The Eclipse Parallel Tools Platform Project

EclipseCon 2005

Los Alamos NATIONAL LABORATORY EST. 1943

The World's Greatest Science Protecting America

LAUR-05-0574

NNSA

Parallel Development Tools
State of the Art

- Command-line compilers for Fortran and C/C++
  - Sometimes wrapped in a GUI
- Editors are vi, emacs and FRED (vintage 1960's)
- Dominant debugger is TotalView (proprietary)
  - Some use DDT (but guess what? - it's proprietary)
- Plethora of stand-alone tools
  - Patform/vendor specific, e.g. DCPI
  - Open source, e.g. TAU, HPCToolkit
  - Proprietary, e.g. Vampir, Assure

Parallel Development Tools Limitations

- Many tools are specific to only one platform or vendor
- They do not interoperate, and never will
  - No integrated UI
  - No ability to share data
  - Functionality limited to that provided by tool
- They do not scale
  - Fine for 1990's machines
  - New machines will have 10,000+ processors

Parallel Development Tools
Industry Best Practice (for everyone else)

- Integrated development environment (IDE)
  - Combines editor, compiler, debugger and other tools into a single consistent user interface
- Integrated management
  - Change control, build management, software quality policies
- Integrated testing
  - Automated unit testing, verification and validation activities
- Integrated documentation
  - On-the-fly documentation generation

Parallel Development Tools
Why Change?

- Reinforce good software engineering practices
- Strengthen auditing
- Enhance work-flow
- Increase productivity
- Improve documentation
- Reduce time-to-delivery

Result = reduced development costs

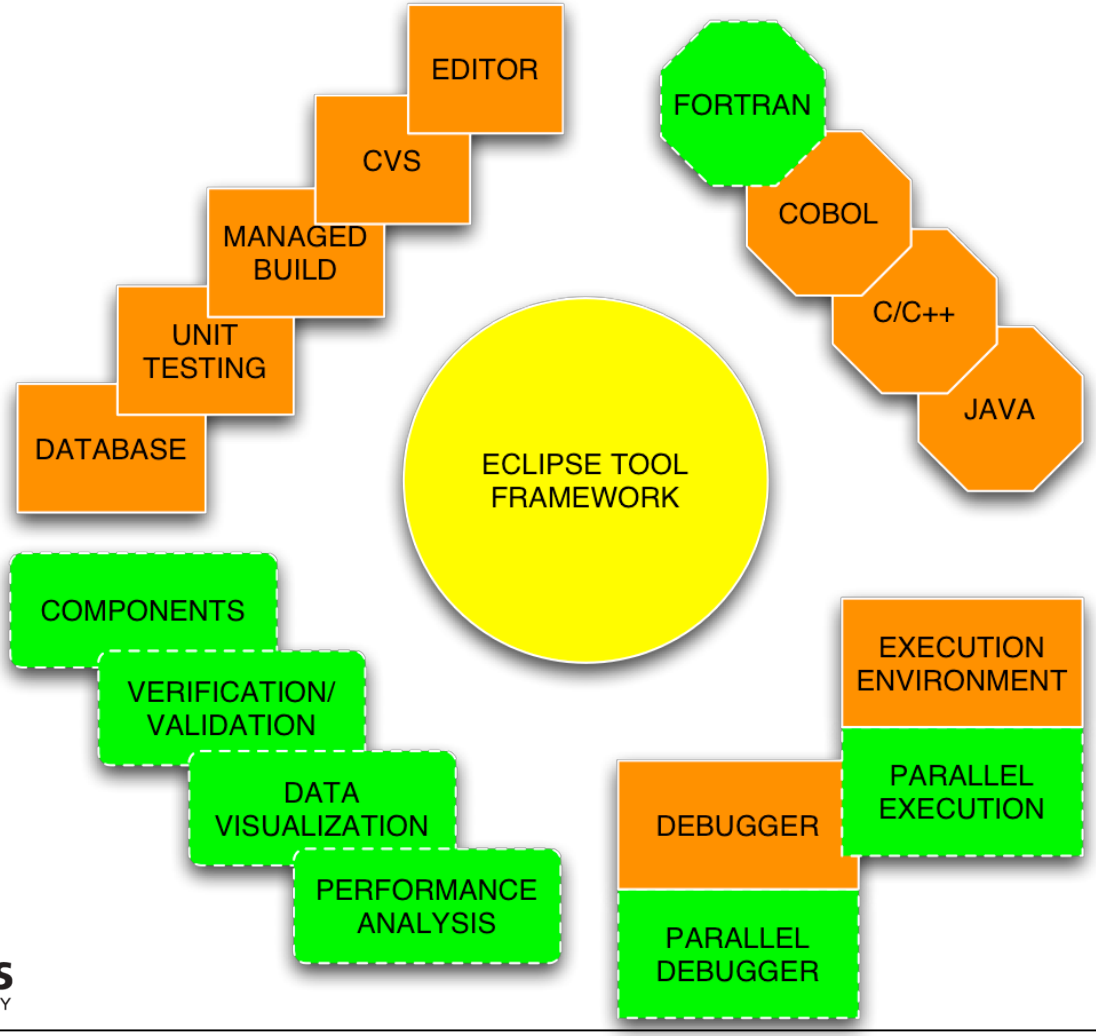
Parallel Development Tools
Barriers To Change

- Must support a range of architectures/platforms
- Must honor existing practices/processes
- Must be scalable and reliable
- Must provide core functionality
- Must be easy to adopt and support
- Must be future-proof

Parallel Tools Platform
Project Objectives

1. Extend Eclipse to support parallel development tools
2. Equip Eclipse with key tools needed to start developing parallel codes
3. Encourage existing parallel tool projects to support Eclipse
4. Exploit enhanced capabilities to develop a new generation of parallel tools

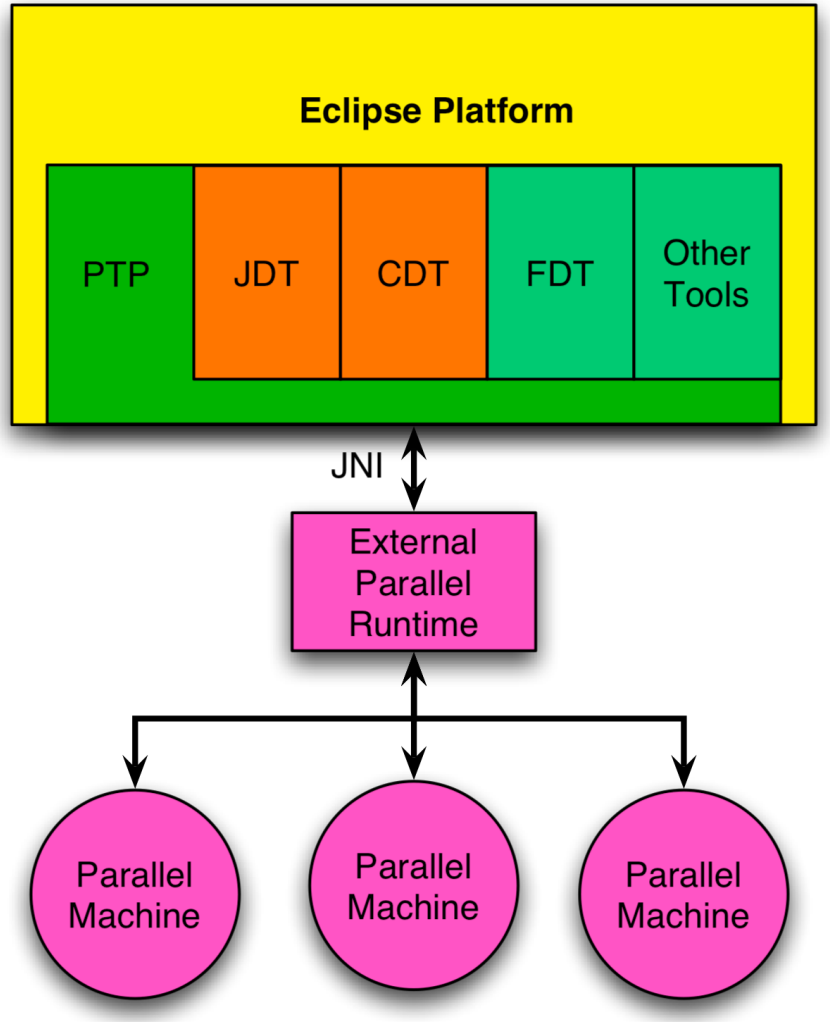
Parallel Tools Platform Components



Parallel Tools Platform Components

- Parallel Execution Environment
  - Extends existing execution environment to support parallel programs
- Parallel Debugger
  - Adds parallel debugging support to Eclipse
- Tools Integration
  - Support the integration of a variety of parallel tools, e.g. performance, verification, visualization, components
- Fortran
  - Adds Fortran support to a similar level as C/C++

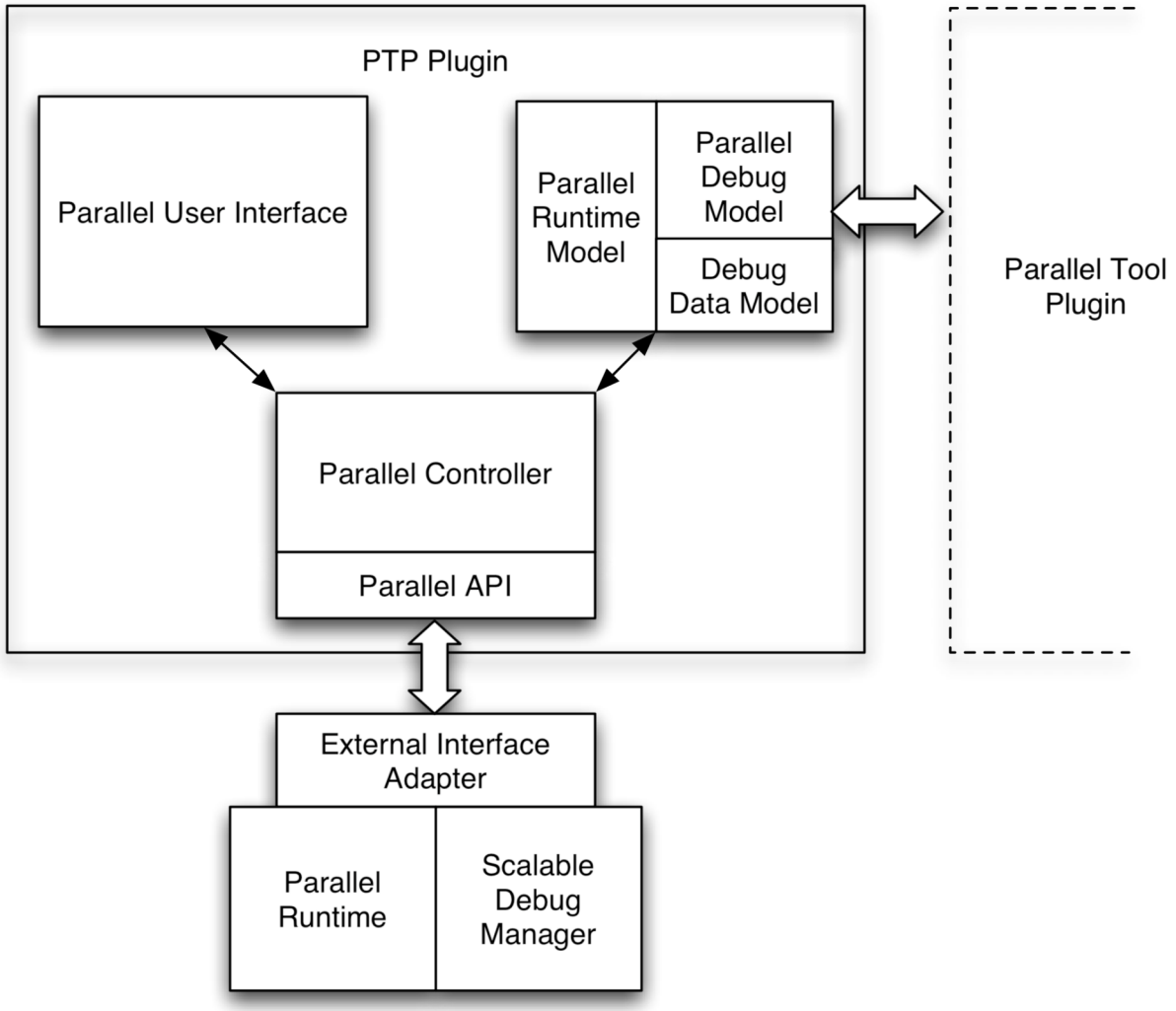
Parallel Tools Platform Architecture



Parallel Tools Platform Architecture

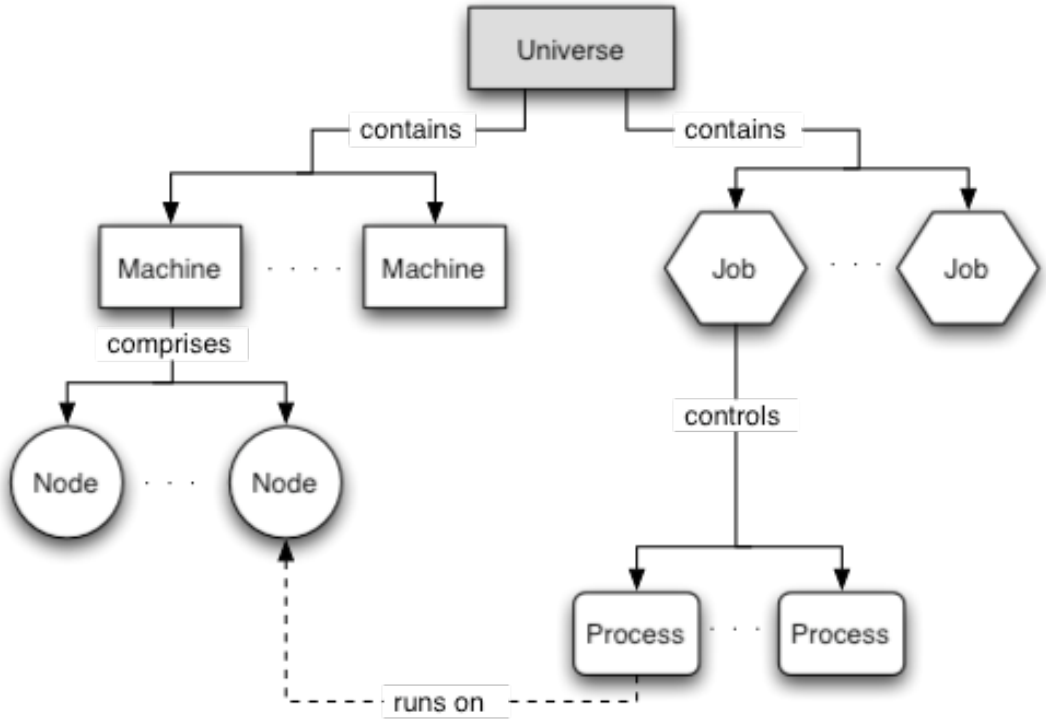
- Parallel Tools Platform Plug-in
  - Extends existing components where necessary (e.g. debug model)
  - Adds new parallel functionality (e.g. parallel launch wizard, user interface components)
  - Utilizes existing language support (e.g. CDT)
  - Provides infrastructure to support other parallel tools
  - Interfaces to external parallel runtime systems
- External Parallel Runtime
  - Target OpenMPI runtime (*not* dependent on MPI)
  - Should support other runtime systems (requires work)

Parallel Tools Platform
Plugin Detail



Parallel Tools Platform
Parallel Model

- Runtime Model
  - Defines notion of *universe*, parallel machines, jobs, processes, etc.



Parallel Tools Platform
Parallel Model

- Debug Model
  - Extends platform/CDT debug model to support parallel processes
  - Support for group operations
  - Scalable event management
- Debug data model
  - Efficient data handling and internal data manipulation
  - Data representation is not UI centric
  - Language independent (support for *arbitrary data types*)
  - Efficient conversion to/from Java native types
  - Efficient manipulation in intermediate form
  - Value caching and lazy evaluation

Parallel Tools Platform
Parallel Controller

- Controls interaction between PTP components and external runtime
- Manages debugger specific operations and event handling

Parallel Tools Platform
Parallel User Interface

- Specifically designed to provide compact and scalable interface to parallel components
- Extends existing debug interface to support parallel processes, array viewing, and new visualization tools
- Launch Configuration
  - Specify resource requirements (e.g. number of processes, execution time, etc.) and interface to launch services

Parallel Tools Platform
Parallel API

- High level interface for interacting with parallel machines, managing jobs, etc.
- High level API for managing debug operations
- Allows PTP to support different parallel runtime systems and parallel debug managers

Parallel Tools Platform
Scalable Debug Manager

- External component
- Manages scalable and efficient debugging of enormous parallel programs
  - Launching processes under debug control
  - Communication with large numbers of processes
  - Efficient data transfers
  - Event management

Parallel Tools Platform
Tool Integration

- Runtime Services
  - A range of services for launching, running and controlling parallel programs
- Debug Services
  - Services for managing parallel programs under debugger control
- User Interface Components
  - Reusable user interface components designed to be compact and scalable
- Tool Specific Services
  - Still being defined

Parallel Tools Platform
End-User Support

- Many components of PTP support *running* and *managing*, rather than *developing* parallel programs
- Utilize RCP to provide an environment that targets the end-user, rather than the developer
- Deal with issues that are usually ignored:
  - How to manage program input and output data
  - Pre- and post-processing of data
  - Visualization, model coupling, etc.

Parallel Tools Platform
Fortran Development Tools

- Still the predominant language for parallel scientific computing
- Support is essential for adoption of Eclipse in the parallel computing community
- Moving target:
  - Many standards: 66, 77, 90, 95, 2003
  - Substantial differences between versions
  - Then there's the extensions:
    - HPF, Cray, VAX, IBM, KAP, LS, etc.

Parallel Tools Platform
Future Work

- New parallel tools
  - What new possibilities become available when tools are tightly integrated?
- Advanced debugging
  - Can the close integration of tools and debuggers lead to new debugging techniques?
- Lightweight tools
  - Can the developer or end-user benefit from an array of simple, useful tools?

Parallel Tools Platform Demo…

Conclusion

- Unique opportunity to move parallel development to best practice
- Designed to address scalability and performance issues from the beginning
- Potential to push the development of parallel tools into new areas
- Growing interest from the parallel scientific computing community
- Seeking YOUR support and involvement!