



parallel tools platform

<http://eclipse.org/ptp>

Using the Eclipse Parallel Tools Platform
to Assist Earth Science Model
Development and Optimization on High
Performance Computers

Jay Alameda

National Center for Supercomputing Applications

UCAR Software Engineering Assembly

22 February 2012

Acknowledgements

- ✦ Portions of this material are supported by or based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under its Agreement No. HR0011-07-9-0002, the United States Department of Energy under Contract No. DE-FG02-06ER25752, the Blue Waters sustained petascale computing project, which is supported by the National Science Foundation under award number OCI 07-25070, and the SI2-SSI Productive and Accessible Development Workbench for HPC Applications, which is supported by the National Science Foundation under award number OCI 1047956
- ✦ The SI2-SSI team is lead by Jay Alameda (NCSA), Greg Watson (IBM), Steven Brandt (LSU), Marc Snir (U Illinois), and Allen Malony (U Oregon). Team members and senior personnel include Beth Tibbitts (IBM), Ralph Johnson (U Illinois), Albert Rossi (NCSA), Rick Kufirin (NCSA), Sameer Shende (U Oregon), Wyatt Spear (U Oregon), Bety Rodriguez-Milla (LSU), Brian Jewett (U Illinois), Galen Arnold (NCSA), and Rui Liu (NCSA)

Outline

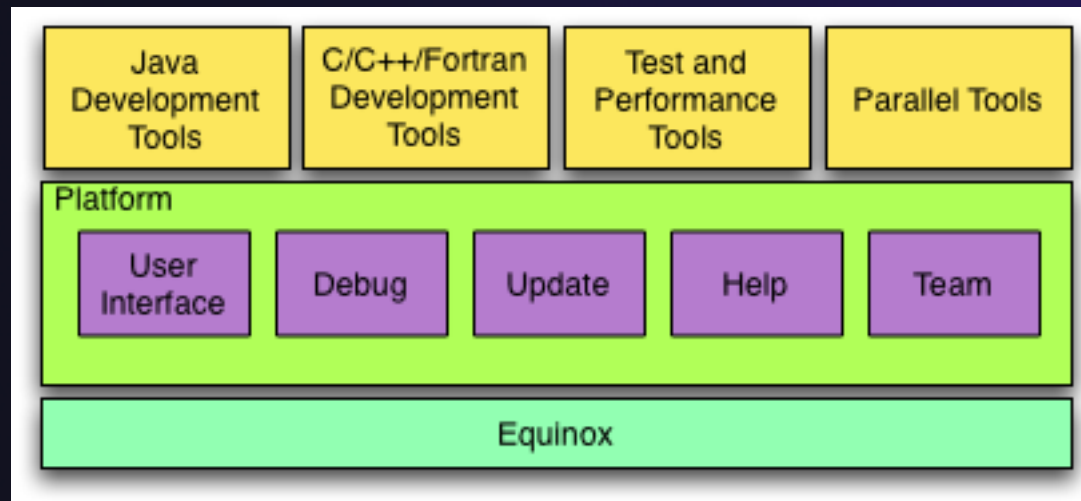
- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

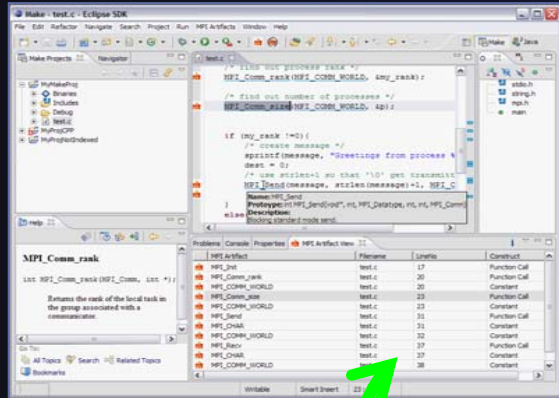
What is Eclipse?

- ✦ A vendor-neutral open-source workbench for multi-language development
- ✦ A extensible platform for tool integration
- ✦ Plug-in based framework to create, integrate and utilize software tools

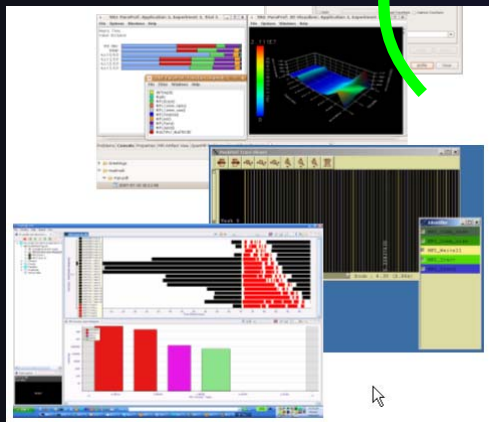
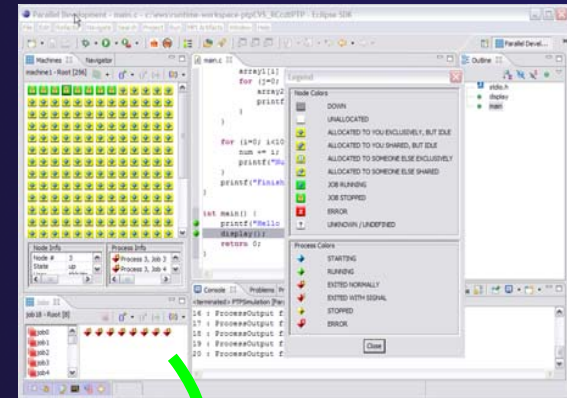


Eclipse Parallel Tools Platform (PTP)

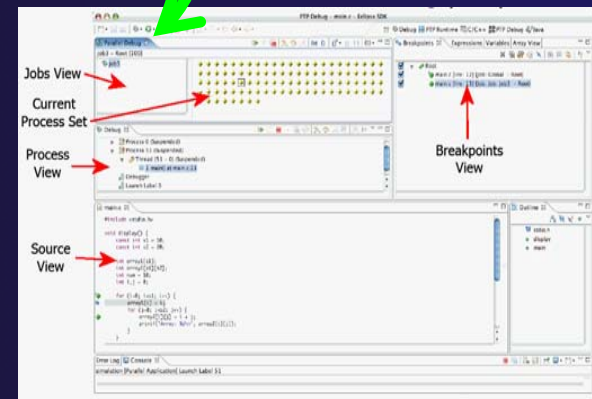
Coding & Analysis



Launching & Monitoring



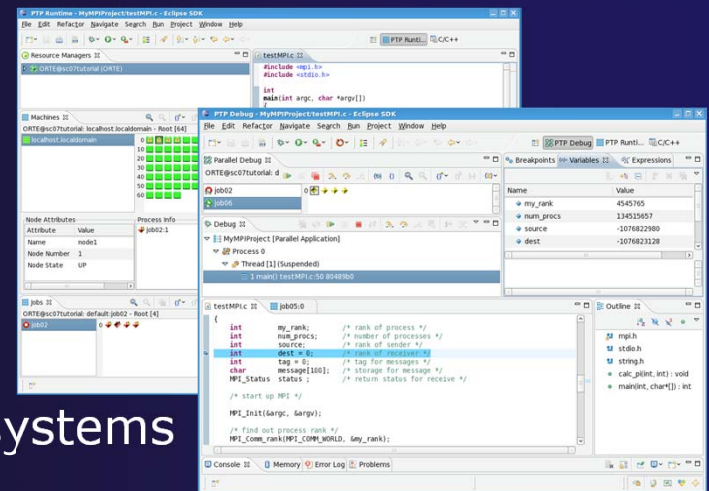
Performance Tuning



Debugging

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, etc.)
 - ★ Support for the integration of parallel tools
 - ★ An environment that simplifies the end-user interaction with parallel systems
- ★ <http://www.eclipse.org/ptp>



Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Why WHPC?

- ★ Stable, portable platform for tool development
 - ★ Focus on tool functionality, manage rapid evolution of HPC platforms
 - ★ Encourage consistent tool look and feel
 - ★ Support for HPC application development practices
 - ★ Edit, build, test, debug, maintain, for maximum developer productivity
 - ★ Remote development, batch execution mandatory
 - ★ Track, store, search, browse code artifact provenance
 - ★ Share tool functionality through an integration framework
 - ★ Maintain tool identity
 - ★ Provides for independent tool development pathways and funding

Why Parallel Tools Platform?

- ✦ High potential to meet needs of a WHPC.
- ✦ Target next generation of HPC developers growing up with IDEs (Eclipse, Visual Studio, ...)
- ✦ For PTP to become a WHPC need to:
 - ✦ Cultivate community of users
 - ✦ Make substantial improvements to PTP around two themes:
 - ✦ Improving usability
 - ✦ Improving productivity

Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Requirements and Impact

- ★ Application-centric approach
 - ★ Use real application codes, with PTP, on production computational resources
 - ★ Identify specific goals to accomplish with each application
 - ★ Use Eclipse PTP to accomplish the goals
 - ★ Identify shortcomings in Eclipse PTP that need to be rectified for Eclipse PTP to be effective with that application workplan
 - ★ This is part of our project team's responsibility
 - ★ Work with application community and learn from their experience with Eclipse PTP

Requirements and Impact (2)

- ★ Application-centric approach
 - ★ Work with application community and learn from their experience with Eclipse PTP
 - ★ Bridge to TeraGrid and (now) XSEDE Advanced User Support
 - ★ Work with targeted organizations to assist with adoption of PTP
 - ★ Monthly user calls
 - ★ Annual user group meeting
 - ★ Hands on tutorials
 - ★ Conference Birds of a Feather

Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Improvements

- ✦ Work within Eclipse release cycle
 - ✦ Major (API-breaking) improvements with coordinated June release
 - ✦ Minor enhancements and bug-fixes with two coordinated service releases in September and February
 - ✦ We are working towards a new major release now (Eclipse 4.2/Juno, released June 2012)
- ✦ Foci of improvements
 - ✦ Improve usability
 - ✦ Improve productivity

Improve Usability

- ✦ Remote support and scalability enhancements
 - ✦ Broaden support of remote capabilities to full PTP
 - ✦ Provide for easy platform configuration management
 - ✦ Provide additional remote features
 - ✦ Automatic remote service deployment
 - ✦ Multiple authentication mechanism
 - ✦ Support wide range of resource managers
 - ✦ Full remote debug support

Improve Usability

- ✦ Integration with other tools
 - ✦ Improve External Tools Framework (ETFw)
 - ✦ Full remote support
 - ✦ Integration of tool output with Eclipse views
- ✦ Improve and broaden parallel paradigm support
 - ✦ Driven by user needs and feedback

Improve Productivity

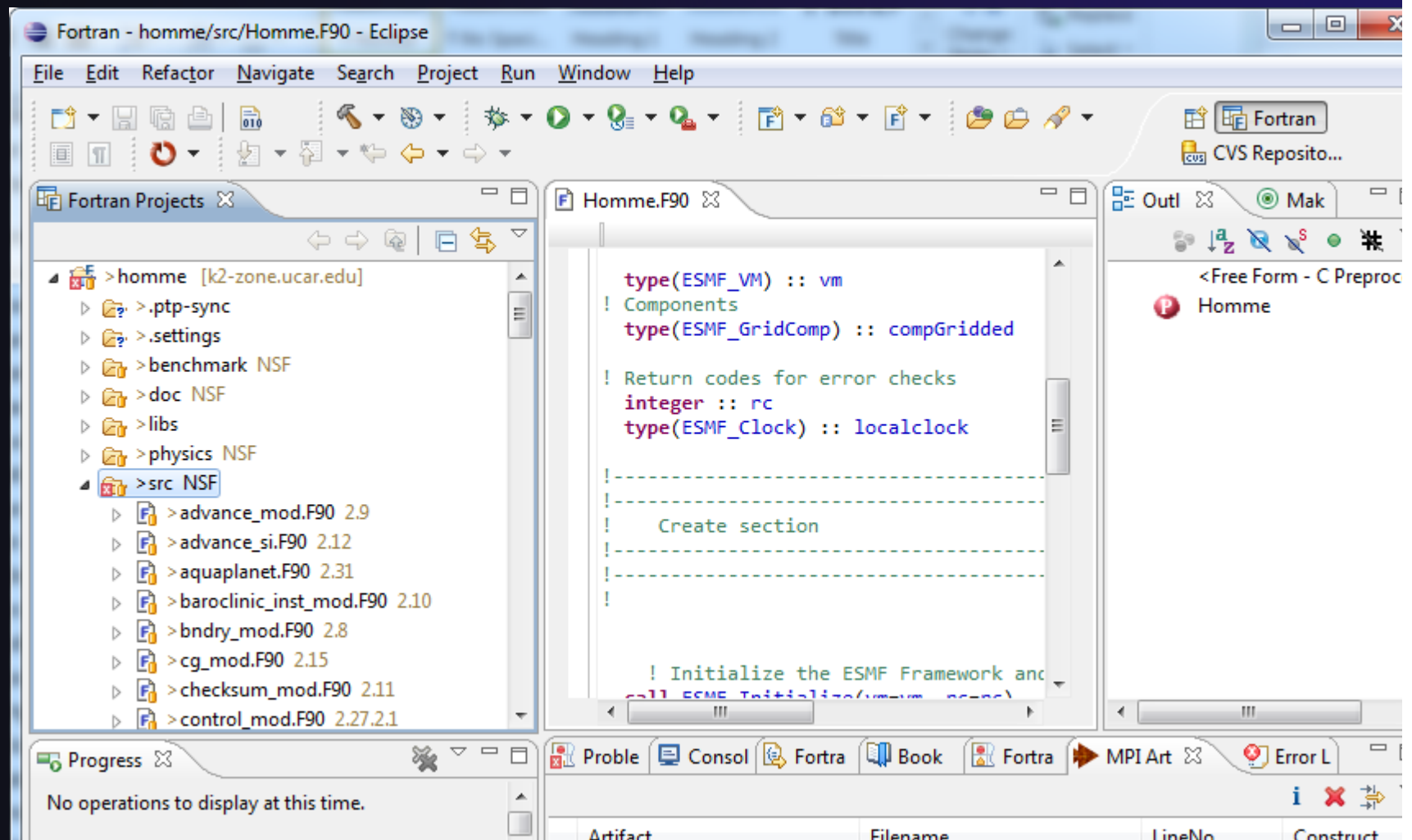
- ✦ Provide support for performance driven refactoring
- ✦ Track source and executable code provenance

Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Software Engineering

★ Code Visibility



The screenshot displays the Eclipse IDE interface for a Fortran project named 'Homme.F90'. The main window shows the source code for 'Homme.F90', which includes several Fortran declarations and comments:

```
type(ESMF_VM) :: vm
! Components
type(ESMF_GridComp) :: compGridded

! Return codes for error checks
integer :: rc
type(ESMF_Clock) :: localclock

!-----
! Create section
!-----
!

! Initialize the ESMF Framework and
call ESMF_initialize(umvm, rc=rc)
```

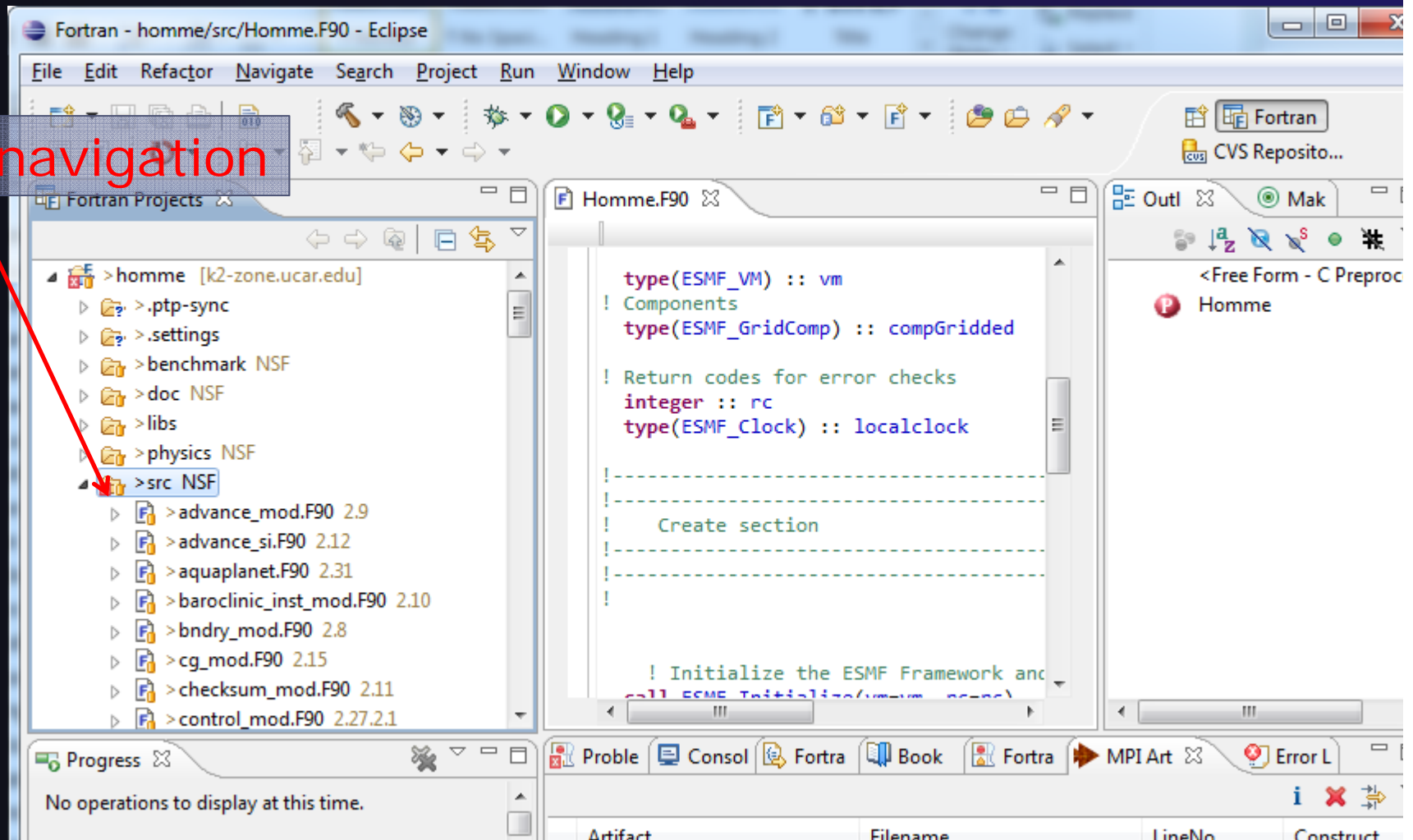
The left-hand side of the IDE shows the 'Fortran Projects' view, displaying a tree structure of the project 'homme' located at 'k2-zone.ucar.edu'. The tree includes subdirectories like '.ptp-sync', '.settings', 'benchmark NSF', 'doc NSF', 'libs', 'physics NSF', and 'src NSF'. The 'src NSF' directory is expanded, showing a list of Fortran source files with their respective version numbers, such as 'advance_mod.F90 2.9', 'advance_si.F90 2.12', 'aquaplanet.F90 2.31', 'baroclinic_inst_mod.F90 2.10', 'bndry_mod.F90 2.8', 'cg_mod.F90 2.15', 'checksum_mod.F90 2.11', and 'control_mod.F90 2.27.2.1'.

The bottom of the IDE features a 'Progress' view showing 'No operations to display at this time.' and a 'Proble' view showing a list of artifacts, including 'Artifact', 'Filename', 'LineNo', and 'Construct'.

Software Engineering

★ Code Visibility

Code navigation



Software Engineering

★ Code Visibility

The screenshot shows the Eclipse IDE interface for a Fortran project. The left-hand side contains a 'Fortran Projects' tree view showing a directory structure under 'homme [k2-zone.ucar.edu]'. A red arrow points from the 'src NSF' folder in this tree to the main editor window. The editor window displays Fortran code with several declarations and comments. A red arrow points from a text box at the bottom to the code. The text box contains the text: 'Syntax-aware editing (navigate to program units and declarations)'. The code in the editor includes:

```
type(ESMF_VM) :: vm
! Components
type(ESMF_GridComp) :: compGridded

! Return codes for error checks
integer :: rc
type(ESMF_Clock) :: localclock

-----
! Create section
!
-----
```

At the bottom of the IDE, there is a 'Progress' window showing 'No operations to display at this time' and an 'Error L' window.

Code navigation

Syntax-aware editing
(navigate to program
units and declarations)

Software Engineering

★ Code Visibility

The screenshot displays the Eclipse IDE interface for a Fortran project. The main editor window shows the source code for 'Homme.F90'. The code includes several declarations and a comment: `type(ESMF_VM) :: vm`, `! Components`, `type(ESMF_GridComp) :: compGridded`, `! Return codes for error checks`, `integer :: rc`, and `type(ESMF_Clock) :: localclock`. Below these is a dashed line and the text `! Create section`. The left-hand side shows a project explorer with a tree view of the file structure, including folders like `.ptp-sync`, `.settings`, `benchmark NSF`, `doc NSF`, `libs`, `physics NSF`, and `src NSF`. The `src NSF` folder is expanded, showing files like `advance_mod.F90 2.9`, `advance_si.F90 2.12`, `aquaplanet.F90 2.31`, `baroclinic_inst_mod.F90 2.10`, `bndry_mod.F90 2.8`, `cg_mod.F90 2.15`, `checksum_mod.F90 2.27.2.1`, and `control_mod.F90 2.27.2.1`. The right-hand side shows the Outline view with a tree view of the code structure, including `<Free Form - C Preproc` and `Homme`. The bottom status bar shows `Progress` and `Error L`. Three red arrows point from text boxes to specific features: one from 'Code navigation' to the project explorer, one from 'Code Outline' to the Outline view, and one from 'Syntax-aware editing (navigate to program units and declarations)' to the code editor.

Code Outline

Code navigation

Syntax-aware editing
(navigate to program units and declarations)

Software Engineering

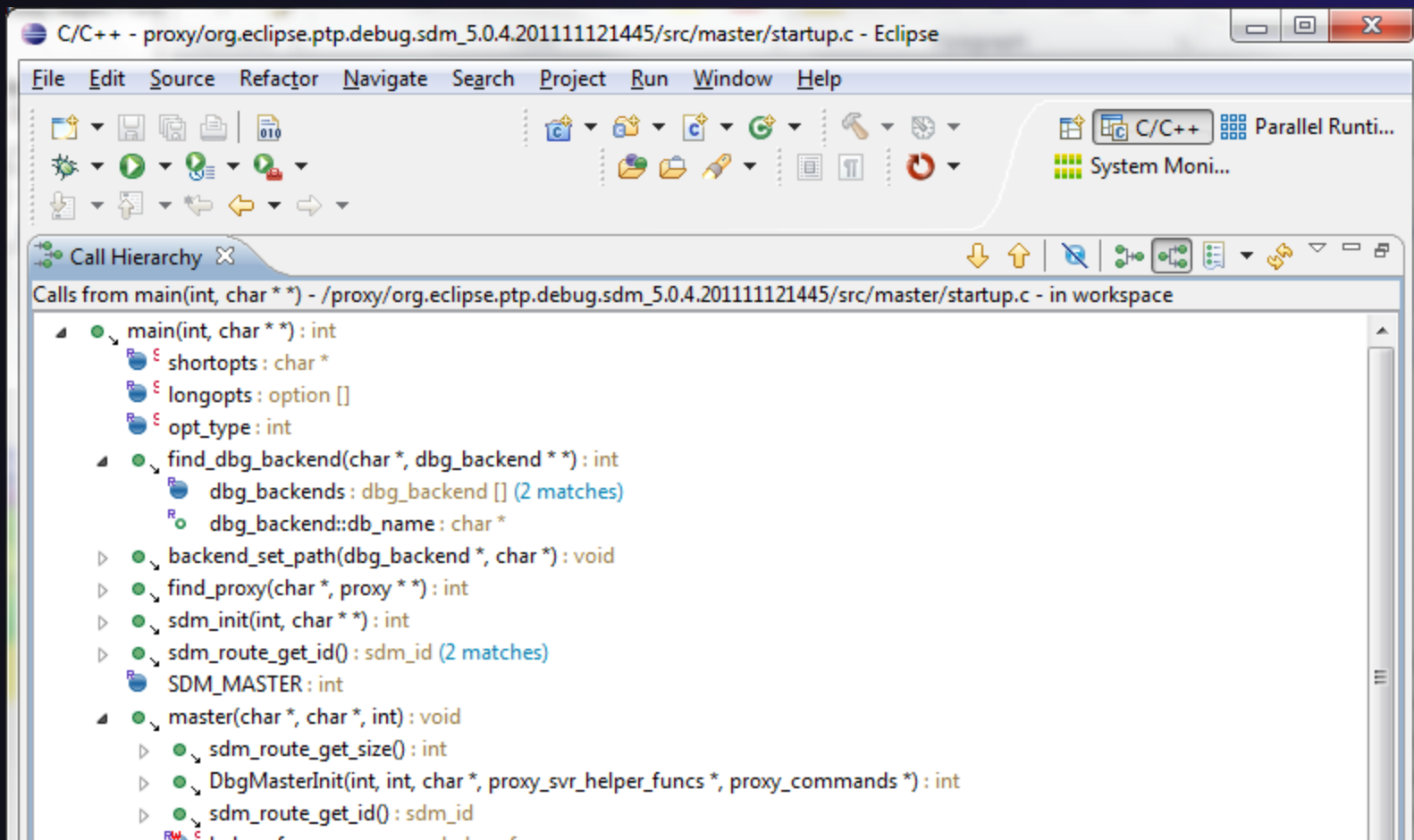
★ Code visibility: deducing call hierarchy

The screenshot shows the Eclipse IDE interface. The Project Explorer on the left displays a project named 'proxy' with several sub-projects. The main editor window shows the source code for 'startup.c', with the 'main' function signature highlighted. A red arrow points from the 'main' function signature to the 'main' function definition in the code. The right-hand side of the IDE shows the 'Call Hierarchy' view, which lists the files and functions that call the selected 'main' function. The list includes 'config.h', 'getopt.h', 'stdlib.h', 'stdarg.h', 'backend.h', 'proxy.h', 'proxy_tcp.h', 'sdm.h', and several preprocessor definitions like '# DEFAULT_BACKEND', '# DEFAULT_PROXY', and '# OPT_TYPE_DEBUGGER'. The code in the main editor is partially obscured by a semi-transparent blue box containing red text.

Would like to understand call hierarchy of this code in relation to "main()" in startup.c

Software Engineering: Call Hierarchy (C/C++)

- ★ After selecting main, right click and select <Open Call Hierarchy>



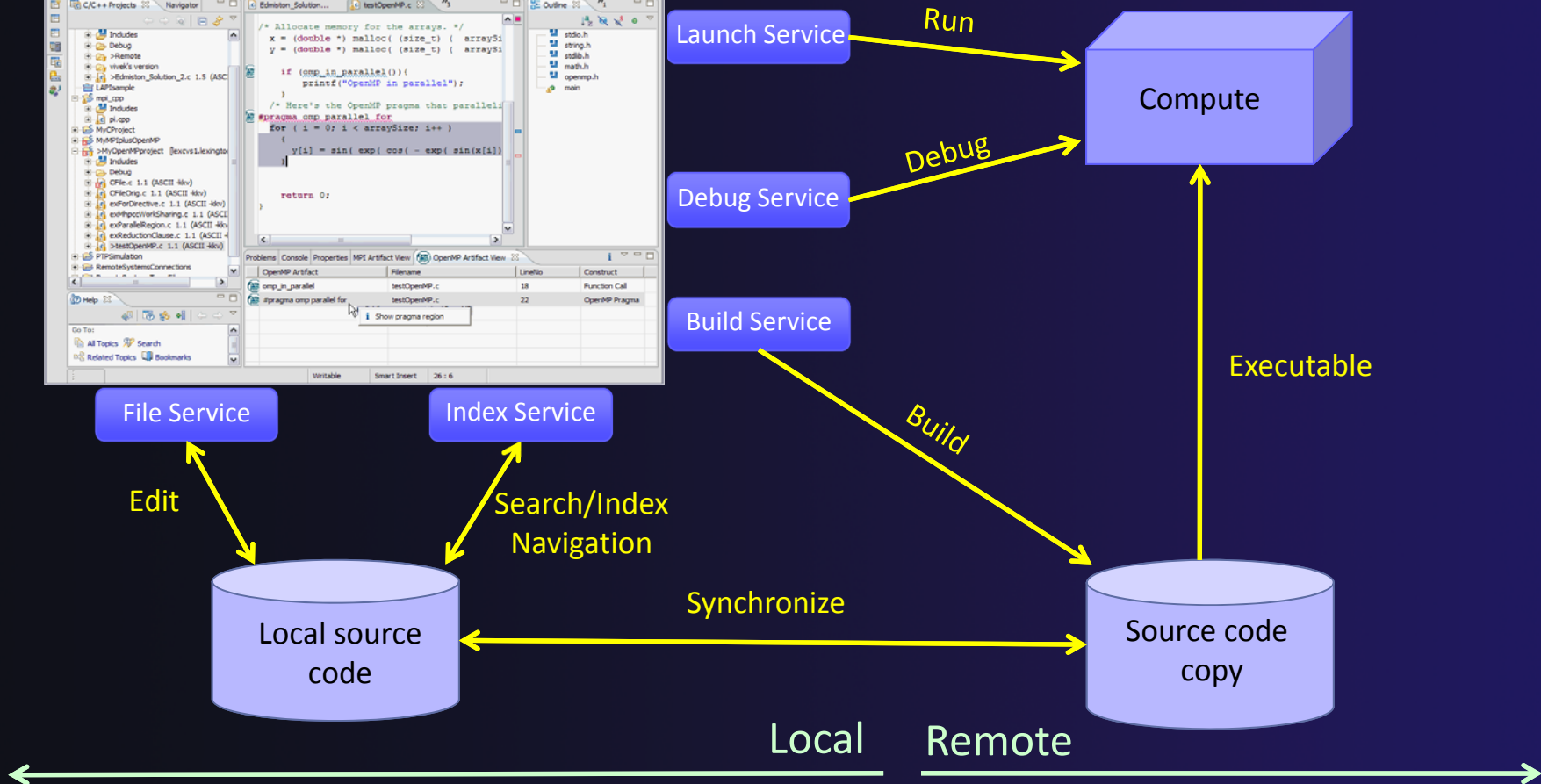
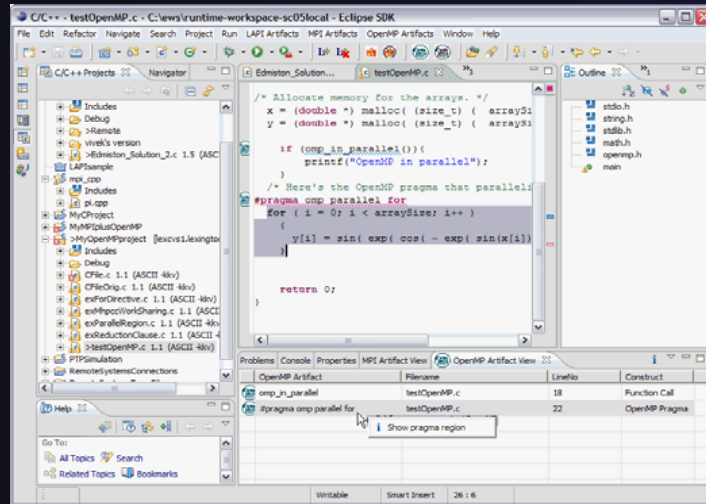
Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Multi-machine build management

- ✦ Local
 - ✦ Source is located on local machine, builds happen locally
- ✦ Synchronized
 - ✦ Source is local, then synchronized with remote machine(s)
 - ✦ Building and launching happens remotely (can also happen locally)
- ✦ Remote
 - ✦ Source is located on remote machine(s), build and launch takes place on remote machine(s)

Synchronized Projects

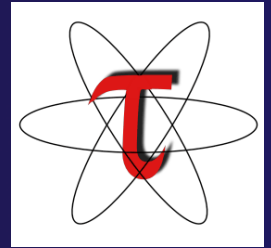


Outline

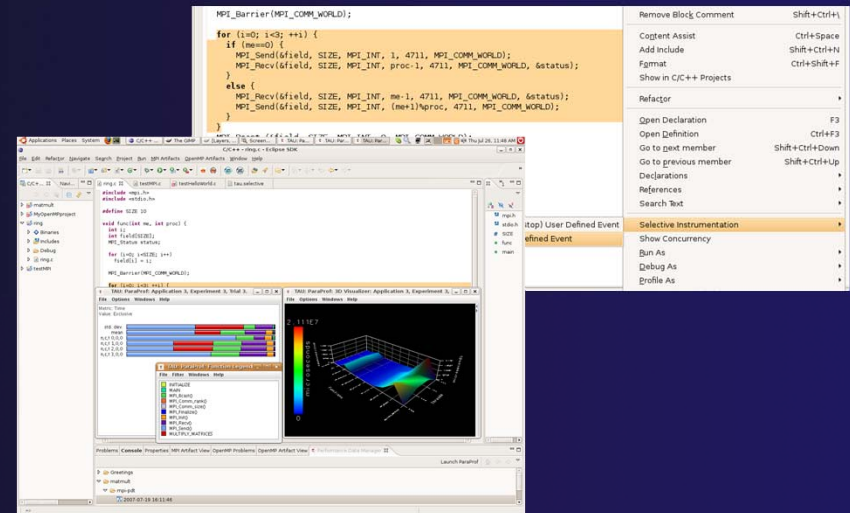
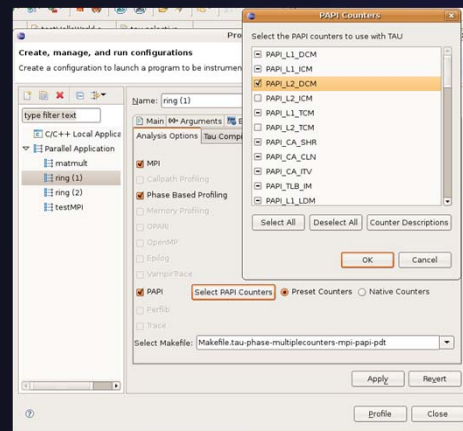
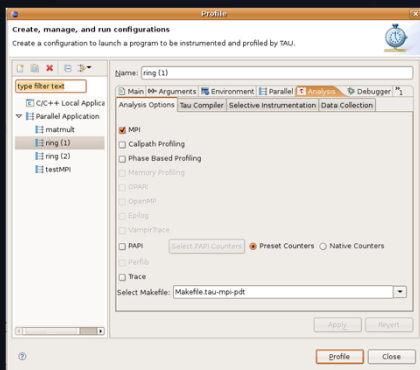
- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Performance Tuning: PTP TAU plug-ins

<http://www.cs.uoregon.edu/research/tau>



- ★ TAU (Tuning and Analysis Utilities)
- ★ First implementation of External Tools Framework (ETFw)
- ★ Eclipse plug-ins wrap TAU functions, make them available from Eclipse
- ★ Full GUI support for the TAU command line interface
- ★ Performance analysis integrated with development environment



Outline

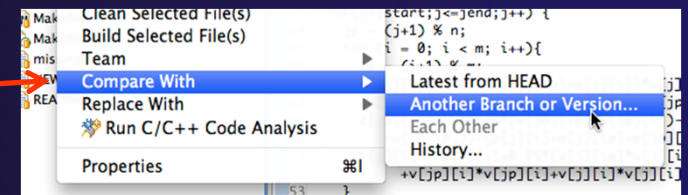
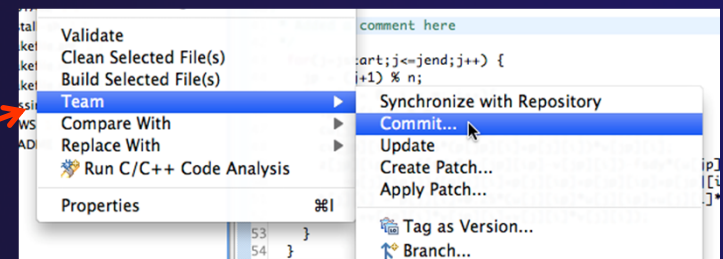
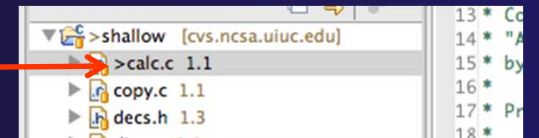
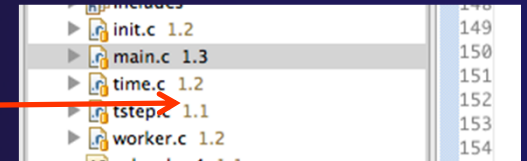
- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Source Code Control: "Team" Features

- ✦ Eclipse supports integration with multiple version control systems (VCS)
 - ✦ CVS, SVN, Git, and others
 - ✦ Collectively known as "Team" services
- ✦ Many features are common across VCS
 - ✦ Compare/merge
 - ✦ History
 - ✦ Check-in/check-out
- ✦ Some differences
 - ✦ Version numbers
 - ✦ Branching

CVS Features

- ★ Shows version numbers next to each resource
- ★ Marks resources that have changed
 - ★ Can also change color (preference option)
- ★ Context menu for Team operations
- ★ Compare to latest, another branch, or history
- ★ Synchronize whole project (or any selected resources)

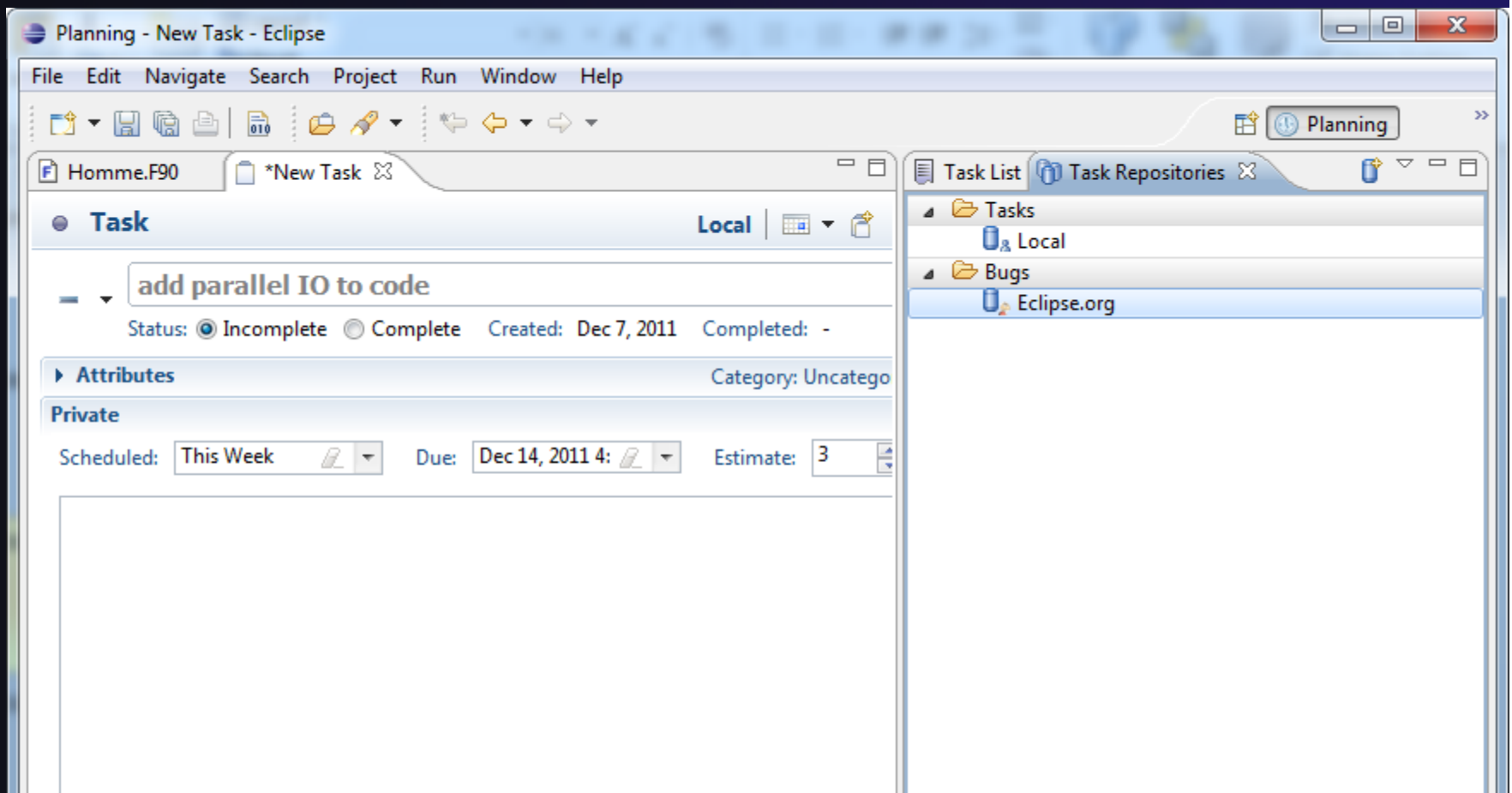


Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Issue Tracking

- ✦ Mylyn Bridge
 - ✦ Tracks tasks, links to source and bug repositories



Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Eclipse Documentation

- ★ Eclipse Help System – built in and standalone (<http://help.eclipse.org>)

The screenshot displays the Eclipse Help System interface. The window title is "Help - Eclipse". At the top, there is a search bar with the text "Search:" and a "Go" button. To the right of the search bar, it says "Scope: All topics". Below the search bar, there is a "Contents" section with a tree view of various user guides. The "Parallel Tools Platform (PTP) User Guide" is selected and expanded, showing a sub-menu with "Contents" selected. The main content area displays the "Parallel Tools Platform (PTP) User Guide" page. The page title is "Parallel Tools Platform" with a release version of "Release 5.0". The text describes the PTP's purpose and lists its functionality:

- tools for developing applications based on the Message Passing Interface (MPI) standard and other environments and APIs including OpenMP, UPC, etc.)
- the ability to launch, control and monitor the execution of parallel programs
- the ability to monitor parallel system status information
- an integrated parallel debugger
- a framework for integrating external dynamic tools

In addition, PTP provides a platform for parallel tool developers to integrate their tools within the Eclipse framework. New tools can take advantage of the user interface components and parallel services that are provided by PTP, without the need to develop and support this infrastructure on multiple platforms.

More information and downloads are available at <http://eclipse.org/ptp>.

Adapting Eclipse Documentation to Other Projects: QMCPack

★ See <http://code.google.com/p/qmcpack-doc/>

The screenshot shows a web browser window with the URL <http://code.google.com/p/qmcpack-doc/>. The page title is "qmcpack-doc" and the subtitle is "QMCPACK document". The navigation menu includes "Project Home", "Downloads", "Wiki", "Issues", and "Source". The "Project Information" section shows "Activity" with a "Low" status and "Project feeds". The "Code license" section shows "New BSD License". The "Members" section lists several contributors: jeongnim.kim, david.ce...@gmail.com, kpes...@gmail.com, jmcminis, bkcl...@gmail.com, jaronkro...@gmail.com, miguel.m...@gmail.com, lshulenb...@gmail.com, and 5 committers. The main content area is titled "Developers' and users' guides" and contains the text: "org.cmscc.qmcpack.doc is developed as an eclipse plug-in for QMCPACK help page. If all goes well, a help document with" followed by a bulleted list: "• build instructions", "• doxygen code documentation", and "• other materials on wiki". Below this, it says "can be downloaded as an eclipse plug-in." and "Licensed under [UIUC/NCSA open-source license](#)". It also says "See more on [UIUC/NCSA license](#)". At the bottom, there is a bold heading "Instructions for viewing help page in eclipse" with a red arrow pointing to it.

Outline

- ★ Overview of Eclipse and Eclipse Parallel Tools Platform (PTP)
- ★ Overview of WHPC: NSF-funded SI2-SSI project to produce a productive and accessible development workbench using Eclipse PTP
 - ★ Determining Requirements, Ensuring Impact
 - ★ Improvements to Eclipse PTP
- ★ Software Engineering Practices Enabled by Eclipse PTP
 - ★ Code visibility
 - ★ Multi-system build management
 - ★ Performance tuning
 - ★ Source code control
 - ★ Issue Tracking
 - ★ Documentation
- ★ Eclipse PTP Resources

Online Information

- ✦ Information about PTP
 - ✦ Main web site for downloads, documentation, etc.
 - ✦ <http://eclipse.org/ptp>
 - ✦ **Wiki for designs, planning, meetings, etc.**
 - ✦ <http://wiki.eclipse.org/PTP>
 - ✦ Articles and other documents
 - ✦ <http://wiki.eclipse.org/PTP/articles>

- ✦ Information about Photran
 - ✦ Main web site for downloads, documentation, etc.
 - ✦ <http://eclipse.org/photran>
 - ✦ User's manuals
 - ✦ <http://wiki.eclipse.org/PTP/photran/documentation>

Mailing Lists

★ PTP Mailing lists

- ★ Major announcements (new releases, etc.) - low volume
 - ★ <http://dev.eclipse.org/mailman/listinfo/ptp-announce>
- ★ User discussion and queries - medium volume
 - ★ <http://dev.eclipse.org/mailman/listinfo/ptp-user>
- ★ Developer discussions - high volume
 - ★ <http://dev.eclipse.org/mailman/listinfo/ptp-dev>

★ Photran Mailing lists

- ★ User discussion and queries
 - ★ <http://dev.eclipse.org/mailman/listinfo/photran>
- ★ Developer discussions –
 - ★ <http://dev.eclipse.org/mailman/listinfo/photran-dev>

Getting Involved

- ★ See <http://eclipse.org/ptp>
- ★ Read the developer documentation on the wiki
- ★ 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
 - ★ Teleconference Monthly
 - ★ Each 4th Wednesday, 2:00 pm ET
 - ★ Details on the PTP wiki

PTP will only succeed with your participation!