parallel tools platform

now and the future

Greg Watson

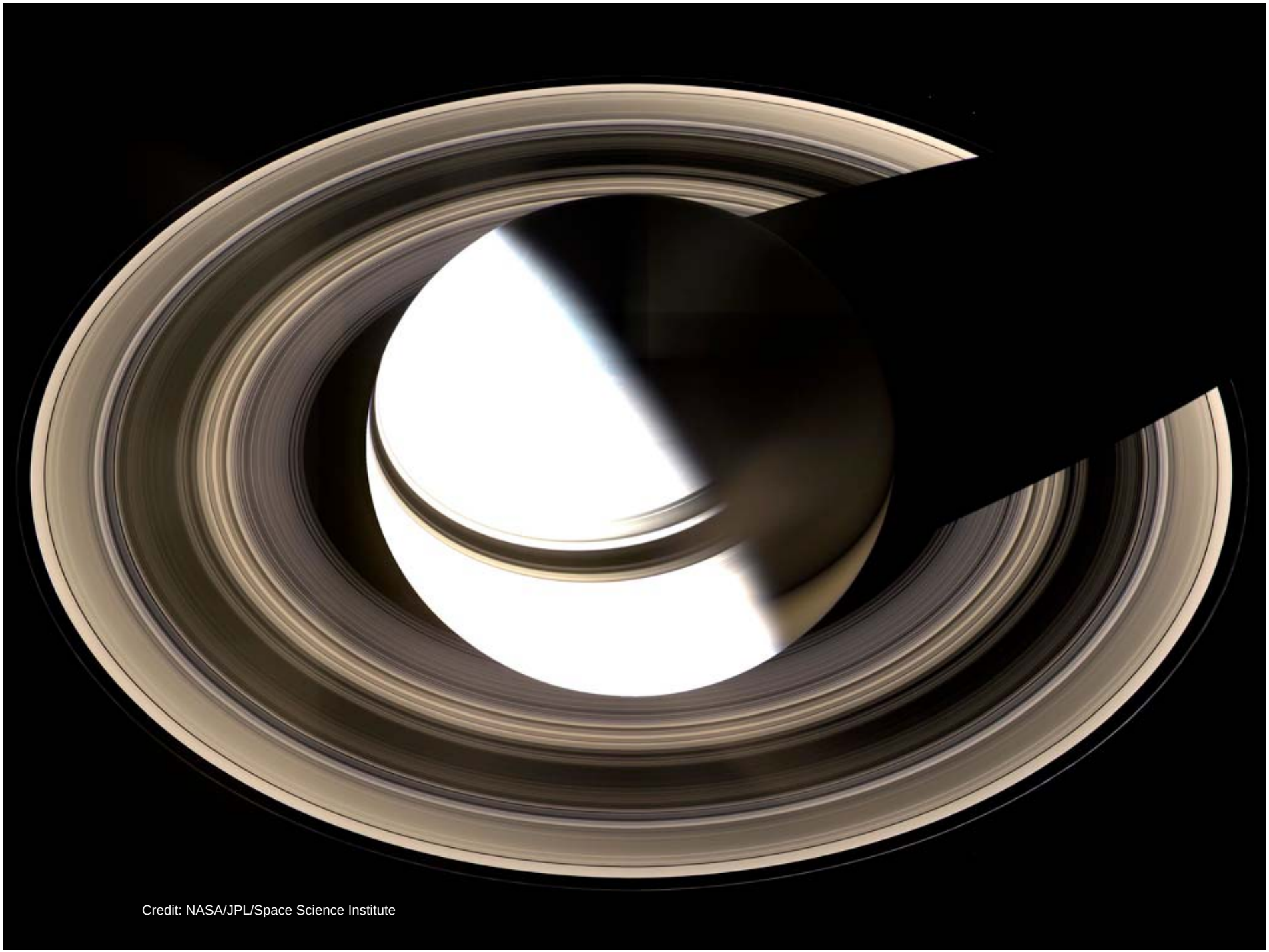
Los Alamos National Laboratory

IBM Research

Overview

- Parallel what?
- Current status
- What's coming in 2.0
- Future work

But first...



Credit: NASA/JPL/Space Science Institute

What do we mean by parallel?

- Hardware

Parallel hardware

embedded

small-medium clusters

“big iron” supercomputers

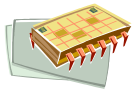
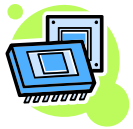
Parallel hardware charateristics

- Embedded
  - Usually small number of processes (< 8)
  - tightly coupled (memory bus or equivalent)
- Common-off-the-shelf (COTS) clusters
  - Usually less than 256 processors
  - Sometimes dual/multi core or SMP nodes
  - Loosely coupled with high-speed interconnect (GigE or Infiniband)
- “Big Iron”
  - 256 - 128K processors
  - Clusters, MPP, vector, ...
  - Myrinet, Infiniband, Quadrics, proprietary interconnects

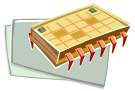
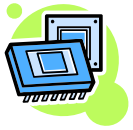
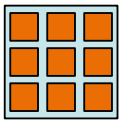
What do we mean by parallel?

- Software

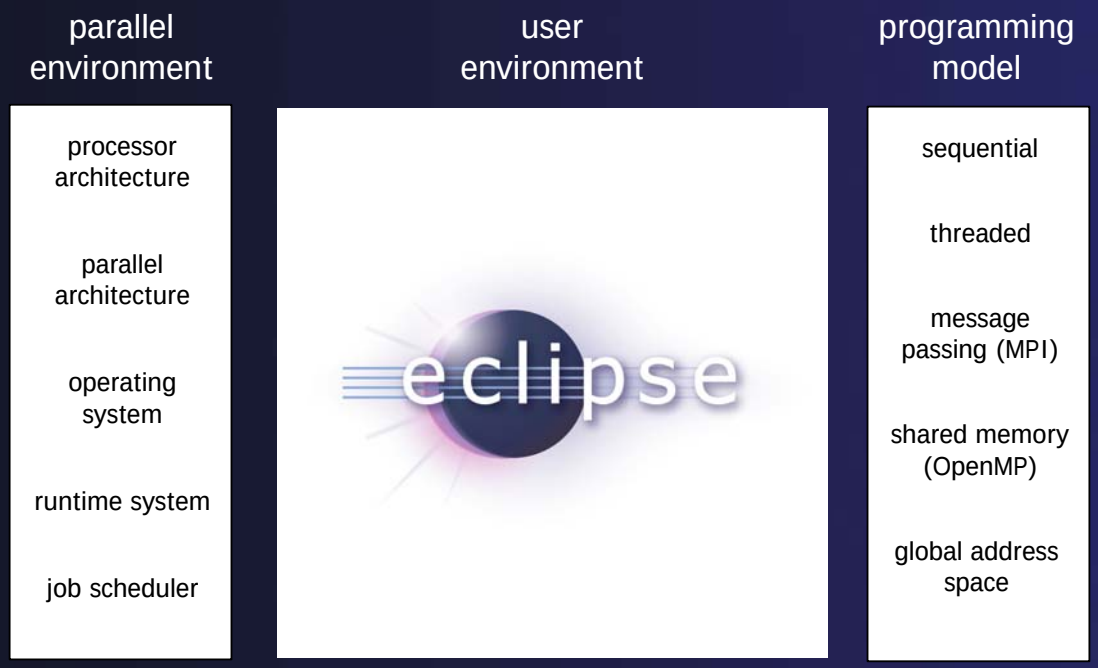
Traditional environment

processor architecture	operating system	user environment	programming model
 uniprocessor		command line	sequential
 SMP/dual core	Windows 	graphical user interface	threaded

Traditional parallel environment

processor architecture	parallel architecture	operating system	runtime system	job scheduler	user environment	programming model
 uniprocessor	cluster		cluster (poe, oscar, rocks, xgrid, xcat, xcpu)	LSF	command line	sequential
 SMP/dual core	MPP		single system image (mosix, openssi, bproc)	OpenPBS		threaded
 multicore	shared memory		mainframe/ proprietary	LoadLeveler		message passing (MPI)
	vector parallel			SLURM	(graphical user interface)	shared memory (OpenMP)
	proprietary			MOAB		global address space
				Condor		

PTP environment



How does PTP help?

- Provides the benefits of an IDE for parallel programmers
- Hides much of the parallel system complexity
- Simplifies the parallel programming task
- An opportunity for developing new/advanced tools and languages

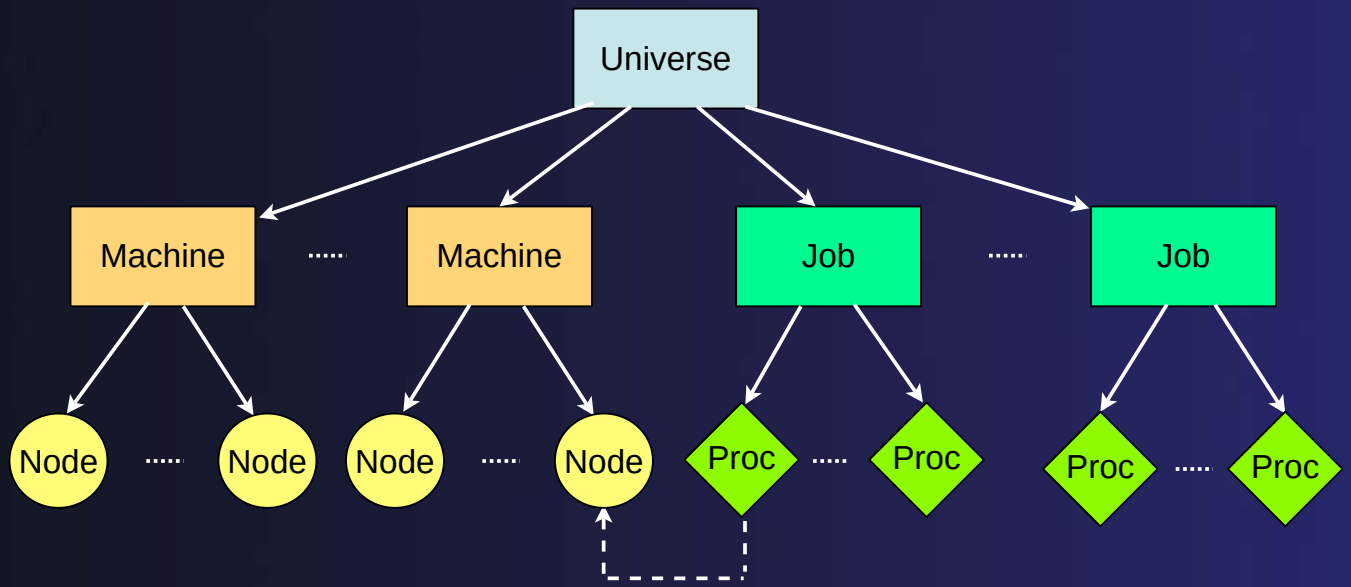
Current status

- Project created in April 2005
- PTP 1.0 released in March 2006
- PTP transitions to Tools Project December 2006
- PTP 1.1 released in February 2007
- PTP 2.0 scheduled for Summer 2007

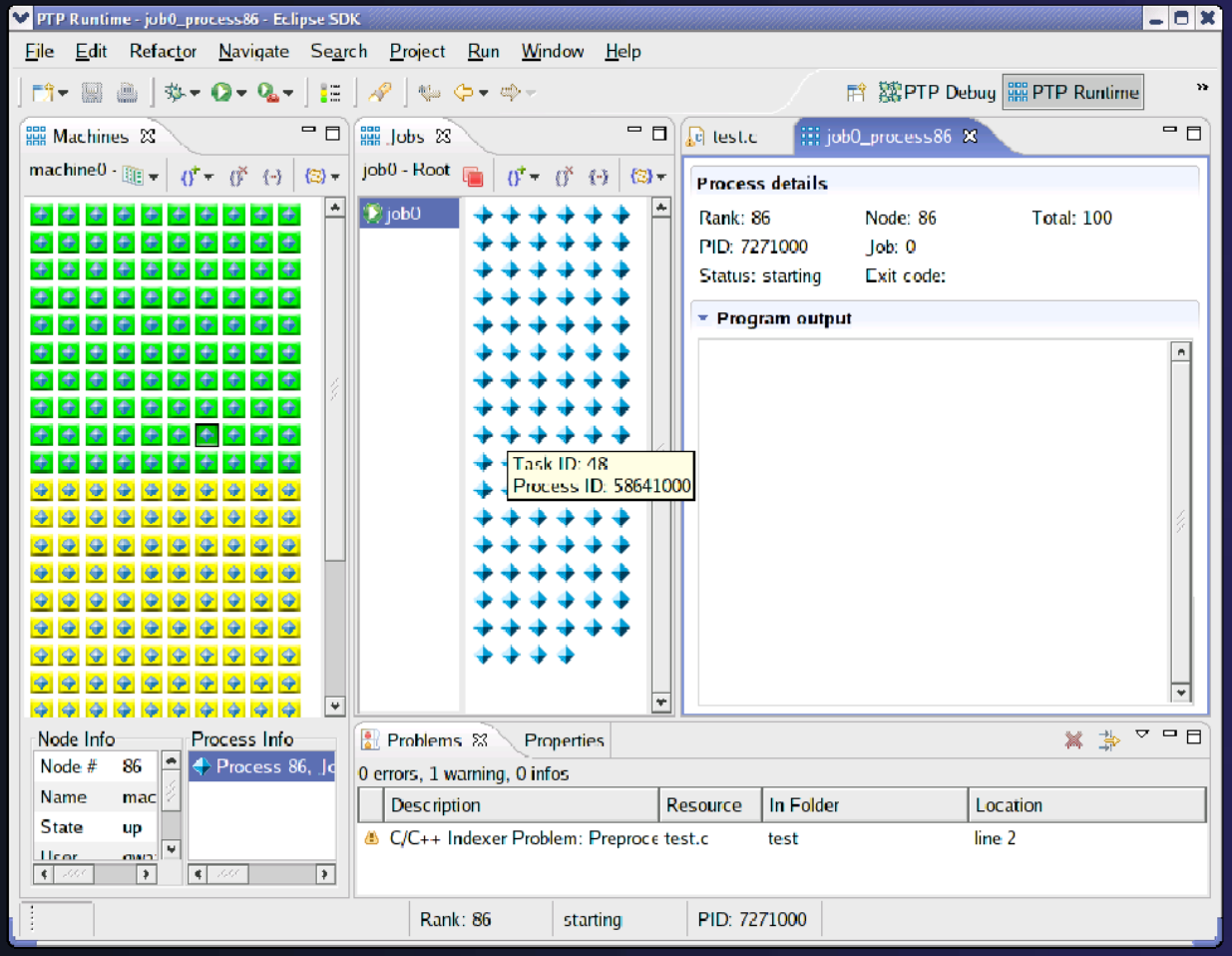
What's in PTP 1.1

- Attributed parallel system model
  - An abstract representation of a parallel system
  - MVC pattern
- New perspectives
  - Runtime and debugger
- Launch configuration for parallel jobs
- Parallel debugger
- Parallel programming tools
- Fortran development tools

Attributed parallel system model (1.1)

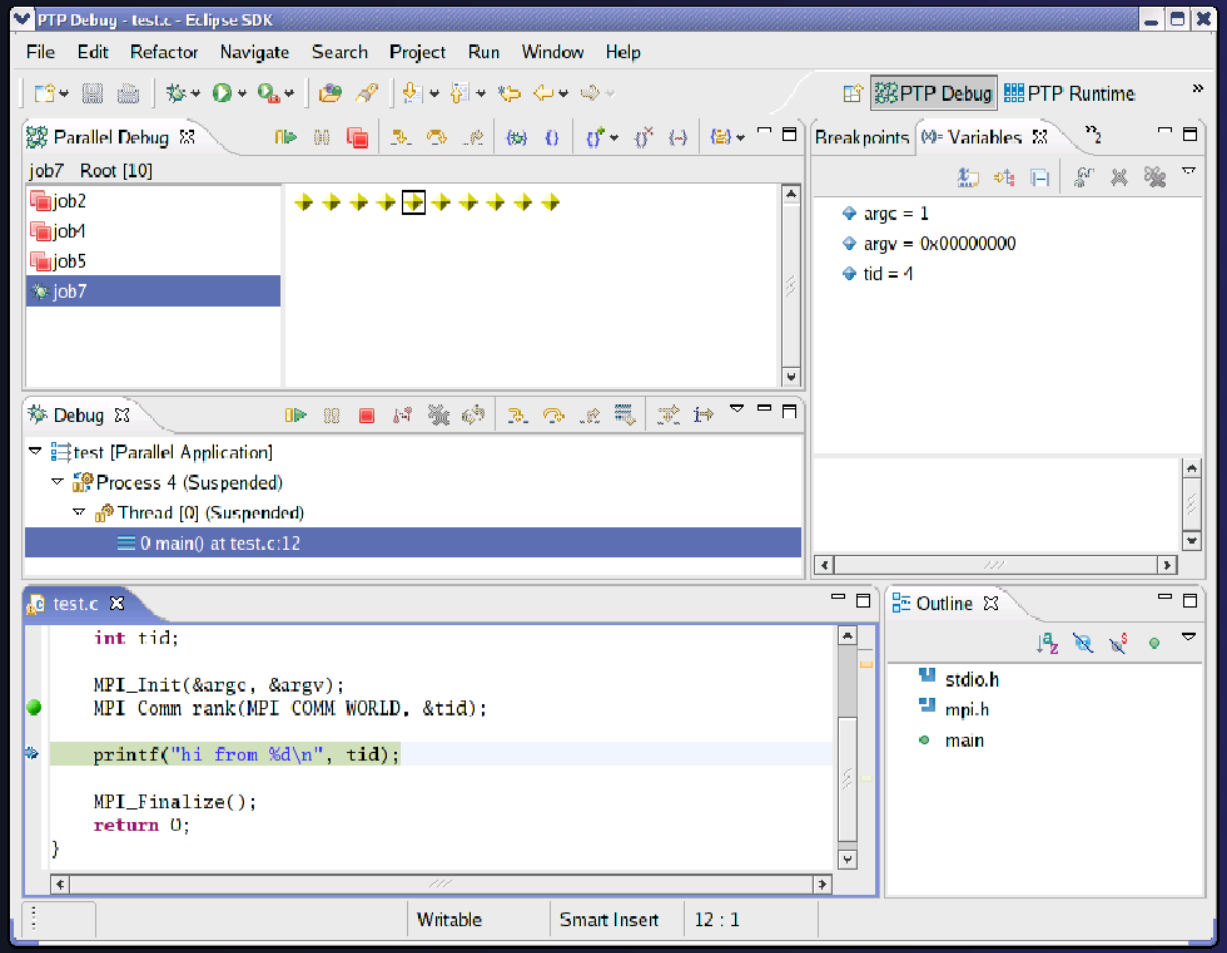


PTP runtime perspective



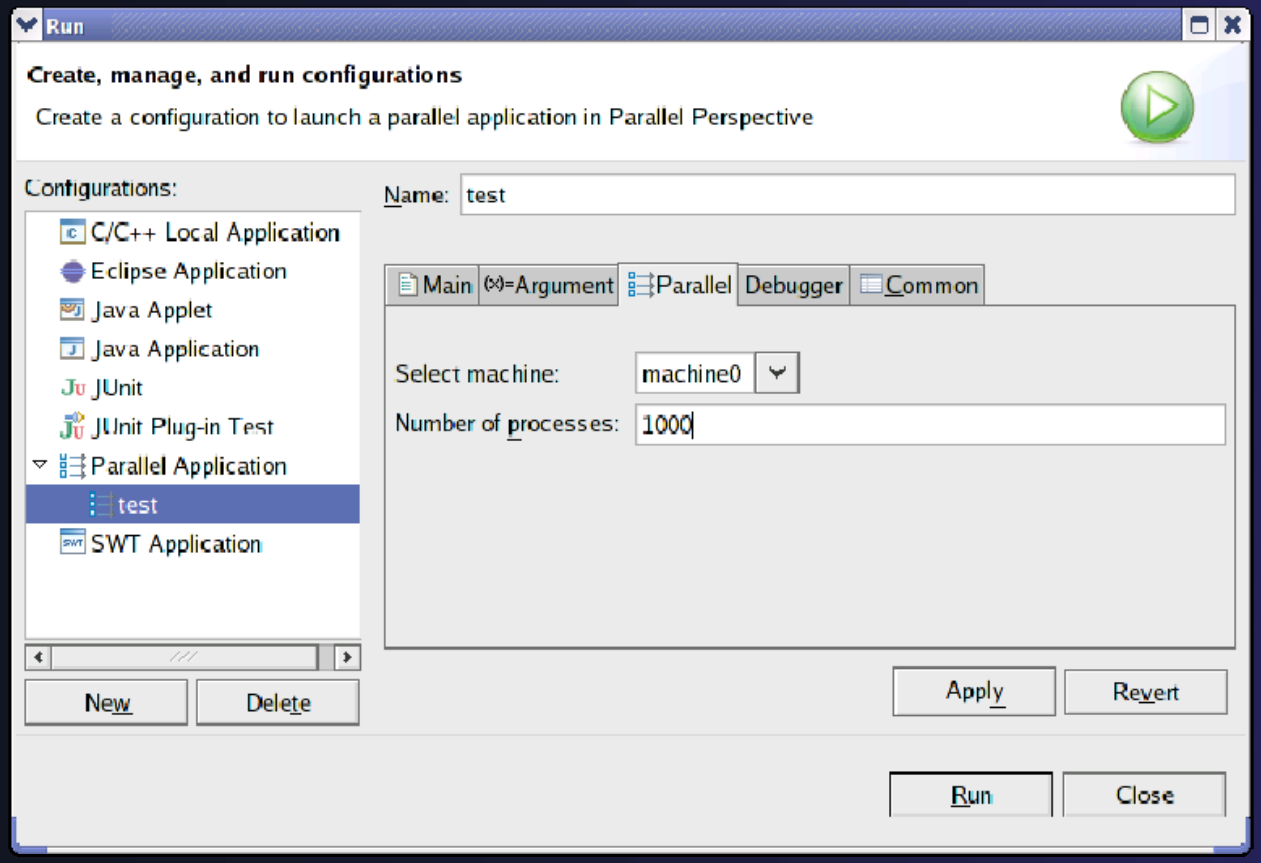
- Machines view
  - node status, node details, and processes running on the node
- Jobs view
  - jobs launched, processes in a job, and process status
- Process details view
  - more detailed process information and standard output from individual processes

Parallel debugger



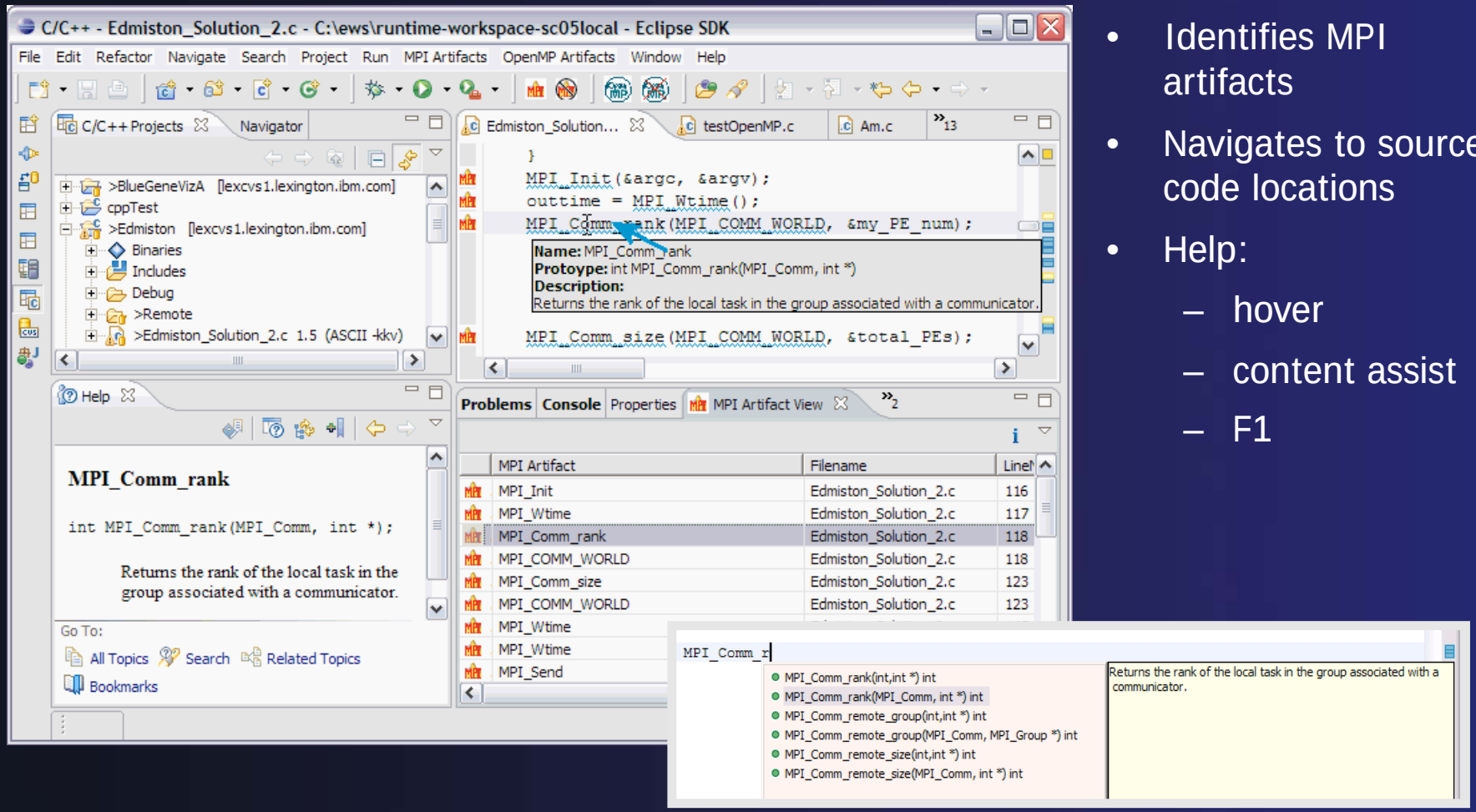
- Parallel debug view
  - debug jobs launched, processes in a job, and process status
  - process sets
  - registered processes
  - tooltip display
- Extended breakpoints and location markers
  - breakpoint color shows which set associated with the breakpoint
  - multiple simultaneous location markers

Parallel launch



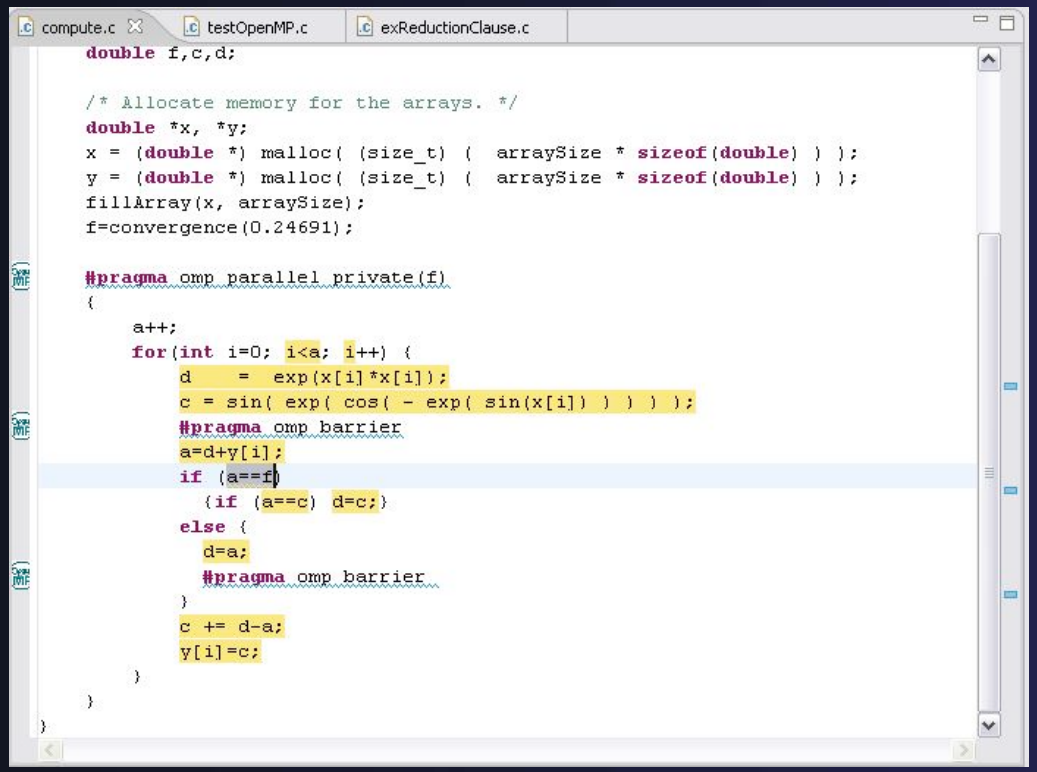
- User can specify the machine and number of processes to launch
- Configure parallel debug launch

Parallel programming tools



- Identifies MPI artifacts
- Navigates to source code locations
- Help:
  - hover
  - content assist
  - F1

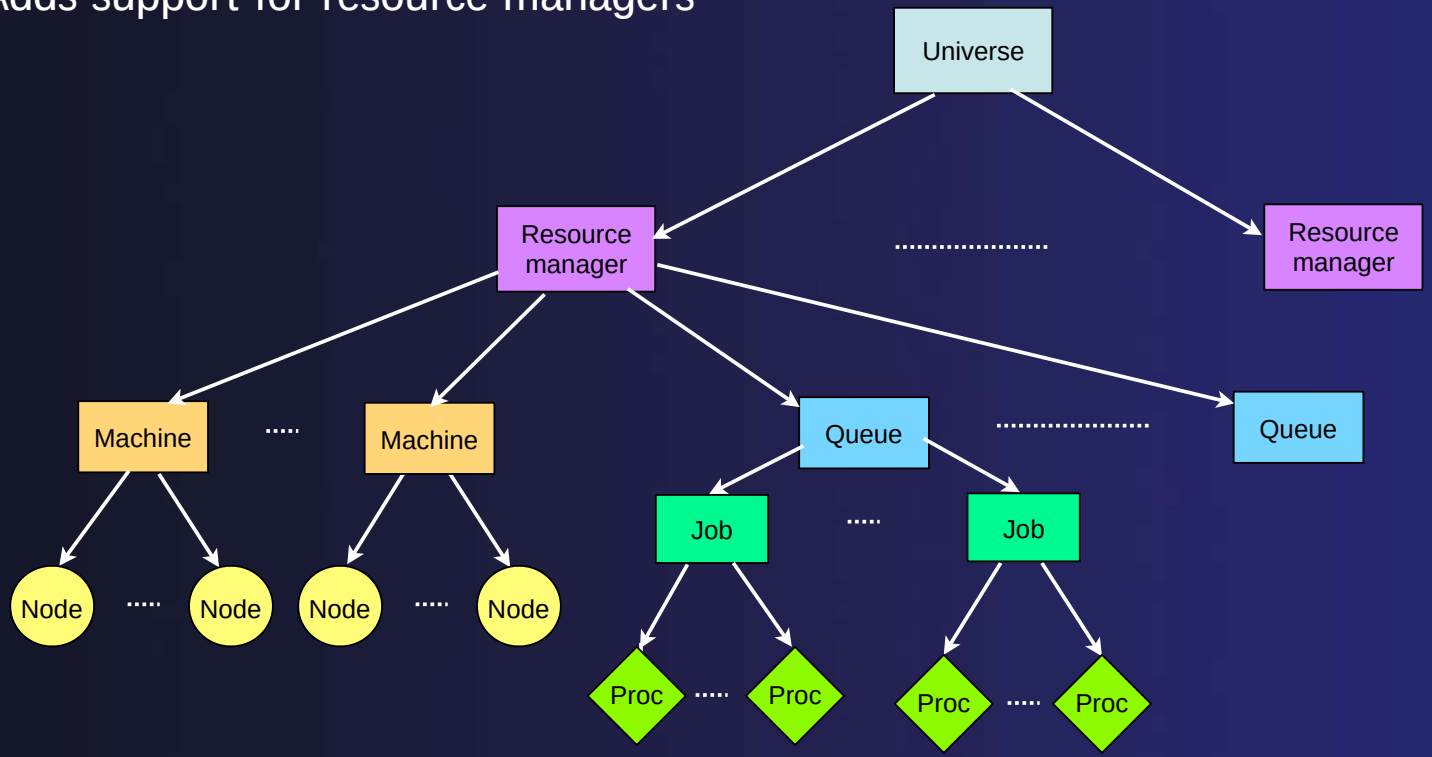
Parallel programming tools (cont...)



- Concurrency analysis shows which statements will be executed in parallel with highlighted statement
- Advanced error analysis detects if directives have been placed in incorrect locations

What's coming in PTP 2.0

- New parallel model elements
  - Adds support for resource managers



 ©2007 by Greg Watson; made available under the EPL 1.0

What's coming in PTP 2.0 (cont...)

- Job scheduler support
  - New resource manager system will allow job submission to multiple job schedulers from a single Eclipse session
  - Viewing of job status and job control will also be supported
  - Initial implementations for LSF, MOAB and SLURM
- Remote services
  - Allows Eclipse to run on local machine
  - Job submission, program launch, and program debugging on remote hosts

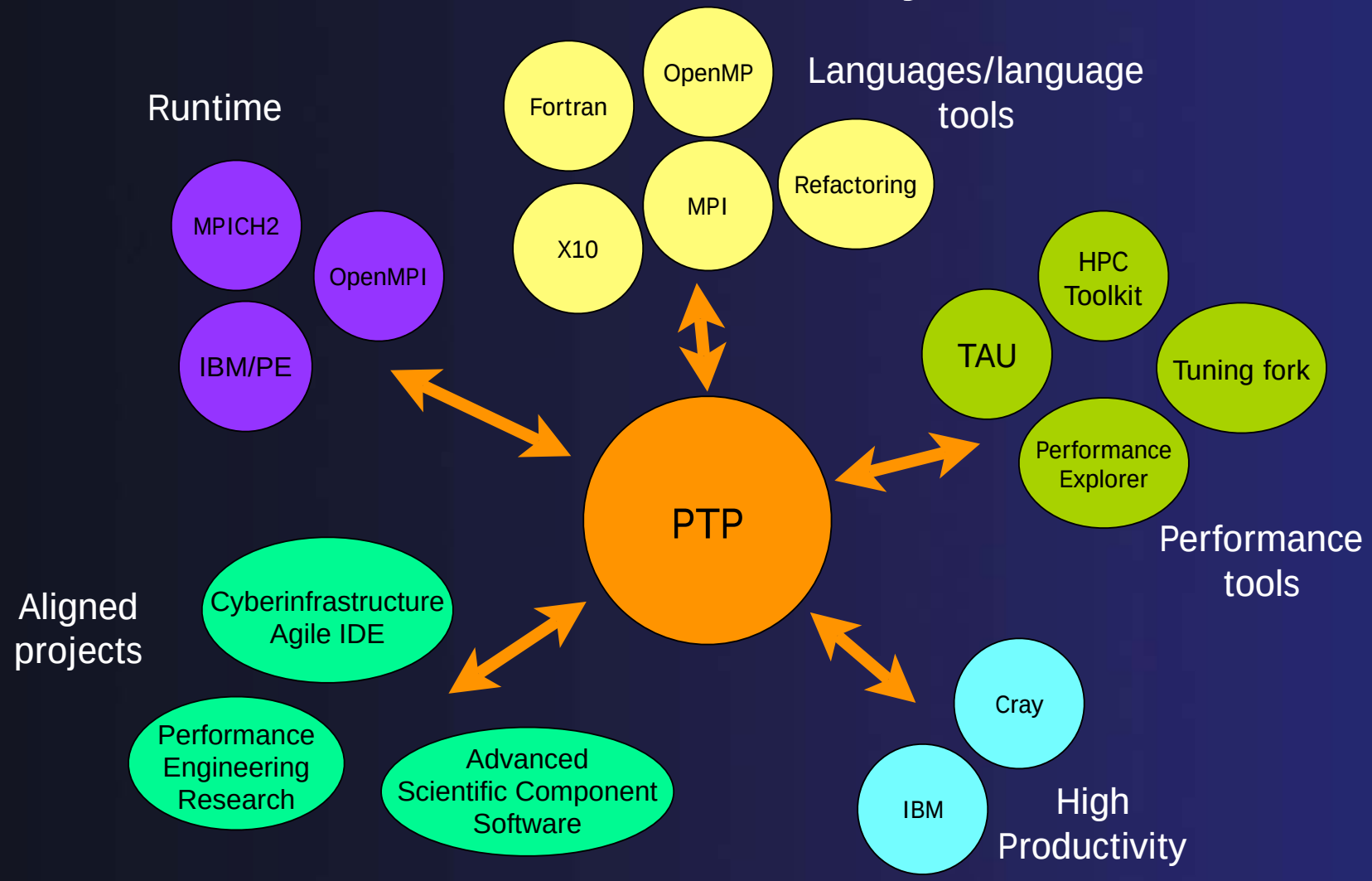
What's coming in PTP 2.0 (cont...)

- Parallel debugger enhancements
  - Scalability improvements
  - Support for non-gdb backend debuggers
  - New user interface features, including multi-variable viewer and array viewer
- Redesigned runtime system interface
  - Java-only will allow installation via software update
- Parallel language tool enhancements
  - MPI analysis and checking tools
  - Fortran support

What's coming in PTP 2.0 (cont...)

- Fortran 2003 support
  - Integration of Fortran parser with CDT DOM
  - Refactoring tools
- Parallel performance tools
  - General framework for integrating parallel performance tools

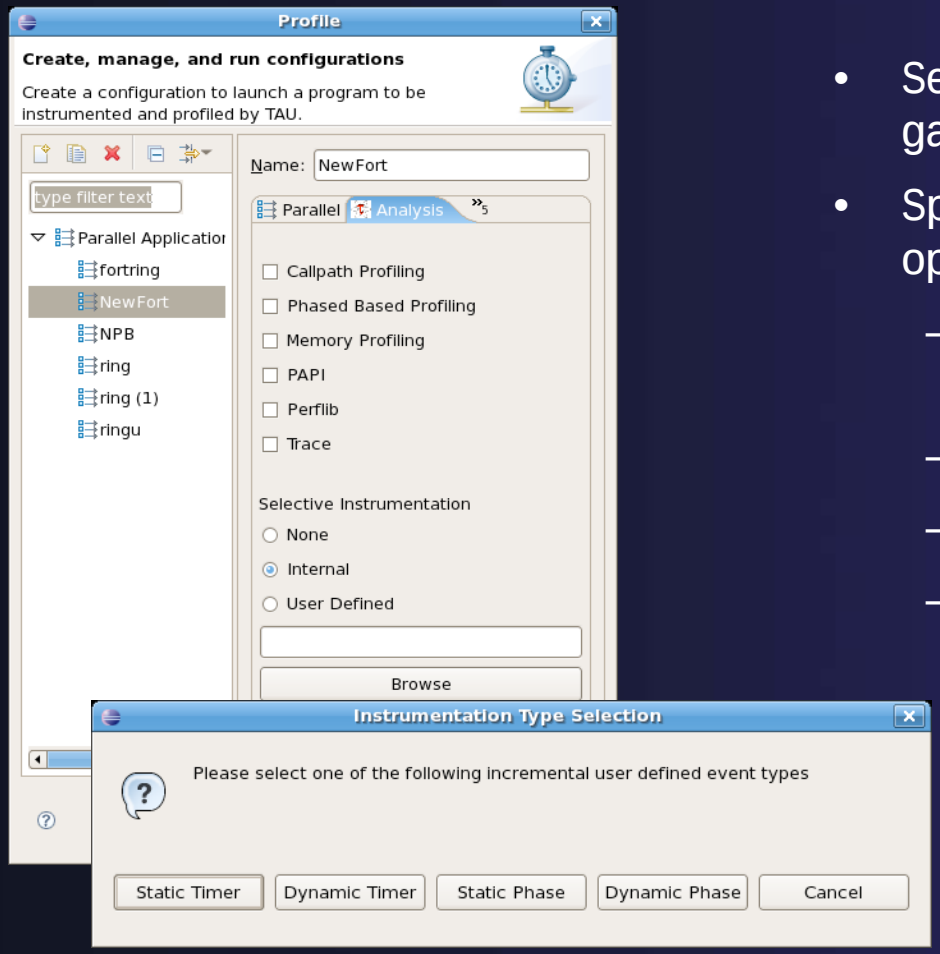
PTP community



Example of community growth

- Testing and Analysis Utilities (TAU) stage 1 integration
  - C/C++ and Fortran projects can be automatically instrumented and compiled with TAU libraries from within Eclipse
  - Performance data output is automatically stored in database
  - The Paraprof tool can be launched automatically to visualize profile output
- Generalizing framework to support other performance tools

Parallel performance framework



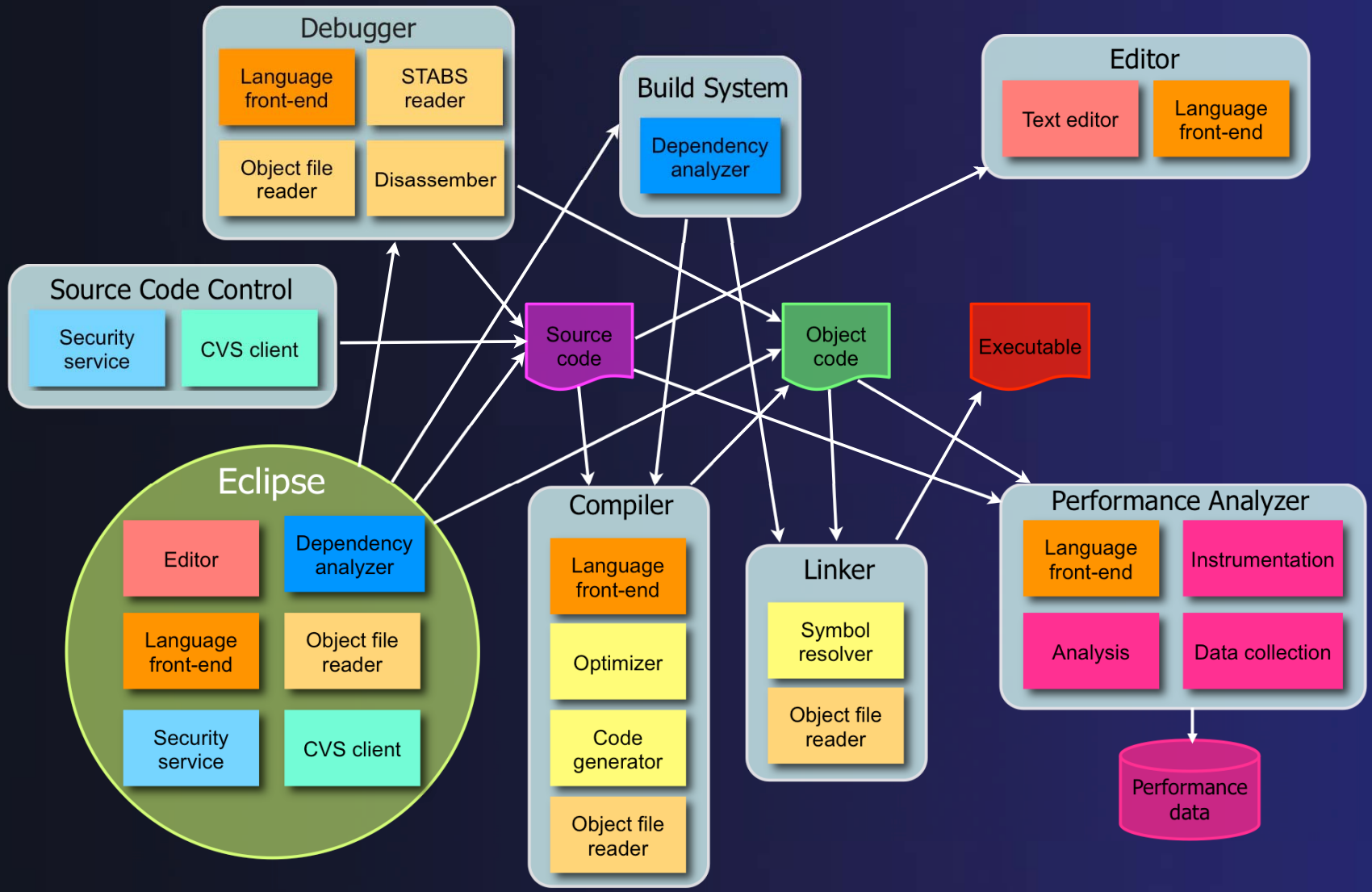
- Select performance data gathering and output options
- Specify selective instrumentation options
  - File and routine inclusion/ exclusion
  - Loop level instrumentation
  - Atomic events
  - Static and dynamic timers and phases

Future work

Trends in high performance computing

- Current HPC environment
  - CPU speeds stagnating
  - Power & cooling requirements are becoming dominant
  - Simple scaling no longer providing performance improvements
- HPC evolution
  - Heterogeneous multi-core architectures
  - Special purpose co-processing hardware
  - GPU's are becoming very attractive
  - Research underway into application specific hardware

Existing Tools Paradigm



Why is this a problem for HPC?

- Very complex machines
  - Require new tools and complex tool chains to just get programs to run
  - Even more difficult to get programs to run **fast**
- Monolithic tool design
  - Assumes relatively simple flow of operation
  - All data for the task is produced/consumed by tool
  - No opportunity to leverage other tools
- Unlikely that monolithic tools will suffice
  - A new paradigm is needed

New Tools Paradigm

Advantages

- Simplifies tool development
  - Reduces/eliminates duplicated functionality
  - Allows incremental development
  - Opportunity to unify common tool functionality
- Data sharing is the norm rather than the exception
- Complex tool chains can be automated
- Supports stand-alone and integrated tools
- Uniform interface

Conclusion

- PTP has demonstrated steady progress over the last 2 years
  - Is not yet self sustaining
- Community support and participation has continued to grow
  - Needs to expand user base
- PTP needs broader commercial tool vendor involvement
- Fortunately there is no viable alternative to Eclipse!

Resources

- PTP Project
  - http://eclipse.org/ptp
- OpenMPI
  - http://open-mpi.org
- MPICH2
  - http://www.mcs.anl.gov/mpi/mpich2
- OpenMP
  - http://www.openmp.org