Falaise/trunk installation report on (X)ubuntu 14.04 LTS (64bits)

François Mauger, LPC Caen <mauger@lpccaen.in2p3.fr>

2016-05-29

In this document we propose an installation procedure for the Falaise/trunk (pre 2.1) library on top of Bayeux/trunk (pre 2.1) and Cadfaelbrew (2016.01) on Xubuntu 14.04 LTS system.

The target system

Architecture:

```
$ uname -a
Linux ... 3.13.0-74-generic #118-Ubuntu SMP Thu Dec 17 ... UTC 2015 x86_64...
```

Processors:

```
$ cat /proc/cpuinfo | grep "model name"
model name   : Intel(R) Core(TM) i7-3540M CPU @ 3.00GHz
model name   : Intel(R) Core(TM) i7-3540M CPU @ 3.00GHz
model name   : Intel(R) Core(TM) i7-3540M CPU @ 3.00GHz
model name   : Intel(R) Core(TM) i7-3540M CPU @ 3.00GHz
```

Linux version:

```
$ cat /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=14.04
DISTRIB_CODENAME=trusty
DISTRIB_DESCRIPTION="Ubuntu 14.04.3 LTS"
```

Setup of Bayeux/trunk

You must have installed a standalone Bayeux/trunk on top of Cadfaelbrew.

Links:

- Bayeux (SuperNEMO Wiki)
- Cadfael (SuperNEMO Wiki) and Cadfaelbrew repository (GitHub)

Once you have installed Cadfaelbrew and Bayeux, you should be able to setup Bayeux:

```
$ brewsh # Enter a *brew shell*
$ do_bayeux_dev_setup
NOTICE: Cadfaelbrew is now setup !
NOTICE: Bayeux/trunk is now setup !
```

You can check the location and version of core software utilities:

```
$ which cmake
{... path to the Cadfaelbrew
     installation directory...}/supernemo/cxx11/Cadfael.git/bin/cmake

$ cmake --version
cmake version 3.4.0

$ g++ --version
g++ (Homebrew gcc49 4.9.2_2) 4.9.2

$ which bxquery
{... path to the Bayeux
     installation directory...}/bin/bxquery
```

Installation of Falaise (trunk)

This Falaise setup is built using explicitely a system Bayeux setup.

Set the software base directory where there is enough storage capacity to host Falaise (> 1 GB). Here we use a simple environment variable `SW_WORK_DIR` which points to a specific directory on the filesystem:

```
$ export SW_WORK_DIR=/data/sw
```

You should adapt this base directory to your own system, for example:

```
$ export SW_WORK_DIR=${HOME}/Software
```

Then create a few working directories:

```
$ mkdir -p ${SW_WORK_DIR}
$ mkdir ${SW_WORK_DIR}/Falaise         # This one is the base working directory for Falaise
$ mkdir ${SW_WORK_DIR}/Falaise/Source  # This one will host the source code
$ mkdir ${SW_WORK_DIR}/Falaise/Binary  # This one will host the build and installation dire
```

Download Falaise/trunk source files:

```
$ cd ${SW_WORK_DIR}/Falaise/Source
$ svn co https://nemo.lpc-caen.in2p3.fr/svn/Falaise/trunk Falaise-trunk
$ cd Falaise-trunk
```

Configure Falaise:

1. Make sure Cadfaelbrew and Bayeux are setup on your system. If you follow the Cadfaelbrew and Bayeux installation report available from the Bayeux wiki page, you just have to invoke:

```
$ brewsh
$ do_bayeux_dev_setup
```

2. Create a build directory and cd in it:

```
$ mkdir -p ${SW_WORK_DIR}/Falaise/Binary/Falaise-trunk/Build-gcc-cxx11-ninja-Linux-x86_64
$ cd ${SW_WORK_DIR}/Falaise/Binary/Falaise-trunk/Build-gcc-cxx11-ninja-Linux-x86_64
```

3. Configure Bayeux with CMake:

```
$ CADFAEL_PREFIX_DIR=$(clhep-config --prefix | tr -d '\"')
$ echo ${CADFAEL_PREFIX_DIR}
/data3/sw/CadfaelBrew
```

```
$ cmake \
 -DCMAKE_BUILD_TYPE:STRING=Release \
 -DCMAKE_INSTALL_PREFIX:PATH="${SW_WORK_DIR}/Falaise/Binary/Falaise-trunk/Install-gcc-cxx11
 -DCMAKE_FIND_ROOT_PATH:PATH="$(bxquery --prefix);${CADFAEL_PREFIX_DIR}" \
 -DFALAISE_COMPILER_ERROR_ON_WARNING=ON \
 -DFALAISE_ENABLE_TESTING=ON \
 -DFALAISE_WITH_DOCS=ON \
 -DFALAISE_WITH_DEVELOPER_TOOLS=ON \
 -DBoost_DIR:PATH="${CADFAEL_PREFIX_DIR}/lib/cmake" \
 -GNinja \
 ${SW_WORK_DIR}/Falaise/Source/Falaise-trunk
```

Build (using 4 processors to go faster):

```
$ time ninja -j4
...
```

Quick check after build

After the build step, Falaise uses the following hierarchy on the file system:

```
$ LANG=C tree -L 1 BuildProducts/
BuildProducts/
|-- bin
|-- include
|-- lib
`-- share
```

Test programs

Before to do the final installation, we run the test programs:

```
$ ninja test
...
```

Installation

Run:

```
$ ninja install
...
```

Check installation

Browse the installation directory:

```
$ LANG=C tree -L 3 -F ${SW_WORK_DIR}/Falaise/Binary/Falaise-trunk/Install-gcc-Linux-x86_64
...
```

Setup your environment for Falaise

Here we explicitely *load/setup* the Falaise environment from a Bash shell with a dedicated function defined in my `~/.bashrc` startup file:

```
# The base directory of all the software (convenient path variable):
export SW_WORK_DIR=/data/sw

# The Falaise/trunk setup function:
function do_falaise_trunk_setup()
{
  do_bayeux_dev_setup # Automatically load the Bayeux (and Cadfaelbrew dependency)
  if [ -n "${FALAISE_DEV_INSTALL_DIR}" ]; then
      echo "ERROR: Falaise/trunk is already setup !" >&2
      return 1
  fi
  export FALAISE_DEV_INSTALL_DIR=${SW_WORK_DIR}/Falaise/Binary/Falaise-trunk/Install-gcc-Li
  export PATH=${FALAISE_DEV_INSTALL_DIR}/bin:${PATH}
  echo "NOTICE: Falaise/trunk is now setup !" >&2
  return;
}
export -f do_falaise_trunk_setup

# Special alias:
alias do_falaise_dev_setup="do_falaise_trunk_setup"
```

When one wants to use pieces of software from Falaise, one runs:

```
$ do_falaise_dev_setup
```

Update the source code from the Falaise/trunk

1. Cd in the Falaise/trunk source directory:

   ```
   $ cd ${SW_WORK_DIR}/Falaise/Source/Falaise-trunk
   ```

2. Update the source code:

   ```
   $ svn up
   ```

3. Cd in the Falaise/trunk build directory:

```
$ cd ${SW_WORK_DIR}/Falaise/Binary/Falaise-trunk/Build-gcc-ninja-Linux-x86_64
```

4. Rebuild and reinstall

```
$ brewsh
$ do_bayeux_dev_setup
$ ninja -j4
$ ninja test
$ ninja install
```