



parallel tools platform

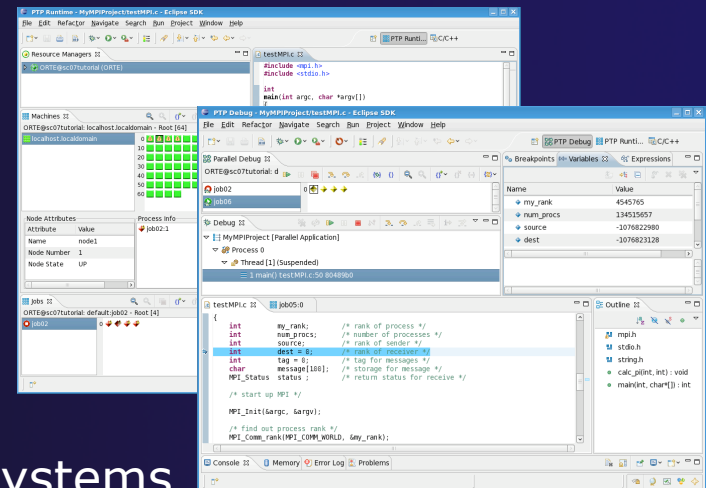
<http://eclipse.org/ptp>

# PTP Overview

Beth R. Tibbitts  
beth@tibweb.com  
SC13 PTP BOF  
November 19, 2013

# Parallel Tools Platform (PTP)

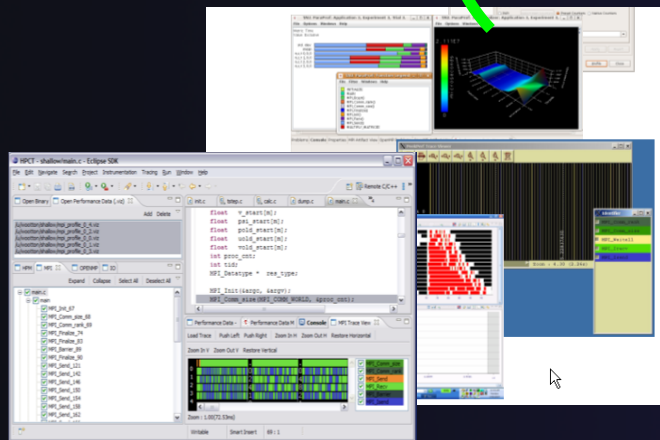
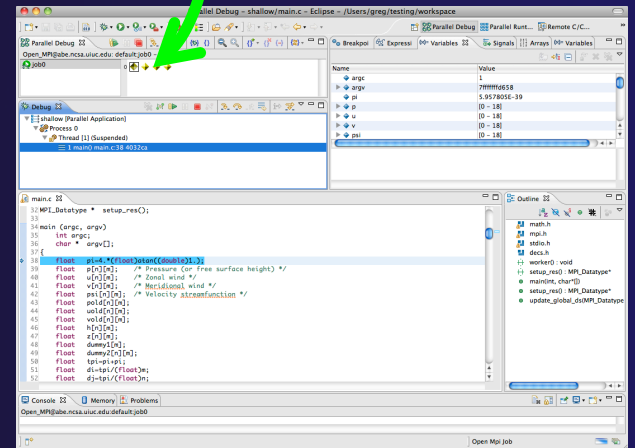
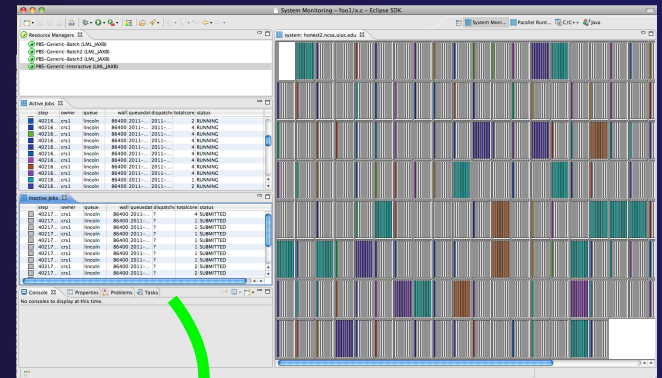
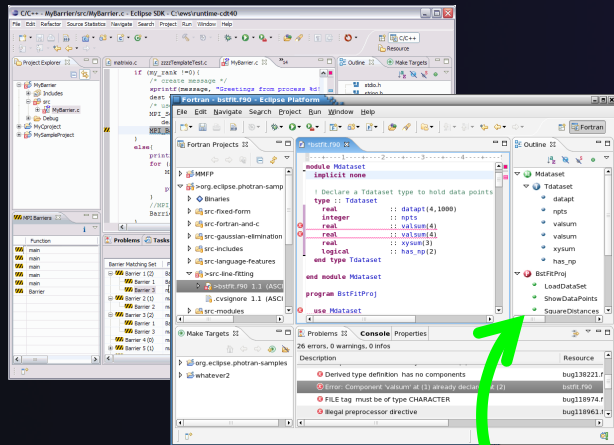
- ★ The Parallel Tools Platform aims to provide a highly integrated environment specifically designed for parallel application development
- ★ Features include:
  - ★ An integrated development environment (IDE) that supports a wide range of parallel architectures and runtime systems
  - ★ A scalable parallel debugger
  - ★ Parallel programming tools (MPI, OpenMP, UPC, OpenSHMEM, OpenACC, etc.)
  - ★ Support for the integration of parallel tools
  - ★ An environment that simplifies the end-user interaction with parallel systems
- ★ <http://www.eclipse.org/ptp>



# Eclipse PTP Family of Tools

Coding & Analysis  
(C, C++, Fortran)

Launching & Monitoring



Performance Tuning & Analysis  
(TAU, GEM, ...)

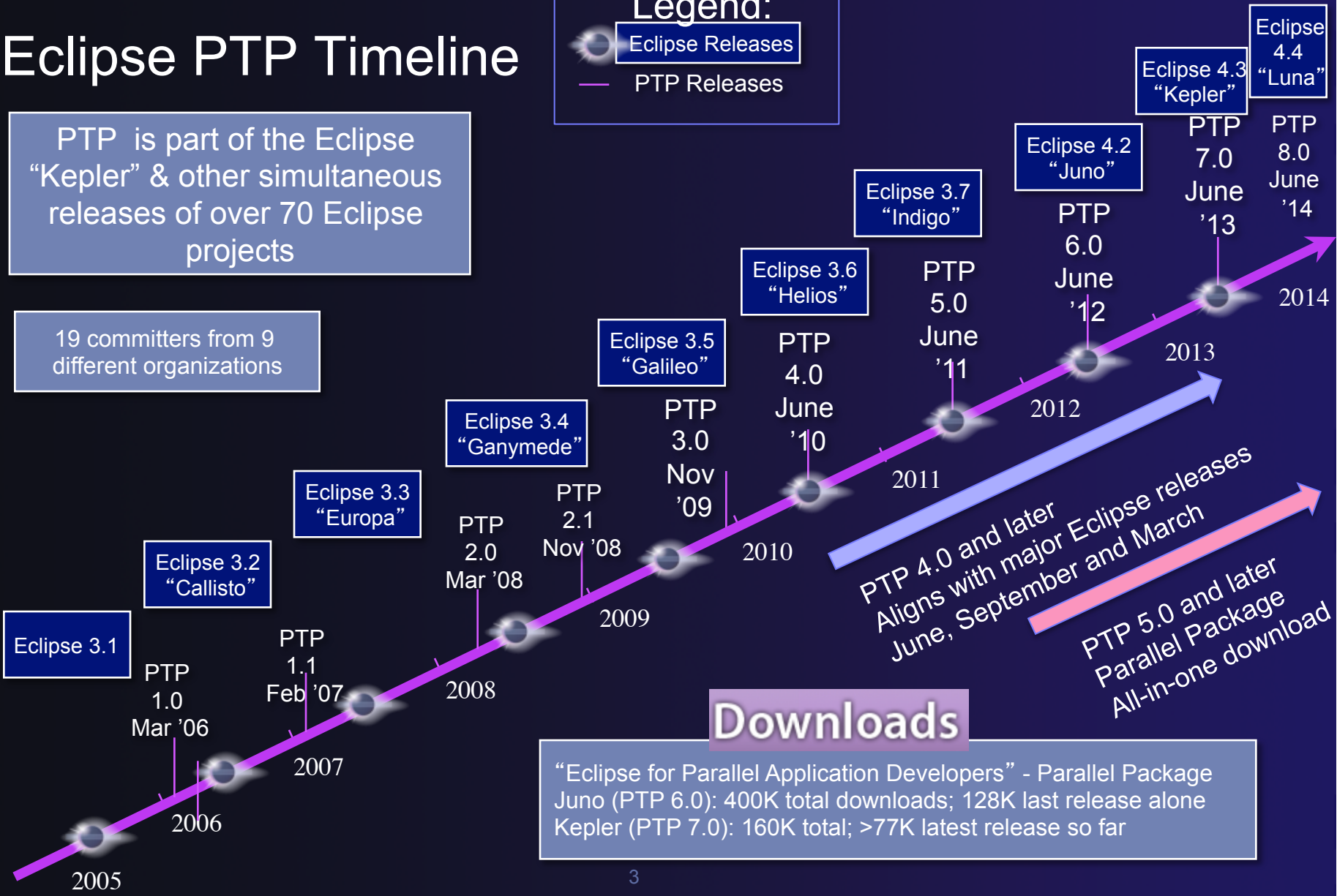
Parallel Debugging

# Eclipse PTP Timeline

**Legend:**  
 Eclipse Releases  
 PTP Releases

PTP is part of the Eclipse "Kepler" & other simultaneous releases of over 70 Eclipse projects

19 committers from 9 different organizations



PTP 4.0 and later Aligns with major Eclipse releases June, September and March

PTP 5.0 and later Parallel Package All-in-one download

## Downloads

"Eclipse for Parallel Application Developers" - Parallel Package  
 Juno (PTP 6.0): 400K total downloads; 128K last release alone  
 Kepler (PTP 7.0): 160K total; >77K latest release so far

## Download PTP

- ★ Eclipse is available in a number of different packages for different kinds of development
  - ★ <http://eclipse.org/downloads>
- ★ For PTP, we recommend the all-in-one download:
  - ★ Eclipse for Parallel Application Developers



**Eclipse for Parallel Application Developers**, 200 MB

Downloaded 88,437 Times

[Details](#)

We often call this the “Parallel Package”

- ★ Update the package to current PTP release via instructions at <http://www.eclipse.org/ptp/downloads.php>

# Highlight of PTP Features

# parallel tools platform

## Workbench

The screenshot shows the Eclipse IDE interface with several components labeled:

- Build**: Points to the Build button in the toolbar.
- Current Perspective name: C/C++**: Points to the perspective name in the top toolbar.
- Debug**: Points to the Debug button in the toolbar.
- Run**: Points to the Run button in the toolbar.
- Switch Perspective**: Points to the Switch Perspective button in the toolbar.
- Rightmouse for Project Properties**: Points to the right-click context menu in the Project Explorer.
- Source Code Repository**: Points to the CVS icon in the Project Explorer.
- Version # in SCR**: Points to the version number in the Project Explorer.
- > Indicates change from SCR**: Points to the change indicator in the Project Explorer.
- Editor**: Points to the main code editor window.
- Outline View**: Points to the Outline view on the right side.
- View**: Points to the Console view at the bottom.

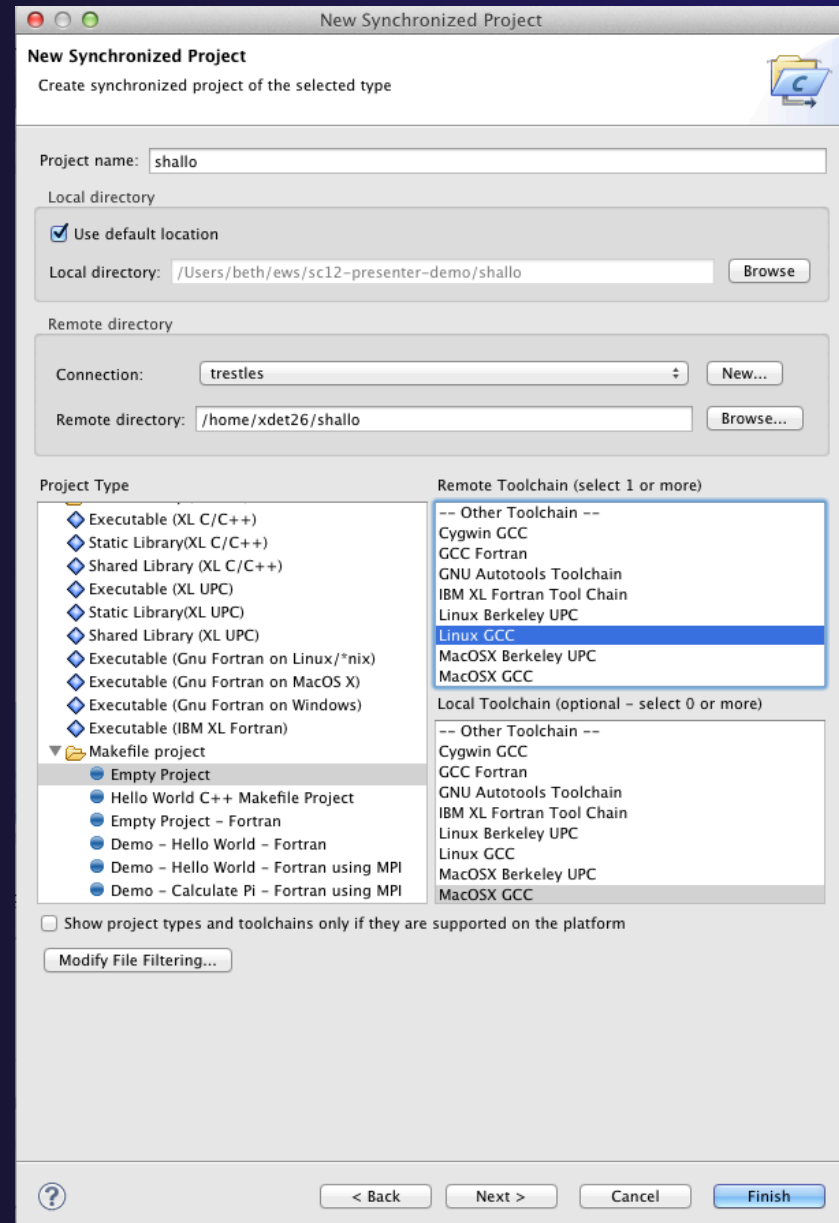
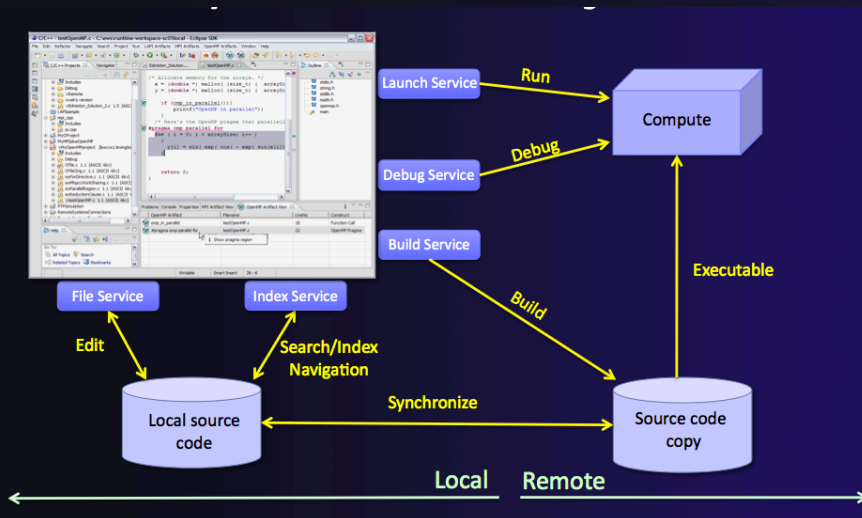
**Preferences:** Menu: Window>Preferences  
Mac: Eclipse>Preferences

Problems view: Build errors etc.

Console view: Build output; Run output

# Synchronized Projects

- ★ Source is mirrored on local and remote host
  - ★ Responsive editing and interactions on local
  - ★ Build and run on remote
  - ★ Generic: Not just C/C++/Fortran

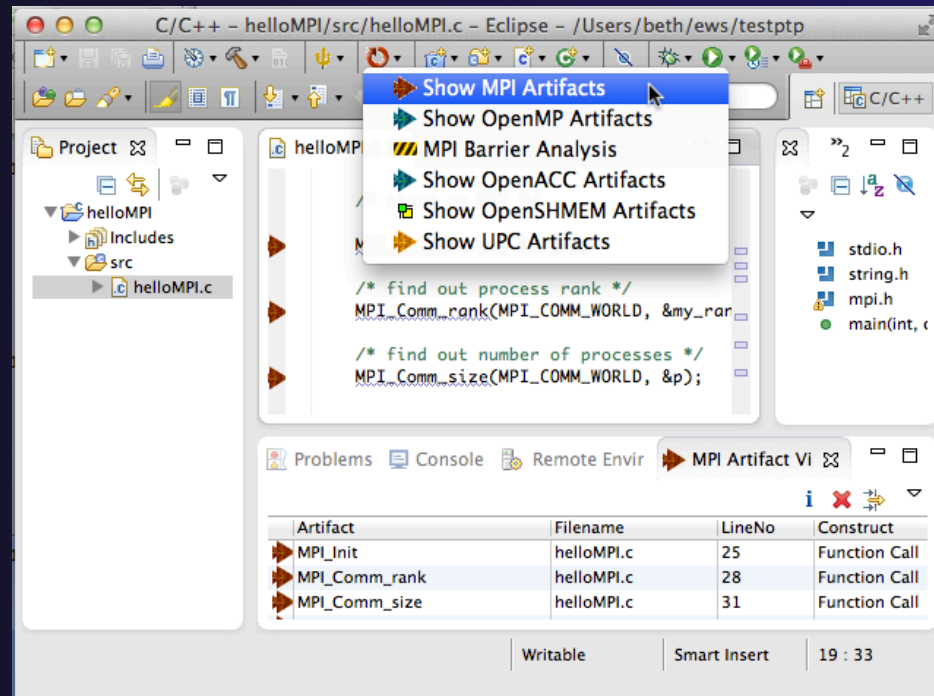




# MPI, OpenMP, UPC, .... Support

Support for a variety of languages/APIs/Libraries for parallel development

- ★ MPI
- ★ OpenMP
- ★ OpenSHMEM
- ★ OpenACC
- ★ UPC



# Editor Features for Parallel Development

```

float vold_start[m];
int proc_cnt;
int tid;
MPI_Datatype * res_type;

MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);
MPI_Comm_rank(MPI_COMM_WORLD, &tid);

MPI_B

```

- MPI\_Barrier(MPI\_Comm) int
- MPI\_Bcast(void\*, int, MPI\_Datatype, int, MPI\_
- MPI\_Bsend(void\*, int, MPI\_Datatype, int, int,
- MPI\_Bsend\_init(void\*, int, MPI\_Datatype, int,
- MPI\_Buffer\_attach(void\*, int) int
- MPI\_Buffer\_detach(void\*, int\*) int
- MPI\_Barrier(MPI\_Comm comm) : int
- MPI\_Bcast(void \* buffer, int count, MPI\_Dataty
- MPI\_Bsend(void \* buf, int count, MPI\_Datatype
- MPI\_Bsend\_init(void \* buf, int count, MPI Data

- ★ Code completion will show all the possible MPI keyword completions
- ★ Enter the start of a keyword then press <ctrl-space>
- ★ Hover over MPI API
- ★ Displays the function prototype and a description

```

float vold_start[m];
int proc_cnt;
int tid;
MPI_Datatype * res_type;

MPI_Init(&argc, &argv);
MPI_Comm_size(MPI_COMM_WORLD, &proc_cnt);
MPI_Comm_rank(MPI_COMM_WORLD, &tid);

return 1;
}

if ( ( n % (proc_cnt - 1) != 0 )
{
    if ( tid == 0 )
        fprintf(stderr, "(number of processes - 1) must be a mu

```

**Name:** MPI\_Comm\_rank  
**Prototype:** int MPI\_Comm\_rank(MPI\_Comm, int \*)  
**Description:** Returns the rank of the local task in the group associated with a communicator.  
Press 'F2' for focus

- ★ MPI Barrier Analysis
- ★ Code templates for common idioms
- ★ Similar tools for OpenMP, UPC, OpenSHMEM, OpenACC

# Fortran

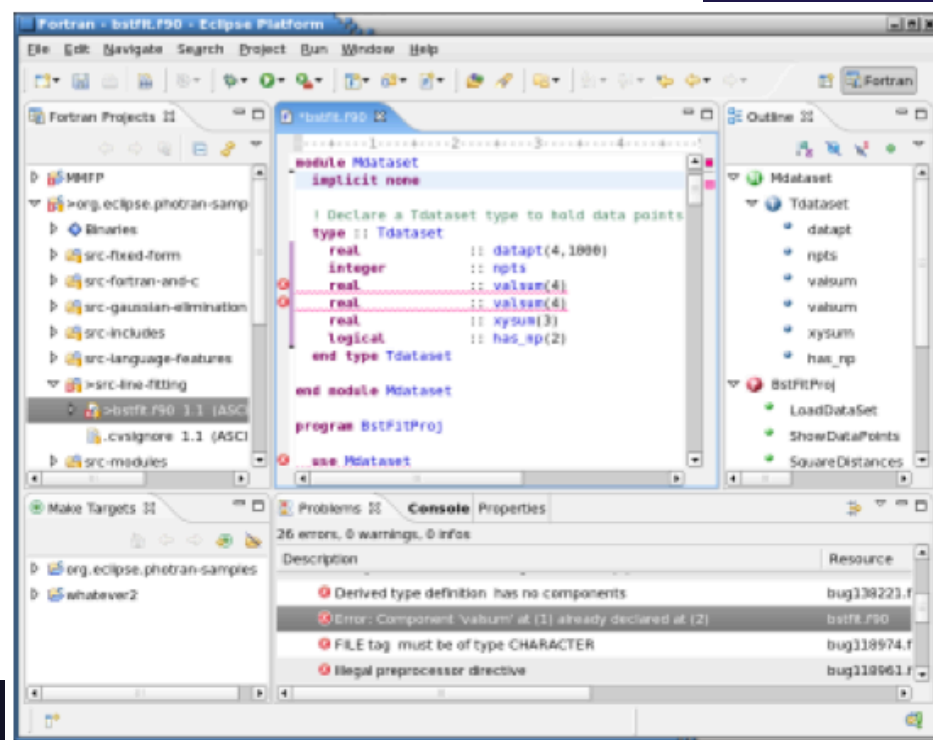
★ Photran: <http://eclipse.org/photran/>

## Photran - An Integrated Development Environment and Refactoring Tool for Fortran

Photran is an IDE and refactoring tool for Fortran based on **Eclipse** and the **CDT**. **Photran 8.0** was released with Eclipse 4.2 (Juno) on June 27, 2012.

Photran 8.0 supports Fortran 77-2008. It includes 39 refactorings (including Rename, Extract Procedure, and loop transformations), as well as the following IDE features:

- Syntax-highlighting editor
- Outline view
- Content assist\*
- Open declaration\*
- Declaration view and hover tips\*
- Fortran language-based searching\*
- Support for CVS & other VCS's\*\*
- Interactive debugger (*gdb* GUI)
- Makefile-based compilation
- Optional Makefile generation



# Refactoring Example: Fortran Loop Transformations

## ★ Unroll Loop

★ Select a loop, click **Refactor** ▶ **Do Loop** ▶ **Unroll Loop...**

```
do i = 1, 10
  print *, 10*i
end do
```



Unroll 4x

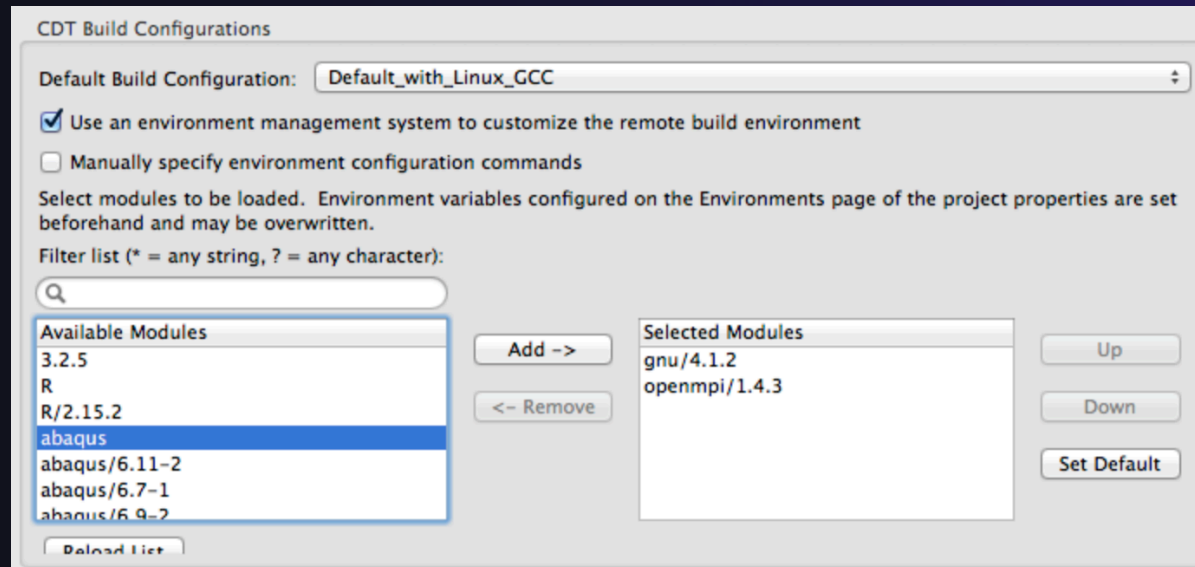
```
do i = 1, 10, 4
  print *, 10*i
  print *, 10*(i+1)
  print *, 10*(i+2)
  print *, 10*(i+3)
end do
```

| Original Source                       | Refactored Source             |
|---------------------------------------|-------------------------------|
| 68                                    | 78 end do                     |
| 69 ! Don't apply time filter on first | 79 end if                     |
| 70 if ( firststep == 0 ) then         | 80                            |
| 71 do j = jstart+1, jend+1            | 81 do j = jstart+1, jend+1    |
| 72 do i = 1, m                        | 82 LoopUpperBound = m         |
| 73 pold(i,j) = p(i,j)+alpha*(pne      | 83 do i = 1, LoopUpperBound,4 |
| 74 uold(i,j) = u(i,j)+alpha*(une      | 84 p(i,j) = pnew(i,j)         |
| 75 vold(i,j) = v(i,j)+alpha*(vne      | 85 u(i,j) = unew(i,j)         |
| 76 end do                             | 86 v(i,j) = vnew(i,j)         |
| 77 end do                             | 87 p((i+1),j) = pnew((i+1)    |
| 78 end if                             | 88 u((i+1),j) = unew((i+1)    |
| 79                                    | 89 v((i+1),j) = vnew((i+1)    |
| 80 do j = jstart+1, jend+1            | 90 p((i+2),j) = pnew((i+2)    |
| 81 do i = 1, m                        | 91 u((i+2),j) = unew((i+2)    |
| 82 p(i,j) = pnew(i,j)                 | 92 v((i+2),j) = vnew((i+2)    |
| 83 u(i,j) = unew(i,j)                 | 93 p((i+3),j) = pnew((i+3)    |
| 84 v(i,j) = vnew(i,j)                 | 94 u((i+3),j) = unew((i+3)    |
| 85 end do                             | 95 v((i+3),j) = vnew((i+3)    |
| 86 end do                             | 96 end do                     |
| 87 end subroutine                     | 97 end do                     |
| 88                                    | 98 end subroutine             |
|                                       | 99                            |
|                                       | 00                            |

# Configuring Build Modules

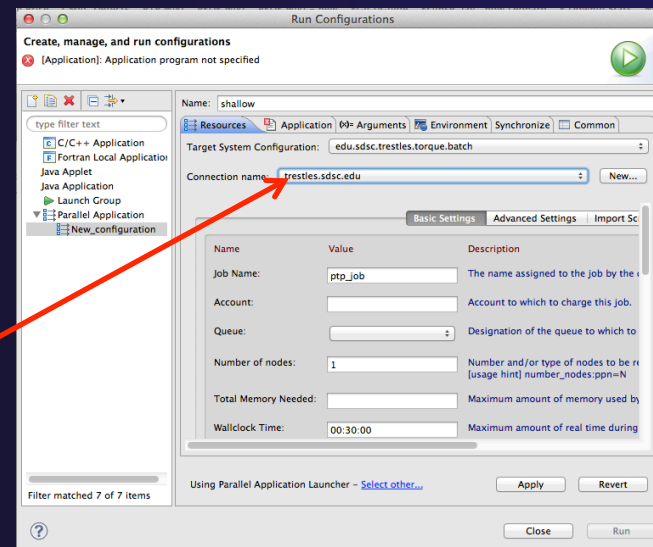
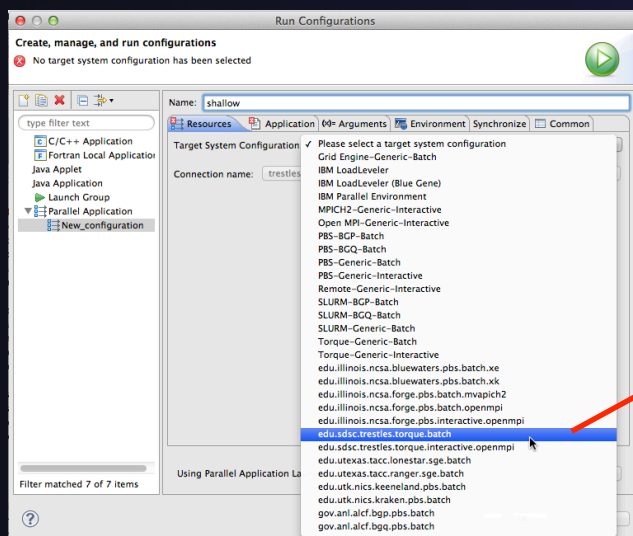
## Environment Management

- ✦ If the remote system has Modules installed, a custom set of modules can be configured for building C/C++ projects
- ✦ On a per-build configuration, run & debug configurations
- ✦ Modules can be ordered



# Launching

- ✦ Flexible Target System Configurations
- ✦ UI solicits input depending on remote target



- ✦ PTP Kepler added support for filtering, IBM Platform LSF & MPI,

# Monitoring (incl. stand-alone "Sysmon")

★ System view

★ Jobs running on system

★ Active jobs

★ Inactive jobs

★ Messages

★ Console

The screenshot shows the Eclipse IDE's System Monitoring window. On the left, there are several panels: 'Monitors' showing the connection 'trestles.sdsc.edu' (TORQUE Resource Manager); 'Active Jobs' with a table of running jobs; 'Inactive Jobs' with a table of pending jobs; 'Messages' showing a terminated process; and 'Console' with a scrollable log. On the right, a large grid of colored squares represents the system's resource usage across multiple nodes. Orange arrows point from the text labels on the left to their corresponding panels in the screenshot.

| step     | owner   | queue  | wall  | queued  | dispatch | total |
|----------|---------|--------|-------|---------|----------|-------|
| 10553... | jmondal | normal | 64800 | 2012... | 201      | 201   |
| 10553... | jmondal | normal | 64800 | 2012... | 201      | 201   |
| 10553... | jmondal | normal | 64800 | 2012... | 201      | 201   |
| 10553... | jmondal | normal | 64800 | 2012... | 201      | 201   |

| step    | owner    | queue  | wall | queued | dispatch | total |
|---------|----------|--------|------|--------|----------|-------|
| 1056... | tibbitts | shared | 1800 | 201... | 201...   | 5     |
| 1056... | tibbitts | shared | ?    | ?      | ?        | ?     |
| 1056... | tibbitts | shared | ?    | ?      | ?        | ?     |

Scroll to see more

# Parallel Debugger

- ★ **Parallel Debug view** shows job and processes being debugged
- ★ **Debug view** shows threads and call stack for individual processes
- ★ **Source view** shows a **current line marker** for all processes
- ★ Automatic deployment of sdm on target machine

The screenshot displays the Eclipse Parallel Debugger interface. The top toolbar includes icons for Parallel Debug, Parallel Run, and Remote C/C++. The main window is divided into several panes:

- Parallel Debug:** Shows a tree view with 'job0' and 'Process 0'.
- Debug:** Shows a tree view with 'shallow [Parallel Application]', 'Process 0', and 'Thread [1] (Suspended)'. The call stack shows '1 main() main.c:38 4032ca'.
- Variables:** A table showing variable names and their values:
 

| Name | Value        |
|------|--------------|
| argc | 1            |
| argv | 7fffffff658  |
| pi   | 5.957805E-39 |
| p    | [0 - 18]     |
| u    | [0 - 18]     |
| v    | [0 - 18]     |
| psi  | [0 - 18]     |
- Source:** Shows the source code for 'main.c' with a current line marker at line 38:
 

```

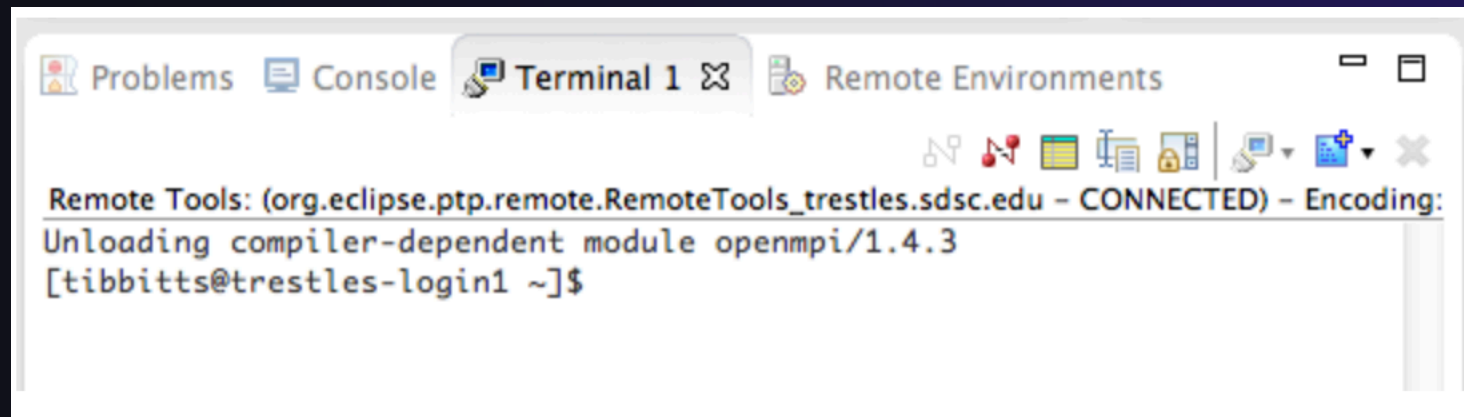
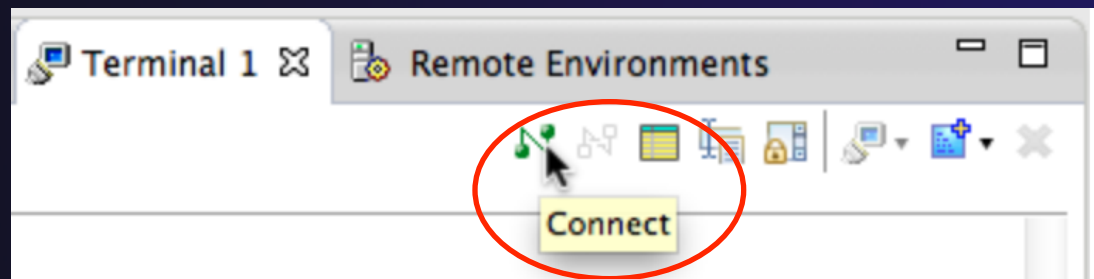
32 MPI_Datatype * setup_res();
33
34 main (argc, argv)
35     int argc;
36     char * argv[];
37 {
38     float pi=4.*((float)atan((double)1.));
39     float p[n][m]; /* Pressure (or free surface height) */
40     float u[n][m]; /* Zonal wind */
41     float v[n][m]; /* Meridional wind */
42     float psi[n][m]; /* Velocity streamfunction */
43     float pold[n][m];
44     float uold[n][m];
45     float vold[n][m];
46     float h[n][m];
47     float z[n][m];
48     float dummy1[m];
49     float dummy2[n][m];
50     float tpi=pi+pi;
51     float di=tpi/(float)m;
52     float dj=tpi/(float)n;
      
```
- Outline:** Shows a list of files and functions, including 'math.h', 'mpi.h', 'stdio.h', 'decs.h', 'worker(): void', 'setup\_res(): MPI\_Datatype\*', 'main(int, char\*[])', 'update\_global\_ds(MPI\_Datatype)', and 'update\_global\_ds(MPI\_Datatype\*'.

The bottom status bar shows 'Open Mpi Job'.



# Built-in Terminal View

Issue commands  
on remote target  
using existing  
connection  
information



# Tutorial, VM

- ✦ Tutorial slides

- ✦ Most recent tutorial:

- <http://wiki.eclipse.org/PTP/tutorials/XSEDE2013>

# Commercial Products that include PTP

# Commercial Products that include PTP

## ★ IBM Parallel Environment Developer Edition

- ★ Eclipse, PTP
- ★ Hardware Performance Counters
- ★ MPI/OpenMP tracing/profiling
- ★ Assistance tools for IBM PAMI
- ★ XLC Compiler Transformation Report viewer



## ★ NVIDIA Nsight development environment

- ★ Includes portions of PTP



# Online Information

- ★ Information about PTP
  - ★ PTP online help
    - ★ <http://help.eclipse.org>
  - ★ Main web site for downloads, documentation, etc.
    - ★ <http://eclipse.org/ptp>
  - ★ Wiki for designs, planning, meetings, etc.
    - ★ <http://wiki.eclipse.org/PTP>
    - ★ Newest features: “New and Noteworthy” link
- ★ Information about Photran
  - ★ Main web site for downloads, documentation, etc.
    - ★ <http://eclipse.org/photran>

# Mailing Lists

- ★ User Mailing Lists
  - ★ PTP
    - ★ <http://dev.eclipse.org/mailman/listinfo/ptp-user>
  - ★ Photran
    - ★ <http://dev.eclipse.org/mailman/listinfo/photran>
  - ★ Major announcements (new releases, etc.) - low volume
    - ★ <http://dev.eclipse.org/mailman/listinfo/ptp-announce>
  
- ★ Developer Mailing Lists
  - ★ Developer discussions - higher volume
    - ★ <http://dev.eclipse.org/mailman/listinfo/ptp-dev>

# Getting Involved

- ✦ See <http://eclipse.org/ptp>
- ✦ Read the developer documentation on the wiki
  - ✦ <http://wiki.eclipse.org/PTP>
- ✦ Join the mailing lists
- ✦ Attend the monthly Developer Meetings
  - ✦ Conf Call Monthly: Second Tuesday, 1:00 pm ET
  - ✦ Details on the PTP wiki
- ✦ Attend the monthly User Meetings
  - ✦ Teleconf Monthly: 4<sup>th</sup> Wednesday, 1:00 pm ET
    - ✦ **Move Nov-Dec meeting to early Dec?**
  - ✦ Details on the PTP wiki
- ✦ Annual PTP User-Developer meeting