 CERTI environment variables

CERTI uses a set of environment variables which may influence its execution behavior.

Variable	Used by	Description
CERTI_HOME	RTIG	the CERTI installation base directory. This is used by the RTIG in order to look for FOM files (see RTIG).
CERTI_HOST	RTIA	machine on which RTIG is running. As soon as it starts the RTIA will try to connect to the RTIG running on CERTI_HOST (see RTIA).
CERTI_TCP_PORT	RTIG, RTIA	TCP port used for RTIA/RTIG communications
CERTI_UDP_PORT	RTIG, RTIA	UDP port used for RTIA/RTIG communications
CERTI_HTTP_PROXY	RTIA	HTTP proxy address in the format `http://host:port`. See HTTP tunneling.
CERTI_NO_STATISTICS	RTIA	if set, do not display service calls statistics

5.1.3 RTIG: CERTI RunTime Infrastructure Gateway

The CERTI RunTime Infrastructure Gateway (RTIG) is a process which coordinate the HLA simulation with CERTI, there should be at least one rtig process for each federation.

However a single RTIG may be used for several federations. The command line usage of the RTIG is following:

rtig [-v 2]

- **-v** (optional) verbosity level
  - 0 -> no output
  - 1 -> small amount
  - 2 -> show fed parse

Once the RTIG is launched an HLA Federate may interact with the RTI. In fact a federate does not talk to the RTIG directly but it uses its RTIA. RTIG is listening to RTIA connection on TCP port:

1. 60400 or,
2. the value of environment variable CERTI_TCP_PORT if it is defined

The RTIG exchange messages with the RTIA in order to satify HLA request coming from the Federate. In particular RTIG is responsible for giving to the Federate (through its RTIA) the FOM file needed to create or join the federation.

5.1.4 RTIA: CERTI RunTime Infrastructure Ambassador

The CERTI RunTime Infrastructure Ambassador (RTIA) is a process which is automatically launched by the federate as soon as its RTIambassador is created.

The command line usage of the RTIA is following:

rtia [-v] [-p <port>]

- **-v** (optional) verbose, display more information
- **-p** (optional) tcp port to be used to communicate with FederateAmbassador

5.2 Sample federate: Billiard

Open a windows command prompt and run the RTIG.

```
rtig
```

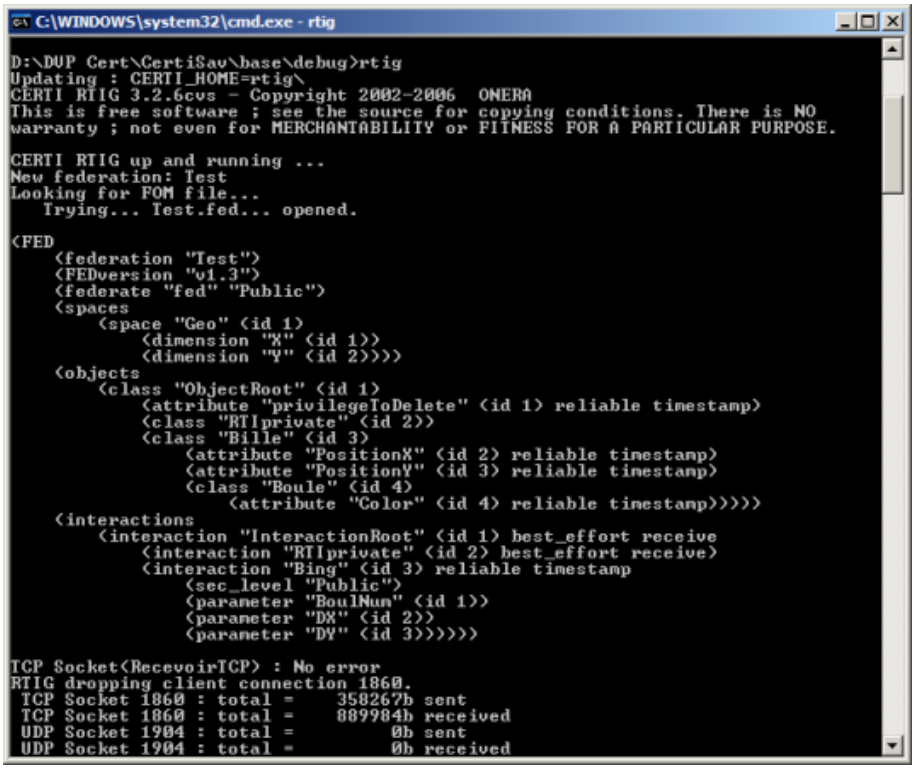

Figure 8: RTIG screenshot

Open another windows command prompt and run the billard program.

```
billiard -n 1 fTest FTest.fed
```

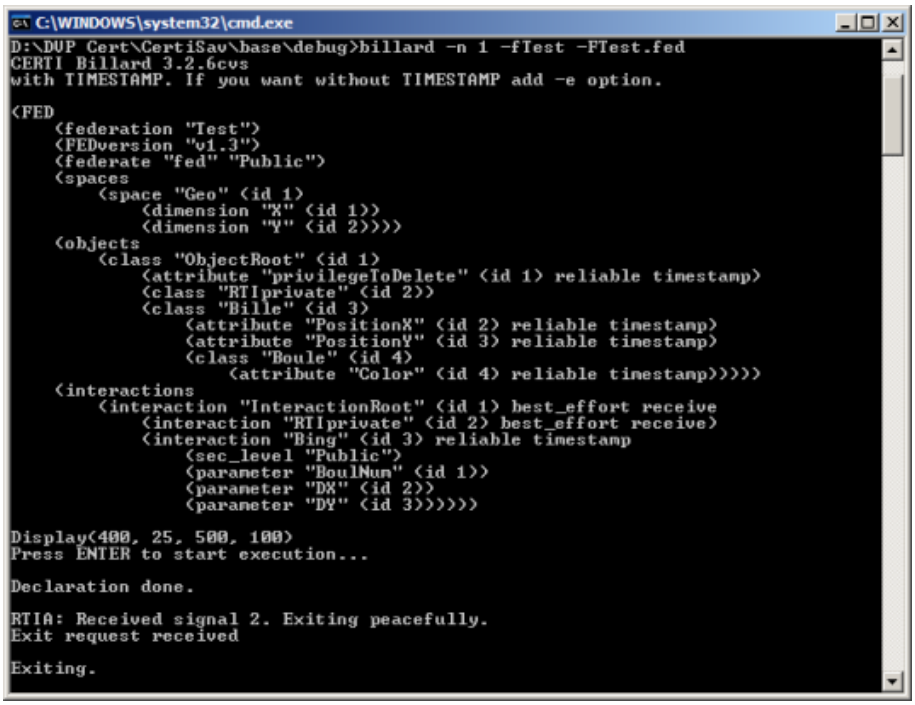

Figure 9: Billard consoleshot

6 Connecting to RTIG via a HTTP tunnel

To pass the RTIA–RTIG connection through firewalls, you may use the HTTP tunnel.

Federates behind a firewall may be unnable to connect to the RTIG. To connect via a HTTP tunnel

1. Set the `CERTI_HOST` and `CERTI_TCP_PORT` environment variables to RTIG address and port.
2. Set the `CERTI_HTTP_PROXY` environment variable to HTTP proxy address in the form `http://host:port`.
3. Run the federate.

Note: In the HTTP proxy configuration you may need to enable the HTTP CONNECT method for the port number defined in `CERTI_TCP_PORT`. For example, in the /etc/squid/squid.conf you may need to configure

```
acl CERTI_ports port 60400   # the value of CERTI_TCP_PORT
acl CONNECT method CONNECT
http_access allow CONNECT CERTI_ports
```

If `CERTI_HTTP_PROXY` is not defined, the system-wide `http_proxy` is used. To disable HTTP tunneling, you must unset both environment variables, or set `CERTI_HTTP_PROXY` to an empty string.

If the HTTP proxy is directly accessible for the federate (RTIA), you can set the `CERTI_HTTP_PROXY` environment variable to address of the HTTP proxy, e.g. `http://proxy.example.com`. The default port is 3128.

If you cannot access the HTTP proxy directly, you may use SSH port forwarding. The SSH client will listen to a local port and will ask the remote SSH server to open an outgoing connection to the HTTP proxy. It will then forward all traffic between the local port and the HTTP proxy inside the SSH connection.

To use SSH port forwarding

1. Set the `CERTI_HTTP_PROXY` environment variable to an arbitrary local port number, e.g. `http://localhost:8808.`
2. Establish an SSH connection as follows.

On Windows you may use the PuTTY client `http://www.chiark.greenend.org.uk/~sgtatham/putty`

Create a SSH session and select the SSH protocol. Open the Connection – SSH – Tunnels configuration. Select "Local", enter chosen arbitrary "Source port" number (e.g. 8808) and the HTTP proxy address as "Destination". Make sure you then click "Add".

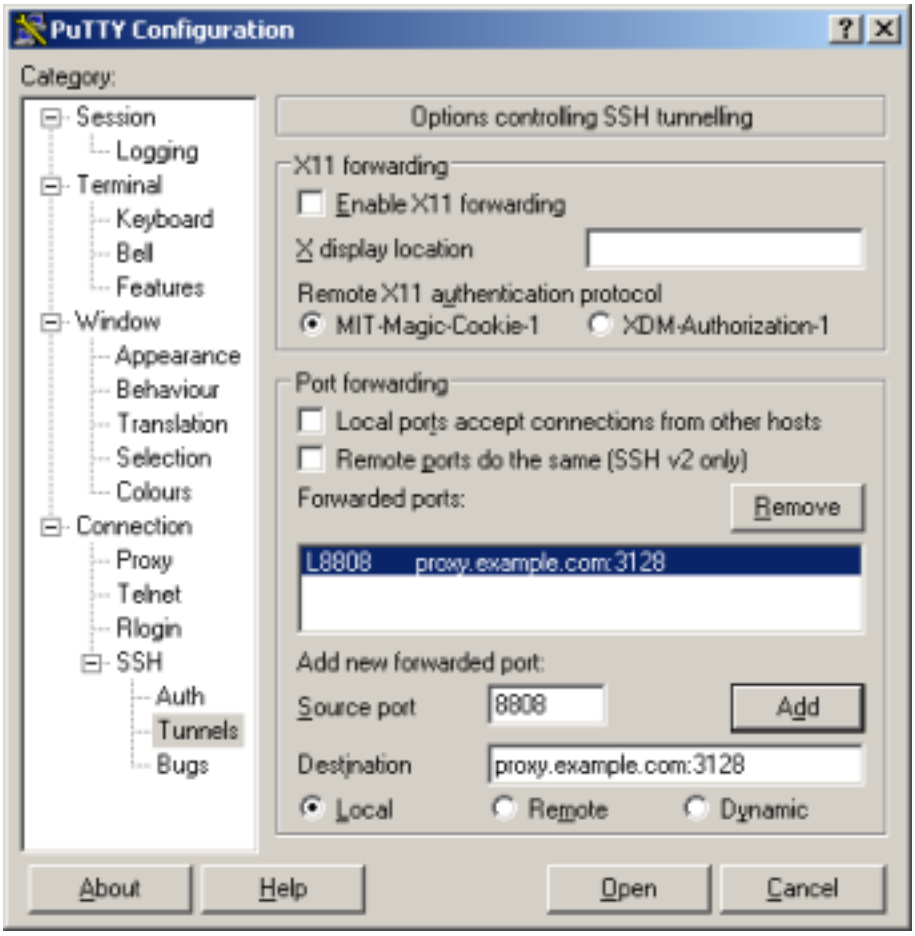

Figure 10: Putty Portforwarding

Most Linux systems have a SSH client installed. Use the `ssh` command.

```
ssh -L8808:proxy.example.com:3128 user@hostname
```

7 Module Documentation

7.1 RTIG

The CERTI RunTime Infrastructure Gateway (RTIG) is a process which coordinate the HLA simulation with CERTI, there should be at least one rtig process for each federation. However a single RTIG may be used for several federations. The command line usage of the RTIG is following:

rtig [-v 2]

- **-v** (optional) verbosity level
  - 0 -> no output
  - 1 -> small amount
  - 2 -> show fed parse

Once the RTIG is launched an HLA Federate may interact with the RTI. In fact a federate does not talk to the RTIG directly but it uses its RTIA. RTIG is listening to RTIA connection on TCP port:

1. 60400 or,
2. the value of environment variable CERTI_TCP_PORT if it is defined

The RTIG exchange messages with the RTIA in order to satify HLA request coming from the Federate. In particular RTIG is responsible for giving to the Federate (through its RTIA) the FOM file needed to create or join the federation.

7.2 RTIA

The CERTI RunTime Infrastructure Ambassador (RTIA) is a process which is automatically launched by the federate as soon as its RTIambassador is created. The command line usage of the RTIA is following:

rtia [-v] [-p <port>]

- **-v** (optional) verbose, display more information
- **-p** (optional) tcp port to be used to communicate with FederateAmbassador