Challenges in Application Scaling In an Exascale Environment

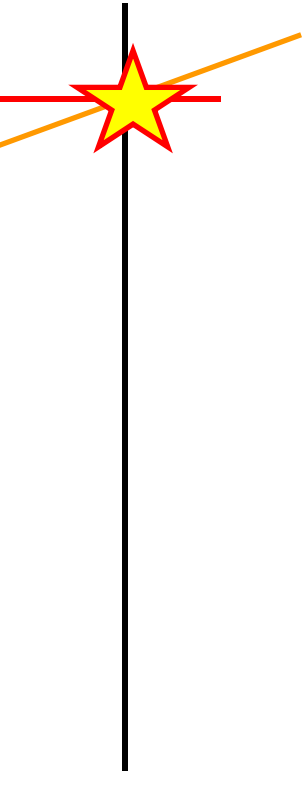
14th Workshop on the Use of
High Performance Computing
In Meteorology

November 2, 2010

Dr Don Grice
IBM

Growth of the Largest Computers by year
1000x every 10-11 years

ExaFlop in 2018-2019!



Technology Trends

Different Scaling Trends for Different Technologies Compute Ratios will Change

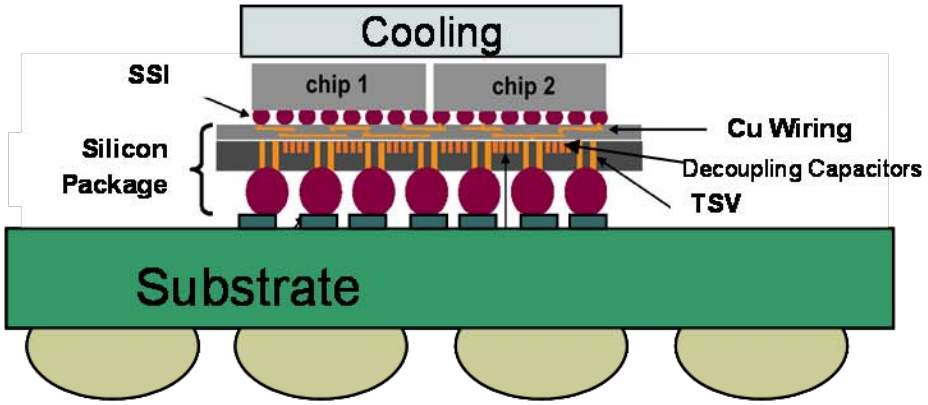
- Driven by Cost and Energy Usage
- Circuit-Flop Densities will Continue to Improve
- I/O BWs and Power will not improve as quickly
  - Technology Improvements may help this
  - Costs may still be limiters
- Memory Volume Costs (and BWs) may be Limiting

The Big Leap from Petaflops to Exaflops

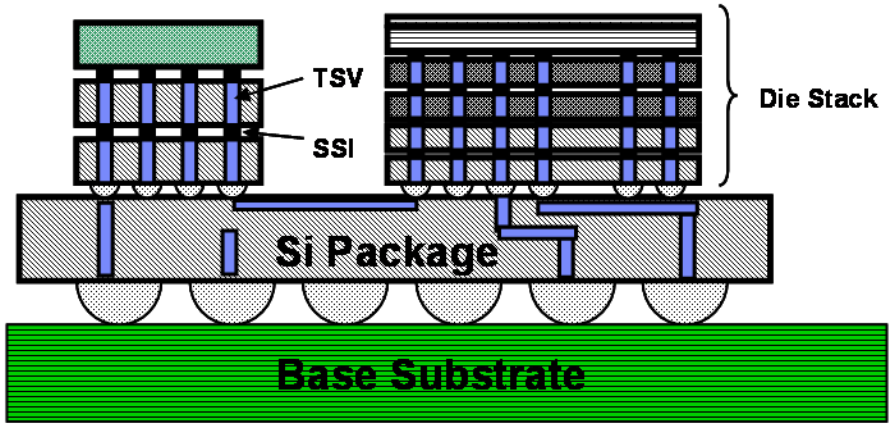
- Technology disruptions in many areas driven by Power and Cost Concerns.
- All Impact System Balance and Application Optimization
  - Silicon power scaling:
    - Frequency Plateau – more threads needed
    - Impacts Application Scaling, Power Usage, and RAS
  - Memory technology – Volume and BW
    - Bytes/Flop ratios decreasing (Locality Counts)
  - Interconnect BW
    - Bytes/Flop ratios decreasing (Locality Counts)
  - Packaging technology – I/O Switching Costs
    - Relative Amount of Power needed to move Data Increasing
- Need to be able to exploit machines. Not just about flops.
  Flop metric promises to be an even poorer predictor of sustained performance than it is now

3D Si Pkg & Si Die Stacking

3D Silicon Pkg Integration



3D Die Stack & Si Pkg Integration



3D Microsystems Technology:

1. Through-silicon-vias (TSV) & Thin Si
2. Fine pitch wiring / interconnection
3. Integrated active devices / passive function
4. Silicon-Silicon Interconnection (SSI) die stacks & Si pkg (C2C, C2W, W2W)
5. Test - Known good die & Die stacks
6. Power delivery / advanced cooling

Opportunities:

- High bandwidth
- Low Power
- Heterogeneous chips
- Modular Design
- Modular System Integration
- System Power Savings
- System Performance Gain
- System Miniaturization
- Time to Market
- Lower cost

DARPA IPTO Sponsored (Information Processing Techniques Office) ExaScale Software Study: Software Challenges in Extreme Scale Systems September 14, 2009

"This document reflects the thoughts of a group of highly talented individuals from universities, industry, and government research labs on the software challenges that will need to be addressed for the Extreme Scale systems that are anticipated in the 2015 – 2020 time-frame."

Used as Reference Point for some Application Challenges

Locality Tension/Trade-offs

- Energy to move data will be significant
- Application run time optimization may run counter to keeping things local
  - Convergence time reduction and Locality may be at odds
  - AMR may disrupt locality
  - Multi-Scale and Multi-Physics may disrupt neighbor locality
  - Unstructured Grids disrupt Neighbor Communication Patterns
  - Global Data Access may aid convergence
- Models need to consider:
  - Asynchronous lightweight tasks and communications
    - Latency Hiding
  - Explicit Locality Model
  - Scalable Coordination and Synchronization
  - Abstract Performance Model: Parallelism, Locality, Energy

Reliability

- New technologies can be leveraged to deliver a reliable system even at an Exascale.
  - The biggest challenge is that architectures must be willing to pay the price of hardware detection.
  - There will be a tension between energy efficiency and error detection
  - Do we need a software construct like a ‘critical section’ that must get the ‘right answer’?

Petascale/Exascale Computing

Software Challenges

Types of Application Scaling

- Strong Scaling
  - Making the same problem run in less time or running more time steps in the same time.
  - Not many applications exhibit strong scaling
- Weak Scaling
  - Increasing the problem size to run on a larger machine in a ‘tractable time’. (e.g. Spatial Scaling)
  - Finer meshes generally imply smaller time steps
  - Creates an issue when a problem has a fixed window to run in
- ‘New Era’ Scaling
  - Increasing the amount of science computed during the solution of the problem.

"New Era" Weak Scaling
Not Just Solving Things Faster – Solving Them Better

"New-era" weak scaling typically adds extra work through one or more of the following:

- Multi-scale
- Multi-physics (multi-models)
- New models
- Interactions
- Mitigation analysis
- Data mining
- Data-derived models

Scaling Limitations

Not all applications will scale to exascale with their current structure due to things like:

- Parallelism
  - O (10**11) Threads required
  - Load Imbalance and Serial Portions
- Locality
  - Vertical (temporal)
    - Data Reuse is not always possible
    - Movement in the Memory Hierarchy Occurs
  - Horizontal (data decomposition)
    - Excessive Movement uses Energy
    - Introduces Latency and Serialization Issues
- Bottlenecks in Memory, Communications, and I/O BWs

Exascale Software Challenges: Summary

- Dealing with Frequency Stabilization
  - Increased number of threads for additional performance
  - Load Balancing is critical for Sustained Performance Gains
- Dealing with Performance Ratio Changes
  - Computation will be ‘free’
  - Recomputing vs moving data may be the best solution
- Dealing with Memory Footprint issues
  - A view of exascale machines as a memory system with embedded compute facilities may be helpful
- Debugging and Optimizing Codes at Scale
  - New Tools and methods are evolving

HPC Software Tools
What's being done to Help with Scaling?

10X System Wide Productivity Improvements

Programmer

- Eclipse Tools
- Compiler Enhancements
- UPC Language
- Automated Performance Tuning

System Operational Efficiency

- Resource & Workload Management
- Protocol Optimizations & PGAS
- Co-scheduling
- Dynamic Page Size Assignments

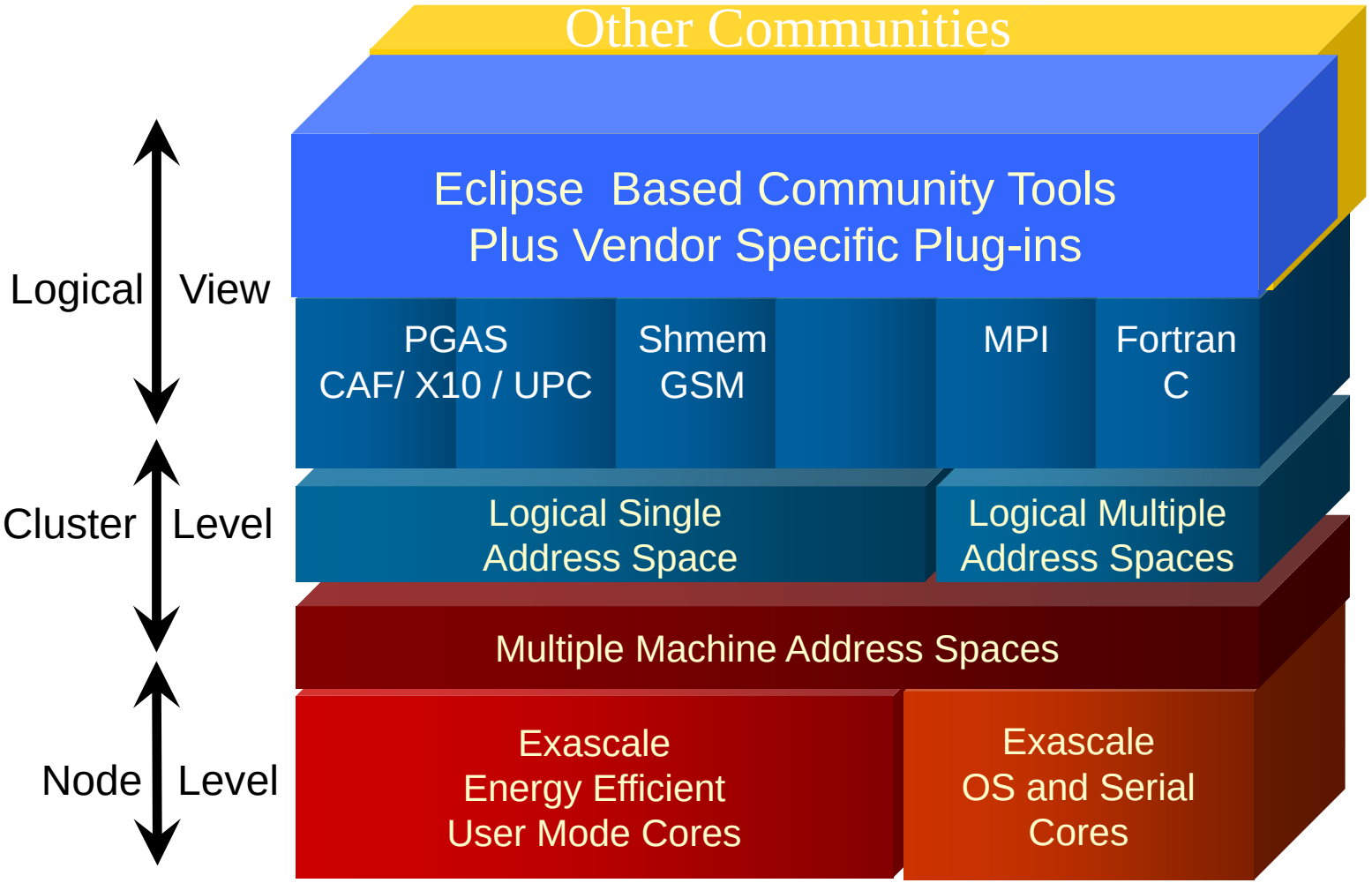
Administrator

- Automated Discovery
- Automated Configuration
- Diskless Boot
- Rolling Updates
- Server, Network, & Storage

Reliability and Serviceability

- Continuous Operation
- Checkpoint/Restart
- Server, Network, & Storage Monitoring
- Problem Determination

Application Writer‘s View of the System

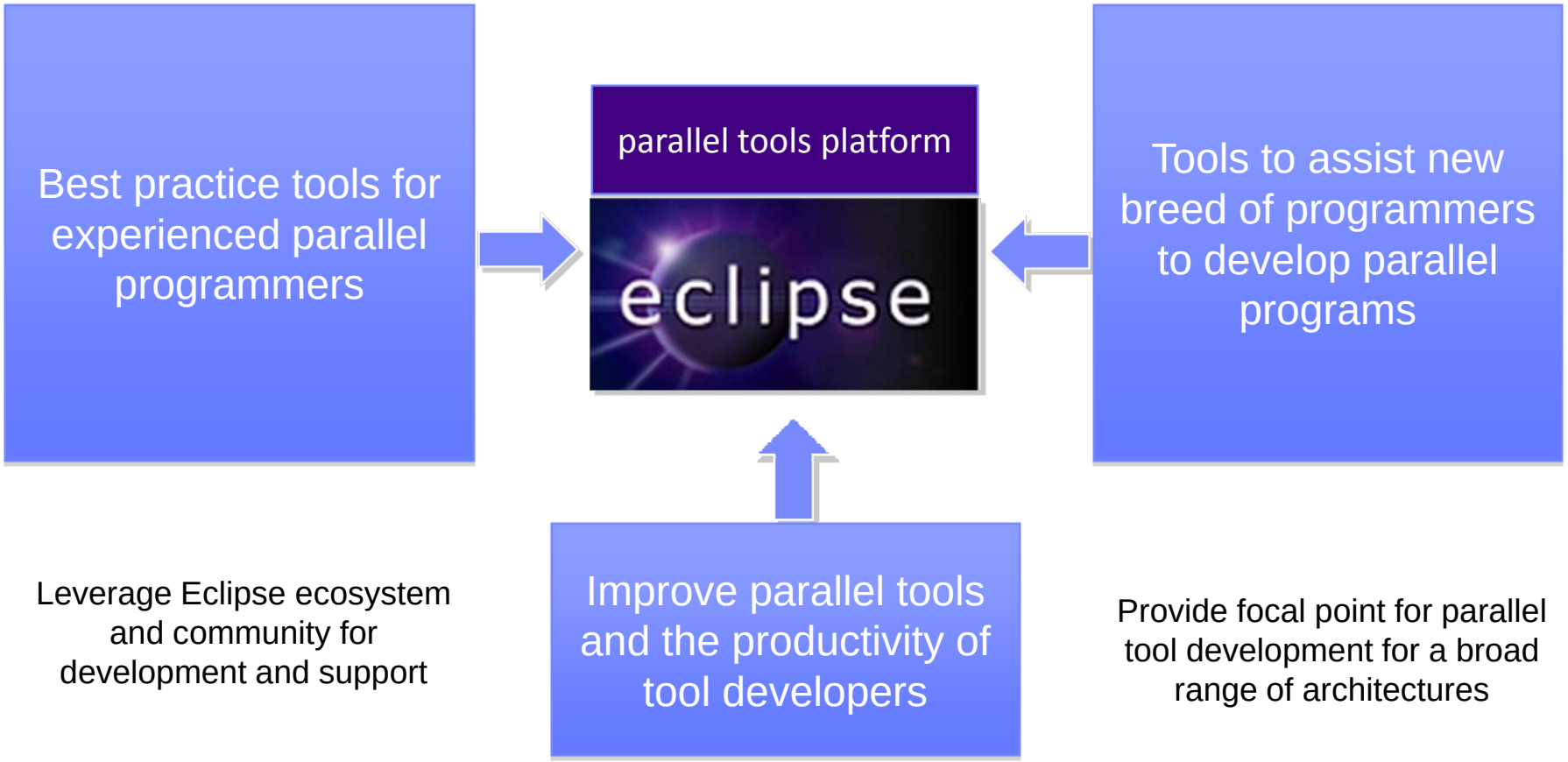


IBM PERCS Goals
Productive, **E**asy to Use, **R**eliable **C**omputing **S**ystem

- Collaborative ecosystem based on Eclipse
  - **Parallel Tools Platform**
  - Integrated tools
- Multi-language support
  - C, C++, UPC, Fortran, X10
- Platform independence
- Highly scalable
  - 1M Tasks
- Enhanced productivity
  - Lower entry point for new users

Parallel Tools Platform (PTP)

Enabling Parallel Application Development

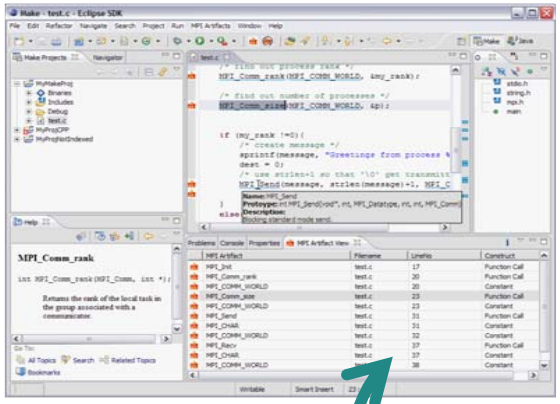


Contributors

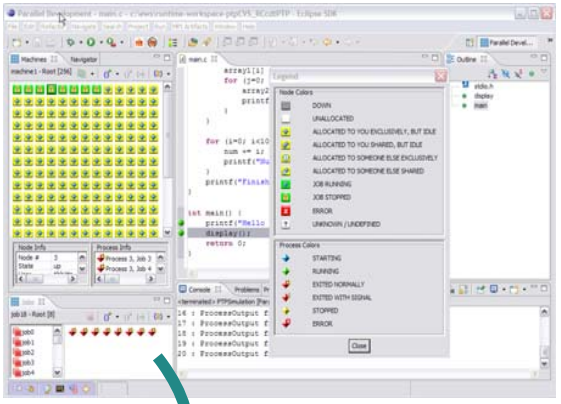
- IBM
- Los Alamos National Laboratory
- Oak Ridge National Laboratory
- University of Tennessee at Knoxville
- University of Oregon
- Monash University
- National University of Defense Technology
- NCSA
- University of Illinois
- University of Utah
- Munich University of Technology
- Forschungszentrum Juelich

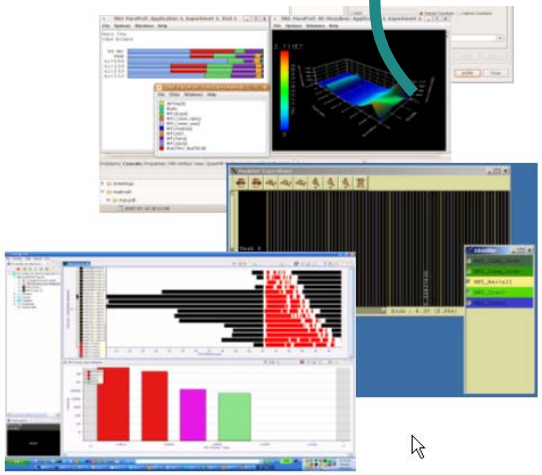
HPC Application Development Cycle

Coding & Static Analysis

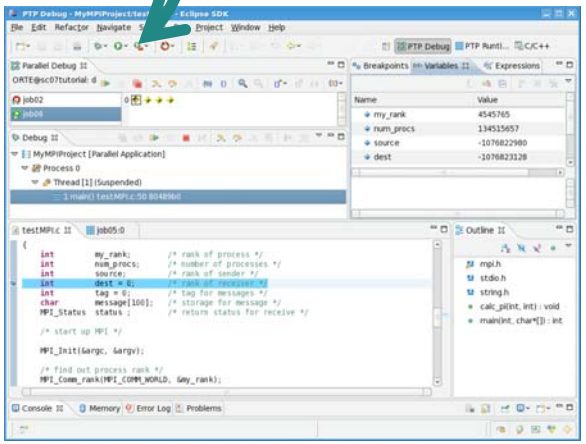


Application Execution





Dynamic & Performance Analysis

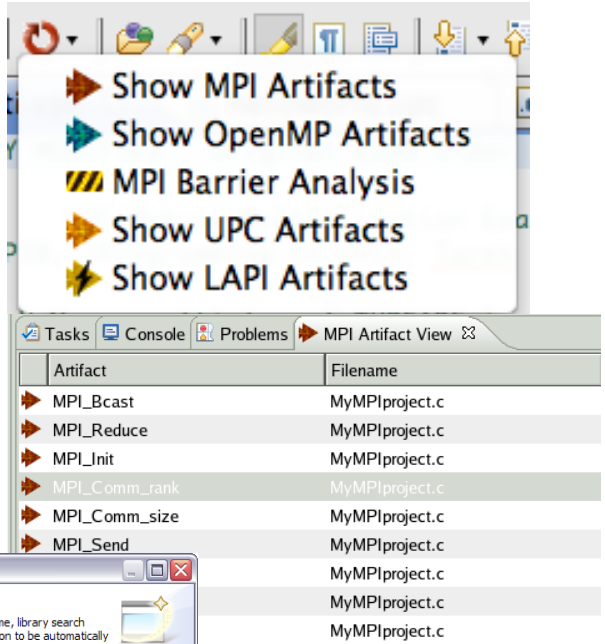


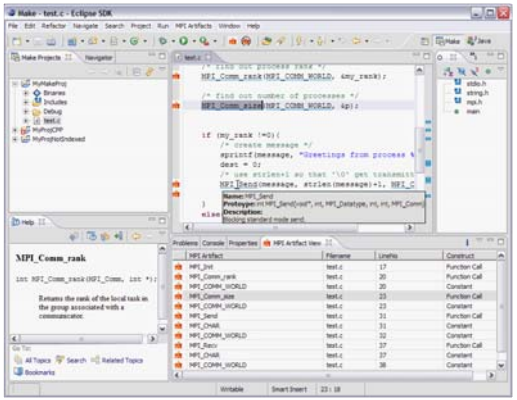
Application Debugging

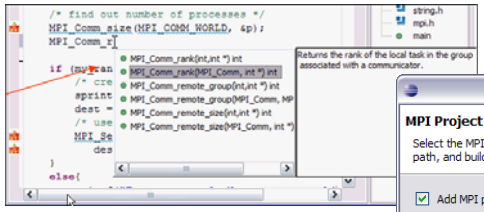
Coding & Static Analysis

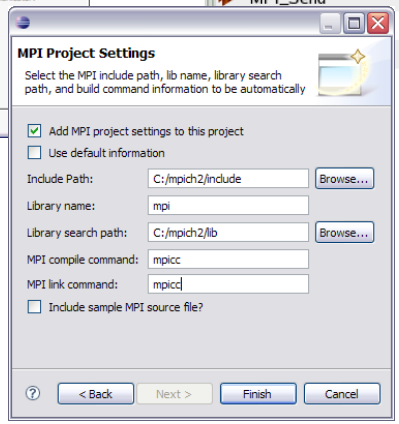
Parallel Language Development Tools (PLDT)

- Assistance tools to increase productivity of parallel programmers
  - New project wizards (MPI, OpenMP)
  - Content Assist (command/API completion), hover help, built-in API help descriptions in an html help “view” (MPI, OpenMP, LAPI, UPC)
  - Location of parallel “artifacts” in code: MPI, OpenMP, LAPI APIs, and UPC



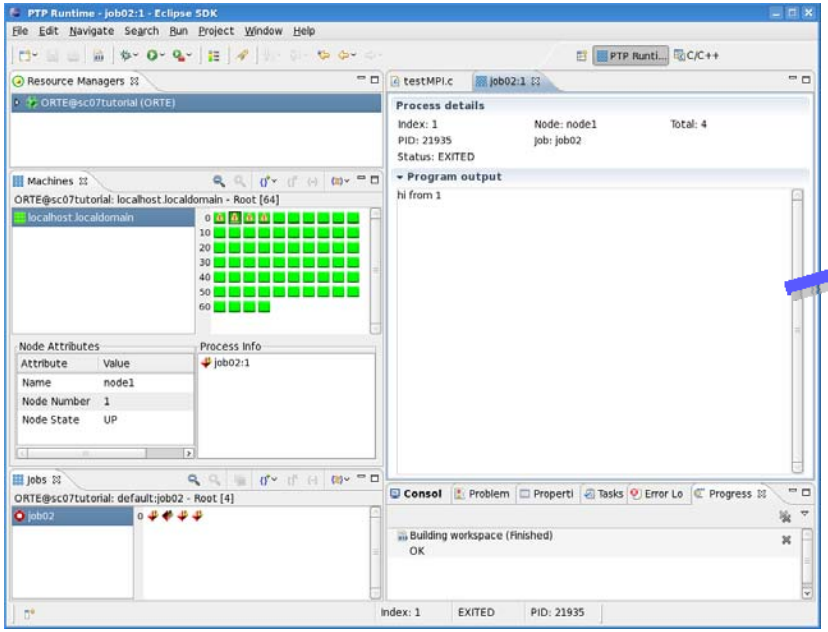


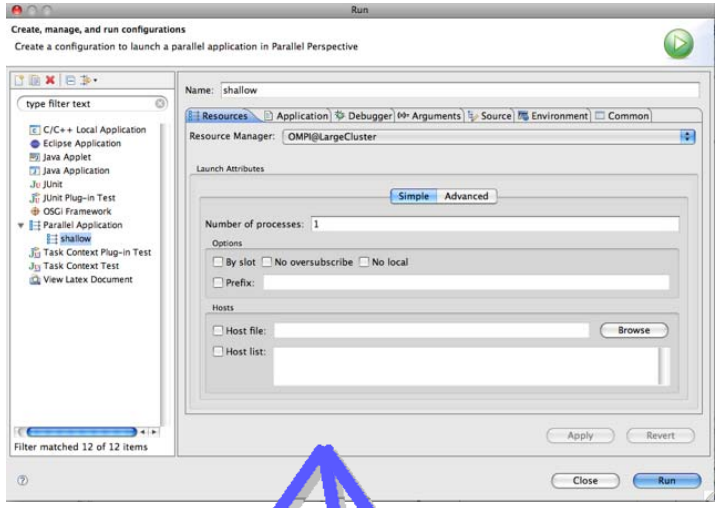


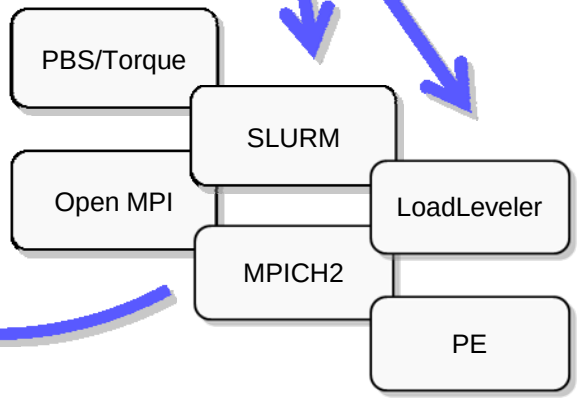


Application Execution

Launching & Monitoring



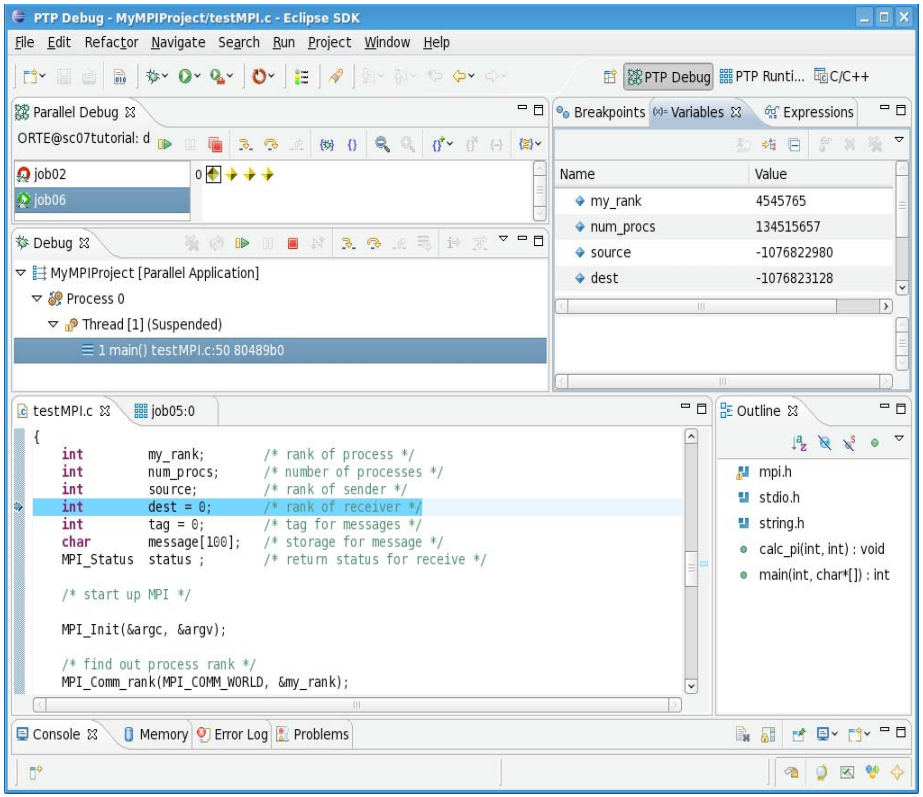




- Improves visibility into target system
- Single point of interface for launching and control
- Manages interaction with different runtime systems and job schedulers

Application Debugging

PTP Parallel Debugger

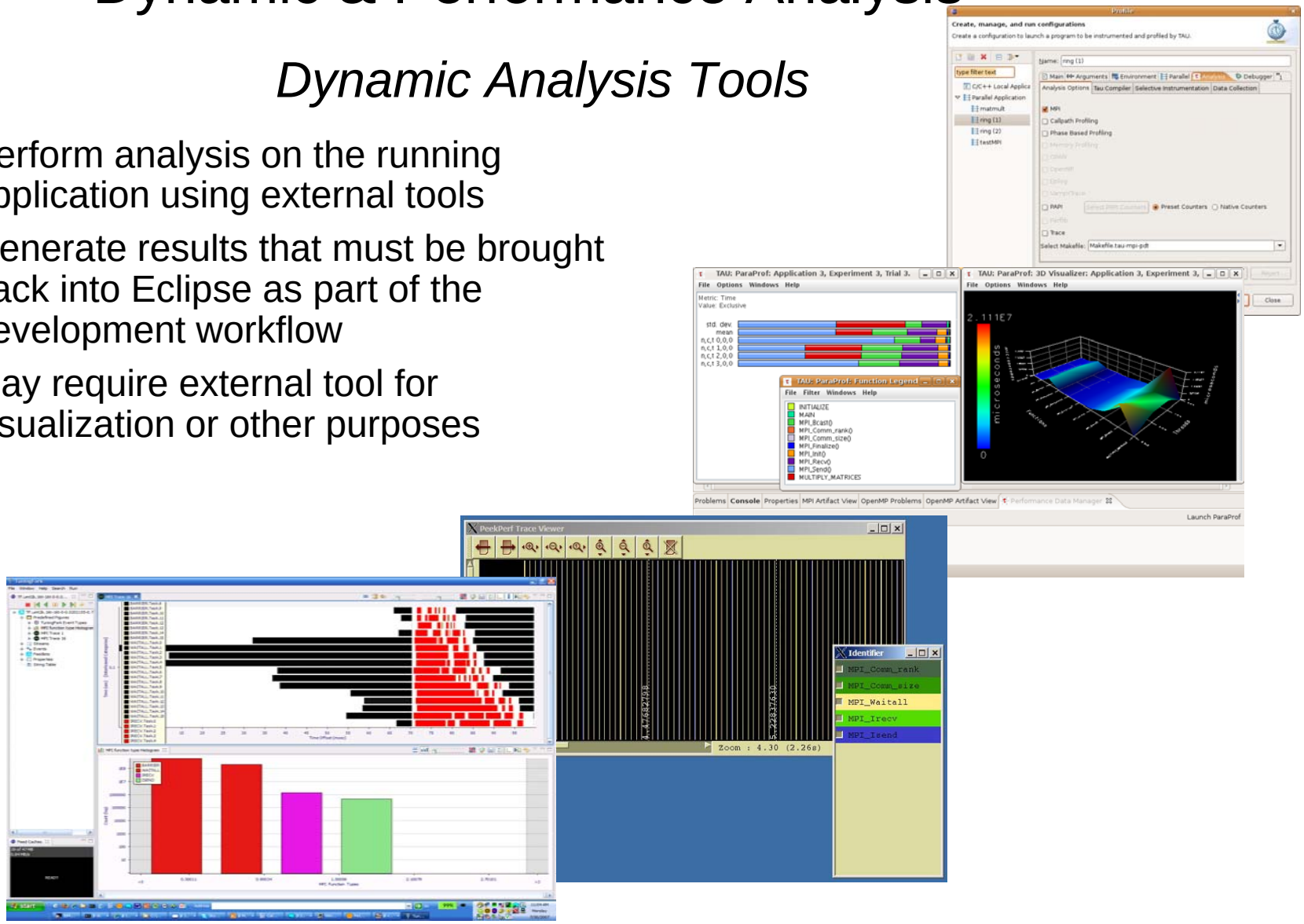


- Mid-scale integrated debugger
- Tightly integrated with Eclipse
- Supports debugging multiple jobs simultaneously
- Utilizes backend debugger (e.g. gdb) for low level operations
- Targeted at SPMD programming models
- Supports mixed MPI & thread debugging
- Single process and group operations
- Platform for building new debugging paradigms

Dynamic & Performance Analysis

Dynamic Analysis Tools

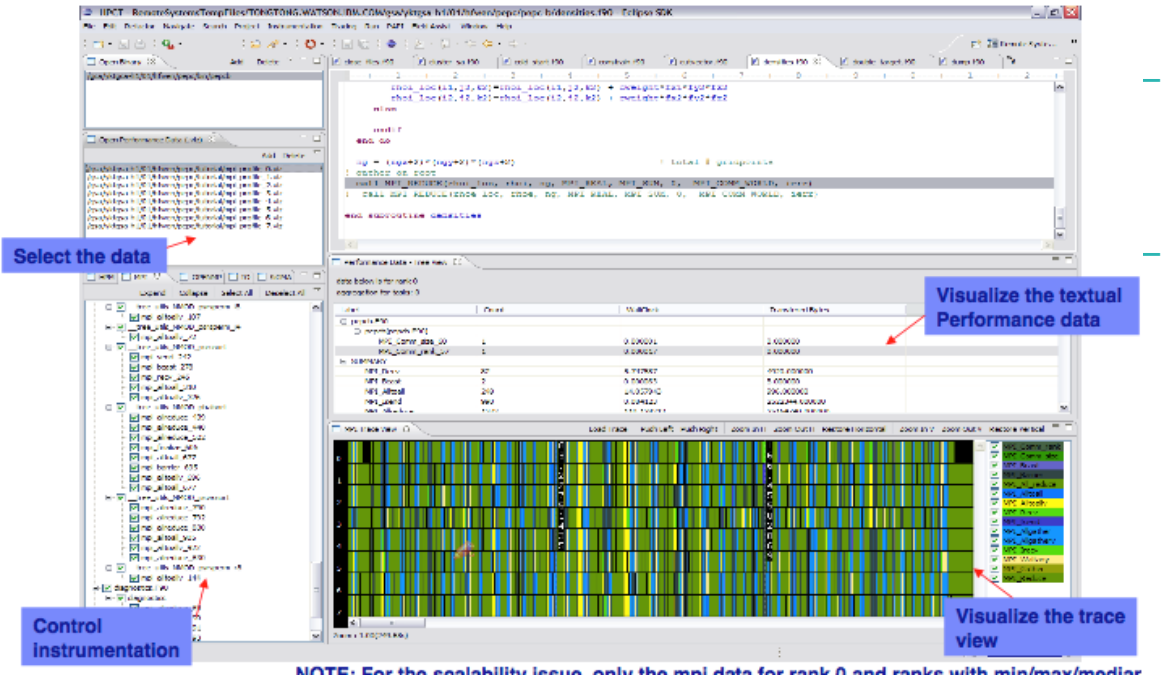
- Perform analysis on the running application using external tools
- Generate results that must be brought back into Eclipse as part of the development workflow
- May require external tool for visualization or other purposes



Dynamic & Performance Analysis

HPC Toolkit

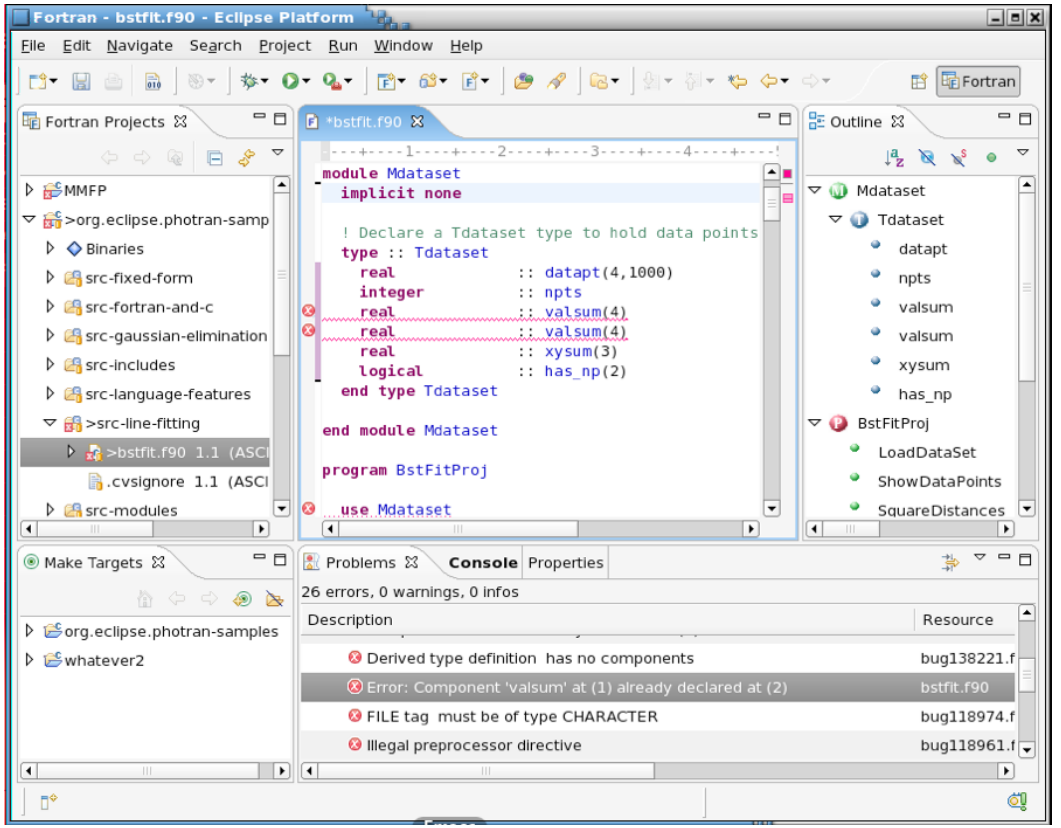
- Integrated with Eclipse and PTP
- Application and tools launched using PE resource manager



- Provides an integrated framework for performance analysis
- Looks at all aspects of performance (communication, memory, processor, I/O, etc) from within a single interface
- Operates on the binary and yet provides reports in terms of source-level symbols
- Full source code traceback capability

Fortran Development Tools

- Photran project
  - http://eclipse.org/photran
  - Tech lead at UIUC, Jeff Overbey
  - More UIUC students are contributing
  - Merged with PTP in 2009
  - Photran 6.0 released with PTP 4.0 Helios
- Photran features:
  - Supports Fortran 77-2008
  - Syntax-highlighting editor
  - GUI interface to gdb
  - Makefile-based compilation
  - Compiler error extraction
  - Outline view
  - Open declaration
  - Fortran refactorings
  - C preprocessor support



Fortran Refactoring

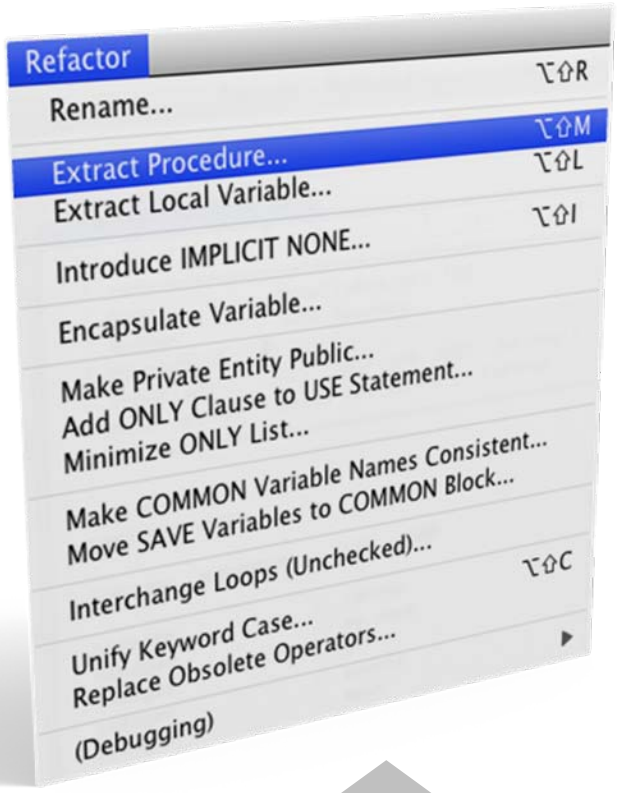


Image courtesy Jeff Overbey, UIUC

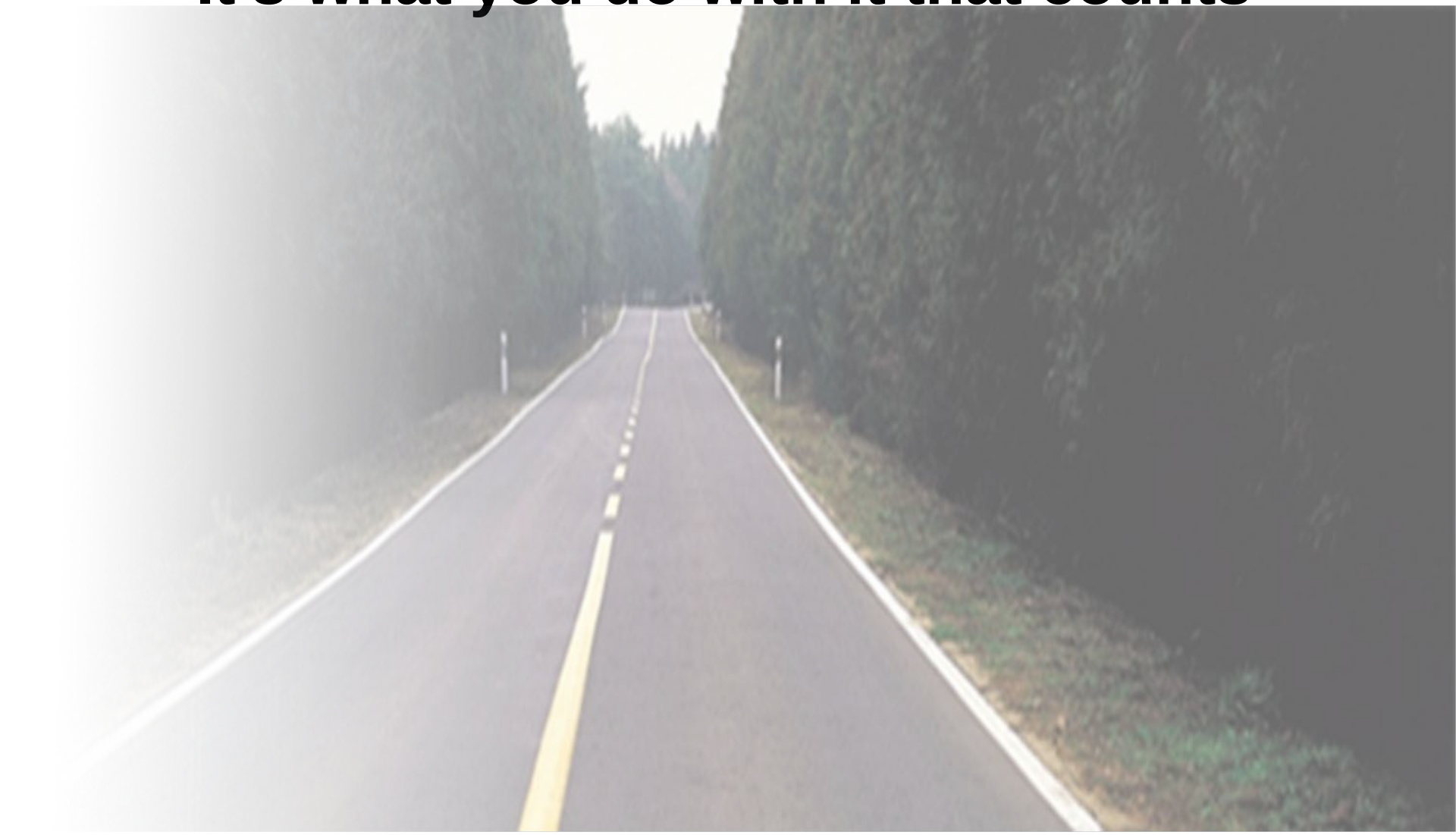
Some samples:

- Rename
  - Change name of every use of a variable, function, etc.
  - Only proceeds if new name will be legal
- Extract procedure
  - Moves statements into a new subroutine, replacing statements with a call
  - Local variables are passed as arguments
- Introduce implicit none
  - Adds an ‘implicit none’ statement
  - Adds explicit variable declarations
- Currently 16 refactorings

Conclusion

- Fundamental Programming Style not likely to change much
  - Multi-Threaded ‘MPI tasks’ will be the norm
  - New Languages are emerging to help with extreme scale
- A Shared Memory model at the Task level will still exist
- Amount of Threading will have to increase
- ‘More Science’ will be a way to use cycles
- Optimization Points will change – Computing is ‘free’
- New Tools are emerging to help create applications ‘at scale’

In the end, it's not about the technology; It's what you do with it that counts



Thank you...

...any Questions?