Module 5: PTP and Parallel Language Development Tools

- Objective
  - Learn to develop and run a parallel program

- Contents
  - Learn to use PTP’s Parallel Language Development Tools
  - Learn to launch a parallel job and view it via the PTP Runtime Perspective

LACSI 2006
Parallel Tools Platform (PTP)

The Parallel Tools Platform aims to provide a highly integrated environment specifically designed for parallel application development.

Features include:
- An integrated development environment (IDE) that supports a wide range of parallel architectures and runtime systems
- A scalable parallel debugger
- Parallel programming tools (MPI/OpenMP)
- Support for the integration of parallel tools
- An environment that simplifies the end-user interaction with parallel systems

http://www.eclipse.org/ptp
Parallel Language Development Tools (1)

Features

- Analysis of C and C++ code to determine the location of MPI and OpenMP Artifacts
- "Artifact View" indicates locations of Artifacts found in source code
- Navigation to source code location of artifacts
- Content assist via `ctrl+space`
- Hover help
- Reference information about the MPI and OpenMP calls via Help
  - `F1` on Windows
  - `ctrl-F1` on Linux
  - `Help` on Mac
Parallel Language Development Tools (2)

- These features for MPI were available in the PTP v1.0 MPI Development Tools
- New features for PLDT v1.0:
  - OpenMP features similar to MPI
  - OpenMP problems view of common errors
  - OpenMP “show #pragma region” action
  - OpenMP “show concurrency” action
  - MPI New project wizard automatically configures Managed Make MPI projects.
To use the PTP Parallel Language Development Tools feature for MPI development, you need to:

- Specify the MPI include path in preferences

Open Window Preferences...
- Open the PTP item
- Open the Parallel Language Development Tools item
- Select MPI
- Select New... to add MPI include path

If running OpenMP, add its include file location here too.
Managed Make Project Setup

- Create a new **Managed Make C Project**
- Name the project e.g. ‘helloMPI’
- After the **Additional Project Settings** wizard page, you should see the **MPI Project Settings** wizard page
- Check **Add MPI include paths to this project** to add include and lib paths to MPI projects
- Currently only works on Managed Make C projects
- (Note: we plan to add a **New MPI Project** project type in a later version)
Changing the MPI build Command

If you wish to change the way your MPI program is built:

- Open the project properties
- Select C/C++ Build
- Select GCC C Compiler
  - Change the command
- Select GCC C Linker
  - Change the command
- It’s also possible to change compiler/linker arguments
Content Assist

- Type an incomplete MPI function name e.g. "MPI_Init" into the editor, and hit **ctrl-space**

- Hover over the MPI Artifact identified in the source file to see additional information about that function call, for example

---

**Module 5**

LACSI 2006
Context Sensitive Help

- Press help key when the cursor is within a function name
  - Windows: F1
  - Linux: ctrl-F1
  - MacOS X: Help
- A help view appears (Related Topics) which shows additional information
- Click on the function name to see more information
Create Source File

- Create new source file called ‘mpitest.c’
  - Right click on project
  - Select **New** ▶ **Source File**
  - An editor view will automatically open on the empty file
- Double-click on any source file in project view to open an editor on that file
Enter Program

🔹 Type in hello world program

```c
#include <stdio.h>
#include <mpi.h>
int main(int argc, char *argv[]) {
    int rank;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    printf("my rank=%d\n", rank);
    MPI_Finalize();
    return 0;
}
```

🔹 Try content assist
🔹 Try context sensitive help
Project Created and Built

- Save the source file
- Should build automatically
- Console shows results of build
Show MPI Artifacts (1)

- Select the source folder/file to analyze, to find the MPI artifacts
- Click the **Add MPI Artifacts** button in the tool bar to annotate source with markers
- Click the **Clear MPI Artifacts** to remove all the MPI artifacts
Show MPI Artifacts (2)

- Markers indicate the location of the artifacts in the editor
- In **MPI Artifact View** sort by any column (click on column heading)
- Navigate to source code line by double-clicking on the artifact
- Run the analysis on another file and its markers will be added to the view
OpenMP tools

- Similar functions to the MPI tools
- Find artifacts, sort, navigate to source code
- Help, content assist, etc.
- Show #pragma regions
- Show Concurrency (next slide)
Show Concurrency

- Select a statement
- Select the context menu on the highlighted statement, and click **Show concurrency**
- Other statements will be highlighted in yellow
- The yellow highlighted statements can execute concurrently to the selected statement
PTP Runtime Perspective (1)

- Jobs view
- Machines view
- Processes on node
- Node details

To see **Jobs** view beside (not behind) **Machines** view, drag its tab to the right until its outline occupies about half of the Machines view.
PTP Runtime Perspective (2)

- Double-click a node in machines view to see more info
- Hover over node or process for tooltip popup
- Double-click a process to see process detail
- Process output
Process and Job Icons

★ Use legend icon in toolbar

Legend

Node Colors
- DOWN: Gray
- UNALLOCATED: White
- ALLOCATED TO YOU EXCLUSIVELY, BUT IDLE: Green
- ALLOCATED TO YOU SHARED, BUT IDLE: Green
- ALLOCATED TO SOMEONE ELSE EXCLUSIVELY: Red
- ALLOCATED TO SOMEONE ELSE SHARED: Red
- JOB RUNNING: Green
- JOB STOPPED: Green
- ERROR: Red
- UNKNOWN / UNDEFINED: White

Process Colors
- STARTING: Blue
- RUNNING: Green
- EXITED NORMALLY: Red
- EXITED WITH SIGNAL: Red
- STOPPED: Red
- ERROR: Yellow

Module 5
LACSI 2006
Running a Parallel Program (1)

- Create a parallel launch configuration:
  - Open the run configuration dialog
  - Run Run...
  - Select Parallel Application
  - Select the New button
Running a Parallel Program (2)

- In **Main** tab, select **Browse** button to find the **Application program** (executable)
- Probably under **Debug** configuration
Running a Parallel Program (3)

- Select **Parallel** tab
- Enter the number of processes for this job
- 4 is a good number for this tutorial
Running a Parallel Program (4)

- In **Debugger** tab, select **SDM** from the **Debugger** drop-down menu
- Select **Run** button to launch the application
Viewing Program Output

- Double-click on process 0 (diamond icon) in the jobs view
- Standard output will be visible in process detail view
- Double-click on other processes to see their stdout