Sheaf System Programmer's Guide

Linux Installation How-To

1 Installing on Linux

These notes assume you are using a csh or tcsh shell on Linux.

2 Running the installer

First decide where you want the SheafSystem Programmer's Guide installed, that is, the absolute path to the root of the SheafSystem Programmer's Guide installation tree. We'll refer to that as <install_dir> in this document. For instance, you might choose:

<install_dir> = /usr/local/SheafSystem

You do not have to create this directory first, but where ever you choose, make sure you have write access to it if it exists and to its parent folder if it does not. In the remainder of this document, mentally replace <install_dir> with your chosen path.

Now, start a nice fresh terminal window without any unknown environment lurking in its history, go to where ever you downloaded the installer to, and enter:

```
>sh sheafsystemprogrammersguide_install.bin
```

Follow the instructions given by the installer. In particular, it will ask you "Where do you want to install?". Enter your choice for <install_dir>.

When the installer is finished installing, it will present you with a "done" button. It takes a a short time after you've pushed done to completely finish. The <install_dir> folder now has several subfolders. Two of them are important in the rest of this document:

- The location of the SheafSystem binary tree, <install_dir>/SheafSystem-<version>, which we will refer to as <ss_dir>.

- The location of the SheafSystemTest source tree, <install_dir>/SheafSystemTest-<version>, which we will refer to as <sst_dir>.

3 Configuring the examples module

Change you working directory to examples directory <install_dir>/examples

```
>cd <install_dir>/examples
```

The SheafSystem libraries are delivered compiled for Gnu C++ compiler or the the Intel C++ compiler. Setting your environment depends on which compiler.

3.1 Setting your environment for the Gnu C++ compiler

For the Gnu C++ compiler, all you need to do is set the environment variable CXX for the C++ compiler and CC for the C compiler. For instance:

```
>setenv CXX /usr/bin/g++
>setenv CC /usr/bin/gcc
```

3.2 Setting your environment for the Intel C++ compiler.

The Intel compiler is not completely self-sufficient, it relies on finding the Gnu C++ standard header files, and it searches your PATH environment variable to find the Gnu compiler. The SheafSystem is compatible only with g++ version 4.2.2 or later. So you have to make sure that a g++ installation with version 4.2.2 or later is in your PATH. In particular, if your Linux installation has a g++ installation in /usr/bin that is older than 4.2.2, you have to make sure the path to a 4.2.2 or later is in your PATH *before* /usr/bin.

Now you'll need to know where your Intel compiler installation bin directory is. A common location is:

<intel_bin_dir> = /opt/intel/bin

What ever <intel_bin_dir> is on your system, the C++ compiler is <intel_bin_dir>/icpc, the C compiler is <intel_bin_dir>/icc, and the compiler environment script is <intel_bin_dir>/compilervars.csh.

Set the environment variable CXX for the C++ compiler, CC for the C compiler, and source the compilervars script.:

```
>setenv CXX  <intel_bin_dir>/icpc
>setenv CC <intel_bin_dir>/icc
>source <intel_bin_dir>/compilervars.csh intel64
```

The above asumes you are running an a 64 bit system. Remember, replace <intel_bin_dir> by the actual path on your system.

3.3 Running the configuration script

Now we're ready to configure. The examples module requires CMake version 2.8.6. If your CMake is older, you will have to upgrade. If your version is new enough, run the configuration script:

```
>./cmboot
```

The script will request the absolute path to the SheafSystem binaries installation, for instance:

/usr/local/SheafSystem/SheafSystem-3.0.10

The cmboot script will configure the examples module in a subdirectory <install_dir>/examples/build. If anything goes wrong, for instance you didn't set the CC environment variable correctly, delete the build directory before you try again. If cmboot runs ok, it will finish with a recommendation to change directory to build and source the set_env_vars.csh script. Do it.

```
>cd build
>source set_env_vars.csh
```

4 Running the examples

Now we're still in the build directory and ready to build the examples. To build exampe1 for instance:

```
>make example1
```

To build all the examples

```
>make examples
```

The set_env_vars script set LD_LIBRARY_PATH for you, so you can just run the example without any further prep:

```
>./example1
```

Roughly speaking the examples should be run in order, since some require the output of earlier examples. Example32 in the field subdirectory is a special case. It requires the output of example30, example30.hdf, which must be copied into the field directory before running example32.