

# **Parallel Programming Tutorials**

Remote Development and working with PTP

Ventsislav Petkov  
SS2008

## Outline

Remote System Explorer.....	3
New remote connection.....	3
New remote workspace folder.....	3
New project folder in your remote workspace.....	3
Configuration of the Parallel Language Development Tools(PLDT).....	4
OpenMP Include Folder.....	4
MPI Include Folder.....	4
Configuration of a resource manager.....	5
Remote Environment (Remote Tools).....	5
OpenMPI resource manager (ORTE).....	6
Creation of a new project.....	7
Empty OpenMP project.....	7
Make over SSH.....	7
Local source folder.....	8
Remote folder.....	8
Sample Makefile.....	8
Run a remote OpenMP program.....	9
Run configuration.....	9
Environment variable.....	9
Standard output from an remote program.....	9

## Remote System Explorer

The Remote System Explorer (RSE) is a toolkit in the Eclipse Workbench that allows you to connect and work with a variety of remote systems. You can look at remote file systems, transfer files between hosts, edit files remotely and execute commands.

Activate the RSE Perspective from:

*Window->Open Perspective->Other->Remote System Explorer*

### ***New remote connection***

Switch to the Remote System Explorer Perspective.

Right-click in the *Remote Systems* view and select *New connection*. If a remote connection is already selected, then right-click and select *New->Connection*.

For the *System type* select **SSH Only** and click *Next*.

The *Host name* is **hlrb2.lrz-muenchen.de**.

The *Connection name* can be freely chosen or you can leave the default one.

You can type some information about the connection in the *Description* field if you want.

Click *Next* and select **SSH Connector Service**.

Click *Finish* to create to create the new remote connection. Your new remote connection will appear in the *Remote Systems* view.

In the *Remote Systems* view click on the arrow before the name of your remote connection. You should be able to see two sub items: *Sftp Files* and *Ssh Shells*. The first is used to access the remote file system.

Click on the arrow before *Sftp Files* to get two options: *My Home* and *Root*. The first is your home folder on the remote system and the second is the top-level folder there. You will mainly use the *My Home* option to browse your files.

The first time you try to browse the remote system, you will be asked for your username and password. This should be your *h039uxx* username and the corresponding password.

### ***New remote workspace folder***

To create a new folder, right-click on *My Home* and select *New->Folder*.

Type the name of your new folder (i.e. workspace).

Your folder should appear under *My Home* and in your home folder on the remote system.

### ***New project folder in your remote workspace***

Right-click on your workspace folder (under *My Home*) and create a folder for your project. Repeat this procedure before you start working on a new project.

## Configuration of the Parallel Language Development Tools(PLDT)

The PTP Parallel Language Development tools (PLDT) provide tools to aid in OpenMP and MPI Programming. Features of the PLDT include:

- Analysis of C and C++ code to determine the location of OpenMP and MPI Artifacts
- Content assist via Ctrl+Space that completes API names while typing and fills in the arguments
- Hover help that shows API names, arguments, and descriptions
- Reference information about the OpenMP and MPI calls via F1 (Ctrl-F1 on Linux; Help key on Mac)
- OpenMP problems view of common errors, OpenMP "show #pragma region" action, OpenMP "Show Concurrency" action
- MPI Barrier Analysis to detect potential deadlocks in MPI programs

We need to specify the location of the header files for both OpenMP and MPI before using the PLDT. Just the header files are required in order to know which "artifacts" should be located.

The configuration of PLDT is under:

*Window->Preferences->PTP->Parallel Language Development Tools*

### ***OpenMP Include Folder***

Click on *New* and type the following include path:

***/usr/local/dist/DIR/eclipse-3.3/include\_omp***

Click *OK* to set the path and again *OK* to close the window.

### ***MPI Include Folder***

Click on *New* and type the following include path:

***/usr/local/dist/DIR/eclipse-3.3/include\_mpi***

Click *OK* to set the path and again *OK* to close the window.

## Configuration of a resource manager

PTP uses the term resource manager to refer to any subsystem that controls the resources required for launching a parallel job. If the target parallel system employs a job scheduler for controlling access to compute resources, then the job scheduler would be considered the resource manager for the system. For a cluster with Open MPI installed, the Open MPI runtime system would be considered the resource manager.

Currently, two different resource managers can be used to run parallel programs on the HLRB2 system: *ORTE(OpenMPI)* and *Intel MPI*. However, debugging is currently supported only with the *ORTE* resource manager.

These managers do not use the *PBS job scheduler(batch run)* and run in interactive mode. In the future there will be another resource manager that will work with *PBS*. For now, if you want to run huge projects, please use the batch mode.

In order for the communication between the local machine and the HLRB2 to work, there are two options: RSE and Remote Tools. The latter is preferred because it can automatically tunnel all traffic in a single ssh connection. This is mainly needed because of the restrictive firewall configurations.

### **Remote Environment (Remote Tools)**

The Remote Tools are a light weight ssh-based implementation which comes as a part of PTP. The configuration of different connection profiles and their status can be access from the Remote Environments view:

*Window->Show View->Other->Remote Environments*

A new connection profile can be defined at any time or during the configuration of a new resource manager. In case you made a mistake with the settings, you can always change them from the above mentioned view. You can also start/stop different remote connections from there.

When creating a new profile, use the following settings:

- Target name: the profile name – can be freely chosen
- Remote host: select this to connect to a remote host
- Host: **hlrb2.lrz-muenchen.de**
- Port: 22
- User: your **h039uux** username
- Public key based authentication: the path to your **private** ssh key (`~/.ssh/id_rsa`) and, in case you have a passphrase, type it in the corresponding field
- Timeout(sec.): min. **60** (click Advanced to access this option!)

## ***OpenMPI resource manager (ORTE)***

Switch to the PTP Runtime Perspective. To open the PTP Runtime perspective, select *Window > Open Perspective > Other...* and choose *PTP Runtime* from the list.

After switching to the PTP Runtime perspective, right-click in the *Resource Managers* view and select *Add Resource Manager*.

Choose *ORTE* as a resource manager type and click *Next*.

For the *Remote service provider* choose *Remote Tools*.

If you already have a remote tools connection profile, choose that for the *Proxy server location*. If you do not have a profile, click *New* and follow the instructions above: [Remote Environment \(Remote Tools\)](#).

***Path to proxy executable is /lrz/sys/parallel/openmpi/ptp/ptp\_orte\_proxy***

Select *Use port forwarding* to tunnel all the traffic in one ssh connection.

Either click *Finish* or go through the rest of the configuration setting. Normally, the default setting are correct.

## Creation of a new project

Unfortunately, complete remote development is currently not supported with the C/C++ Development Tooling (CDT). This should change with the next release of CDT (ver. 5.0) later this year. Until the release of the new version of CDT we should use a workaround that will enable us to develop remote applications. We will create a local C/C++ project with two different source folders: a local one and a remote one. The source files should be copied/synchronized within eclipse between the local and the remote folder. The program will be build using a *makefile* on the remote system and changing the default make command to run over ssh.

### ***Empty OpenMP project***

Start the new project by running the new project wizard.

*File->New->Project*

Select *C->C Project* and click *Next*.

Choose a name for you new project and select *Executable->OpenMP Empty Project* for the project type. Click *Next*.

Check if the default *OpenMP Project Settings* are correct. The include path should be the one specified when you configured OpenMP.

Click *Finish* to create your new project. It should appear in the *Project Explorer* of CDT.

### ***Make over SSH***

After creating a new project, you need to change the default build command to execute *make* over ssh and build the project on the remote system.

Right-click on your new project and select *Properties*.

In the Properties window select *C/C++ Build*.

Deselect *Use default build command* and type the following build command:

```
ssh <h039uxx>@h1rb2.lrz-muenchen.de cd ~/workspace/<proj_folder> && make
```

Replace:

- <h039uxx> with your username
- <proj\_folder> with the folder you created using RSE for your new project

Deselect *Generate Makefiles automatically*.

Click *OK* to confirm your new settings and close the Properties window.

## ***Local source folder***

Right-click on your project and select *New->Source Folder*.

Choose a name for the new source folder (e.g. src) and click *Finish* to create it.

## ***Remote folder***

Right-click on your project and select *New->Folder*.

Click *Advanced* and select *Link to folder in the file system*.

Choose **RSE** for the type of the file system and click *Browse*.

In the new window select the correct RSE remote connection, i.e. the connection you created in the Remote System Explorer Perspective.

Under *My Home* select the folder on the remote system that will be used for this project and click *OK*.

The folder name is automatically set to that of the remote folder. If you do not like it, you can always choose a different name here. This name will be just shown in your project – it does not affect the name of the folder on the remote system.

Click *Finish* to finally create the folder.

## ***Sample Makefile***

```
#####  
## Sample Makefile  
#####  
CC = icc  
CFLAGS = -openmp -O3 -Wall -pedantic  
DEBUG = 0  
  
ifeq '$(DEBUG)' '1'  
    CFLAGS = -openmp -g -Wall -pedantic  
endif  
  
all: hello  
  
hello: hello.c  
    $(CC) $(CFLAGS) -o hello hello.c  
  
.PHONY: clean  
clean:  
    rm -fr hello
```

## Run a remote OpenMP program

Before you can execute the program on the remote system, you need to build it. The current version of Eclipse tries to build the project automatically. However, that works mainly for simple local projects. To be able to manually build your projects, deselect the *Build Automatically* option from the *Project* menu. After that, you can always right-click on your project and choose *Build Project* to actually build it or create different *make targets*.

Whenever you try to execute a project, Eclipse launches *make* to update the project. However, it takes a lot of time to actually login to the remote system and build the project. You can disable this option from:

*Window->Preferences->Run/Debug->Launching->Build (if required) before launching*

### Run configuration

The run configuration defines execution specific settings like: arguments to pass to the program, running environment, number of processes to use, default debugger, etc.

To open the run configuration window, right-click on you project and select *Run As->Open Run Dialog* or from *Run menu -> Open Run Dialog*. Double-click on *Parallel Application* in the Run window to create a new configuration for your project.

Use the following run configuration settings:

- Main tab
  - Resource Manager: the resource manager you created. It should be already running to see it there
  - Parallel Project: your local project. It should be correct by default. If it is not, you can click *Browse* and select the correct one.
  - Application program: the executable file on the remote system. Click *Browse* to find that file. After building your program, it should be in the folder of your project in the remote workspace.
- Resources tab
  - Number of Processes: number of MPI processes to run. To run OpenMP programs use only **1** process.
- Debugger
  - Debugger: **SDM** (Scalable Debug Manager)
  - Path to debugger executable: **/lrz/sys/paralle/openmpi/ptp/sdm**
  - Debugger session address: **localhost**
- Arguments tab – any program arguments you want to pass to the program.
- Environment tab – the running environment – see below.

## ***Environment variable***

You can specify the number of threads that your OpenMP program will use with one environment variable, i.e. **OMP\_NUM\_THREADS**. You need to set this environment variable in the *Environment* tab of your run configuration.

Change to the *Environment* tab and click *New*.

Name: **OMP\_NUM\_THREADS**

Value: the number of threads you want

Click *OK* to set the new variable and select *Replace native environment with specified environment*.

## ***Standard output from an remote program***

After clicking *Run*, your current perspective will be switched to the PTP Runtime and the remote program will launch. There you can see that program in the *Jobs* view. Select the job and double click on the process of that job to see the standard output of your program.

If the process is green, the program is still executing. When it finishes, the colour will change to red.