Coven Installation Guide

Parallel Architecture Research Laboratory

version 1.4

Nathan DeBardeleben

March 6, 2003

Contents

1 Introduction

This document is intended to serve as an tutorial on installing Coven. Coven is a product of the Parallel Architecture Research Laboratory at Clemson University.

Coven is composed of three main components:

- a graphical front-end,
- a parallel back-end driver,
- a set of other utilities.

The graphical front-end is maintained in one project *pse* (which stands for Problem Solving Environment). The back-end driver and utilities are contained in other package, *coven*.

2 Getting Started

You will need to use CVS to get a copy of Coven and the graphical front-end.

2.1 Setting up CVS

Before using CVS, you must set an environment variable that tells it where to look for the CVS repositories. If you are using tcsh, you can do something like this (and add it to your .cshrc file):

```
setenv CVSROOT /projects/cvsroot
```

and for bash:

```
set CVSROOT=/projects/cvsroot
export CVSROOT
```

2.2 Coven Developer Guidelines

Go to *http://www.parl.clemson.edu/Coven* and download the Coven Developer Guidelines. This file also appears in the `coven-docs` CVS module.

You should read this document paying particular attention to the discussion on editors and the CVS tutorial. You may find parts of this more useful than others but is a good guideline for our group.

3 Java

The machine(s) you work on may already have Java installed, but we suggest the official Java release for Linux directly from Sun. Currently, we feel most comfortable with `j2sdk1.4.0_01`. For this Java release, Sun generally has Java install to `/usr/java/j2sdk1.4.0_01`. Check that the directory exists and that the `bin` directory, containing `javac` and `java` are present as well.

If you need to install Java, it may require root access. Seek help from a senior person in this matter if needed.

Get Java from `http://java.sun.com/downloads/`. You will need the SDK (software development kit) from the standard edition. It's a free download. We currently suggest `j2sdk1.4.0_01`.

Install java and set up your path to point to the bin directory, probably:
`/usr/java/JAVA_VERSION/bin`.

Example: `/usr/java/j2sdk1.4.0_01/bin`.

4 The PSE Front-End

The Coven graphical front-end is written in Java and allows for a visual representation of Coven programs.

4.1 Getting the PSE

You can get the PSE GUI from the Coven web site or from the CVS repository. If choosing to use the CVS approach, note that the PSE project is held in a slightly different CVS repository. DO NOT excute `cvs checkout pse` (which is a separate project) because it will try and checkout /projects/pse if you correctly set your CVSROOT. When checking out the `pse` project the directory `pse` will automatically be created in the working directory. For this document we assume that this checkout command is performed from the user's home directory.

To checkout the `pse` project do:

```
cvs -d /projects/pse/CVS/ checkout pse
```

All of the pse source code heirarchy will be placed in the `pse` directory in the directory you executed the above command from.

4.2 Setting Up Your Environment

Before you can make the Java PSE you will need to first set your `CLASSPATH` environment variable. Let's assume you checked out the `pse` project to `/parl/yourid/pse`. Then, if you are using tcsh or csh set your `CLASSPATH` with the command (or add to your .cshrc):

```
setenv PSE /parl/yourid/pse
setenv JA $PSE/parl/pse/jars
setenv CLASSPATH $PSE/:$JA/log4j-1.2.6.jar:$JA/sshtools-j2ssh-0.0.5-alpha.jar:
$JA/castor-0.9.4.1.jar:$JA/castor-0.9.4.1-xml.jar:$JA/xerces-J_1.4.0.jar
```

The above `CLASSPATH` line is *all on one line.* You may need to `source .cshrc` or relogin to get the above changes into your environment (try the `env` command).

4.3 Compiling the PSE

You will need to compile a number of things in a particular order:

1. pse/parl/pse/cecaad
2. pse/parl/pse/util
3. pse/parl/pse/editor
4. pse/parl/pse/coven

Change to each directory in turn and run `make` to compile the files in that directory.

4.4 Running the PSE

You are ready now to actually run the graphical environment. The *Coven Tutorial* goes step-by-step through how to actually move around in the environment and get comfortable with it. At this time, you should just make certain that you have set the CLASSPATH correctly and that you were able to compile the source code correctly. The easiest way to do this is to just run the PSE.

You can run the PSE by going to the `pse/parl/pse/editor` directory and running
`java parl.pse.editor.Editor`.

The *Editor* is used in a number of PARL projects besides Coven. To run the Coven-specific editor go to `pse/parl/pse/coven` and run
`java parl.pse.coven.CovenEditor`.

You may find the scripts `nbody.sh` and `heat.sh` (in the `pse/parl/pse/coven` directory) useful for loading a pre-built design. Additionally, the Coven tutorial document (`coven-docs/tutorial` in CVS or also available from the web site) will instruct you in a tutorial use of the Coven GUI.

4.5 Starting a Coven Program

The *Coven Tutorial* will instruct you in how to run a sample Coven program through the GUI.

5 The `coven` CVS Project

Now that we have set up the Java GUI we must get the rest of Coven. Coven is maintained in a series of modules which you can check out from CVS with the command `cvs checkout <CVS-module-name>`. These modules are:

- coven-docs
- coven-engine
- coven-language
- coven-module-library
- coven-module-parser

You can check the modules out individually, or just check them all out at once with the command `cvs checkout coven`. For this document it is assumed that you have checked out `coven` into your home directory.

5.1 Module Parser

The module parser is maintained in the `coven-module-parser` directory and CVS module of the Coven project (it can also be found in the utils package on the web site). The parser can be built with `make` command from within the `coven-module-parser` directory. Module C source files with some non-C directives are processed by the parser and converted to remove the non-C directives and replace them with Coven library calls.

As an example, see the file `coven-module-library/nbody/nbody_compute_forces.cov`. This is a pre-processed module file from the `coven-module-library` CVS repository or available in the N-Body module package on the web site. Now run this through the module parser with the command:

```
./module_lang -i ../coven-module-library/nbody/nbody_compute_forces.cov
```

Notice that some macro calls were added as well as some comments. This can be automatically written to a file by using the `-o` option to `module_lang`.

6 Program Compiler

The Coven program compiler is maintained in the `coven-language` directory and CVS module of the Coven project. It is also available in the utils package from the web site. The compiler can be built with the `make` command from within the `coven-language` directory. Coven source programs are translated by the compiler into a series of C data structures which are read dynamically by the backend driver.

As an example, see the file `coven-language/language-example.txt`. This is a basic program to perform an N-Body calculation of 900 bodies for 1000 iterations on 2 processors.

To run this through the compiler/parser use the command:

```
./Coven_lang -i language-example.txt -o output.txt
```

Look at `output.txt` and notice that it is now in a form that is considerably less easy to read and understand what the program does.

7 Parallel Backend

The parallel backend is maintained in the `coven-engine` directory of Coven and CVS project (it is also available on the web site). You will need to unpack the Coven engine on whatever machine you want to run the parallel backend (such as one of our Linux Beowulf clusters).

If you followed the above instructions and checked out `coven` on your workstation then you will have the `coven-engine` directory inside of the `coven` directory. This engine must run on a machine with MPI/MPICH and is probably in the wrong place right now (this is why we have separate CVS modules and you can check them out separately). On the parallel machine you want to use you should move the `coven-engine` directory to your home directory (probably `/home/userid/coven-engine`. It is important to remember that on most parallel clusters the networked user directory (generally `/parl/userid`) will not be visible to all the nodes of the cluster. Therefore, you must place the Coven engine in your *cluster* networked directory - generally `/home/userid`.

In the `coven-engine` directory you should find the `src` (or source) directory. Go there an run `make`. This should compile the engine and any modules you have in the `src/modules` directory (this is where the Coven GUI will transfer your modules to for compilation).

You can run the driver with the `bin/coven-driver-run.sh` command. It requires a number of arguments: the number of slave processors, the number of master tasks, and the number of slave tasks on each slave processor. This can be confusing but is handled for you automatically by the GUI. Furthermore, if you do not have any modules yet in the `coven-engine/src/modules` directory or have yet a `coven-engine/src/coven_program.c` file this will not work. The tutorial will help you get this working.

8 OpenGL Visualizations

We have a few OpenGL visualizations which are used by some of the Coven programs. Look for the projects `glpv` and `glcfd` in the CVS tree. Check them out, compile them, and be ready to hook them up as visualizers if you want to visualize the output of the N-Body or CFD Heat Transfer Coven programs.

Compiling these may be difficult if your machine is not setup correctly for OpenGL. This may take some work to get going.

9 Conclusion

Once you have Coven properly installed you should have installed:

- Java,
- the Coven GUI front-end,
- the Coven language compiler,
- the Coven module parser,
- the Coven module library (whatever portions you want),
- the Coven engine on a parallel machine with MPI,
- the OpenGL visualizations (whatever ones you want).

The next step is to try the tutorial. This can be found on the web site or in the documents directory.