Open MPI Developer's Workshop

April 17-20, 2006
Cisco Building 14
San Jose, CA USA

Open MPI Sponsors

- DoE
  - ASC
  - LANL CCS-1
  - NNSA
- HLRS
- Lilly Endowment
- Microsoft
- NSF

Workshop Sponsor

- Cisco Systems

Instructors

- Brian Barrett
  - Indiana University
- George Bosilca
  - University of Tennessee
- Rich Graham, Galen Shipman, Tim Woodall
  - Los Alamos National Laboratory
- Jeff Squyres
  - Cisco Systems

Logistics

- Building 14 lobby, meet at 7:45am
  - Missed us?
  - Jeff Lesniak: 831-247-1660
  - Jeff Squyres: 502-648-6714
- Breaks
  - Breakfasts, lunches
  - Morning and afternoon breaks

Logistics

- Bathrooms / break area
- Network access
- Cell phones
- Slides
- Cisco-sponsored dinner Tuesday
  - Fault Line Brewery
  - Information in your folder

Week Overview

- This is interactive
- Please interrupt us!
  - Questions, comments, etc.

Week Overview: Monday

- Background / project information
- Developer tools / perspective
- Code base
- Open MPI State of the Union
- Component / plugin system
- Portability layer
- (Optional) Next generation collectives

Week Overview: Tuesday

- Run-time environment
- MPI implementation fundamentals
  - Groups, communicators, datatypes, requests
- MPI-1 collectives
- MPI-1 topologies
- MPI-2 dynamics
- MPI-2 Parallel I/O
- Cisco-sponsored dinner

Week Overview: Wednesday

- Point-to-point frameworks / implementations
  - RDMA-based networks
  - Send/receive-based networks
  - Loopback device
- Multi-threading issues
- Memory management
- MPI-2 one-sided

Week Overview: Thursday

- Gil Bloch, Mellanox
  - Lessons learned – MPI on IB
- Patrick Geofray, Myricom
  - Lessons learned – MPI on Myrinet
- Spill over from anything else

Project Background

The Name

- Two words!
  - Open MPI
  - NOT "OpenMPI"
- Frequently abbreviated "OMPI"
  - Pronounced "oom-pee"
- It's a brand – let's try to get it right ☺

MPI From Scratch!

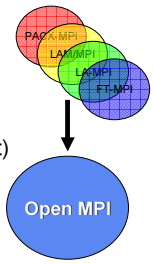
- Developers of FT-MPI, LA-MPI, LAM/MPI
  - Kept meeting at conferences in 2003
  - Culminated at SC 2003: Let's start over
  - Open MPI was born
- Started serious design and coding work January 2004
  - All of MPI-2 (initially skipped one-sided ops)
  - Demonstrated at SC 2004
  - Released at SC 2005

MPI From Scratch: Why?

- Each prior project had different strong points
  - Could not easily combine into one code base
- New concepts could not easily be accommodated in old code bases
- Easier to start over
  - Start with a blank sheet of paper
  - Decades of combined MPI implementation experience

MPI From Scratch: Why?

- Merger of ideas from
  - FT-MPI (U. of Tennessee)
  - LA-MPI (Los Alamos)
  - LAM/MPI (Indiana U.)
  - PACX-MPI (HLRS, U. Stuttgart)



What About the Prior Projects?

- All are in "maintenance" mode
  - Cannot abandon existing user bases
  - New releases (if any) for critical bug fixes
  - [Vast] Majority of time spent on Open MPI
- All major features being [slowly] rolled into Open MPI

Open MPI Members

- Founders
  - High Performance Computing Center, Stuttgart
  - Indiana University
  - Los Alamos National Laboratory
  - The University of Tennessee
- Recent additions
  - Cisco Systems
  - Mellanox Technologies
  - Sun Microsystems
  - University of Houston
  - Voltaire

Multi-Organization Collaboration

- Each organization:
  - Shares some common goals
  - Has non-overlapping / different goals
- …but that is ok!
  - In fact, this is what makes us strong
- Open MPI reflects the priorities of the current members
  - …and the membership just got larger

Project Goals

- Next generation MPI implementation
  - All of MPI-2
  - Reflect over a decade of MPI experience
- Prevent "forking" problem
  - Community / 3rd party involvement
  - Production-quality research platform
  - Rapid deployment for new platforms

Project Goals

- Open source
  - Vendor-friendly license (BSD)
  - Bring together "MPI-smart" developers
- Provide an MPI that "just works"
  - Make a user-friendly experience
- Portable performance
  - Support arbitrary combinations of back-end networks, platforms, run-time environments

Design Goals

- Extend / enhance previous ideas
  - Component architecture
  - Message fragmentation / reassembly
  - Design for heterogeneous environments
    - Multiple networks (run-time selection and striping)
    - Node architecture (data type representation)
  - Automatic error detection / retransmission
  - Process fault tolerance

Design Goals

- Design for a changing environment
  - Hardware failure
  - Resource changes
  - Application demand (dynamic processes)
- Portable efficiency on any parallel resource
  - Small cluster
  - "Big iron" hardware
  - "Grid" (everyone has a different definition)
  - …

Implementation Goals

- All of MPI-2
- Optimized performance
  - Low latency
  - High bandwidth
- Production quality
- Thread safety and concurrency (MPI_THREAD_MULTIPLE)

Implementation Goals

- Based on a component architecture
  - Flexible run-time tuning
  - "Plug-ins" for different capabilities (e.g., different networks)
- Natively support commodity networks
  - TCP
  - Shared memory
  - Myrinet
    - GM, MX
  - Infiniband
    - mVAPI, OpenIB
  - Portals

Operating Systems

- Current
  - Linux
  - OS X (BSD)
- Not frequently tested
  - Solaris
  - AIX
- Development
  - MS Window
- Maybe?
  - HP/UX, IRIX
- Majority of OMPI is POSIX C
  - Not difficult to port to new OS's
- Segregate OS-specific functionality
  - Plugins

Run-Time Environments

- Daemon and daemon-less modes
  - vs. LAM/MPI
- Current support
  - rsh / ssh
  - BProc (current)
  - PBS / Torque
  - SLURM
  - BJS (LANL BProc Clustermatic)
  - Yod (Red Storm)
- Future
  - SGE
  - LSF
  - BProc (Scyld)
  - RMS (Quadrics)
  - Grid ("multi-cell")
- Segregate RTE-specific functionality
  - Plugins

Legal Stuff

This is boring but necessary ☹

Bear with me…

IANAL

- I am not a lawyer
- This is not legal advice
- This is simply my non-legal-professional understanding
- I strongly encourage you to check with your own legal counsel

Intellectual Property

- Commit access requires legal paperwork
- We must have an IP-clean code base
  - Contribution agreements on web site
  - Modeled after Apache contribution agreements
- No copyright assignments
  - Just license contributed code to OMPI
  - Allow redistribution under BSD

Ownership

- Initial entire code base
  - Jointly developed and owned by 4 founders
  - IU, UTK, LANL, HLRS
  - So you'll see copyrights for all 4 in most files
- Since then, ownership is diverse
  - Asserted by copyright notices

Copyrights

- ***Not*** the same thing as licenses
- Copyright notices go in ***every*** file
- Rules of thumb
  - When in doubt, ask
  - Include more copyrights (vs. less)
  - If you edit a file, update your organization's copyright notice in that file

License

- Open MPI licensed under the BSD
  - Not GPL
- All contributed code must be compatible with BSD
  - Therefore, licenses do not go in source files
  - Top-level LICENSE file only
  - One license for all of Open MPI

Importing External Source

- Must be licensed properly
  - Compatible with BSD
  - GPL is not compatible with BSD
- Always include all relevant notices
  - Copyright(s) and license
  - Avoid someone later saying "you used my code; you owe me money"
- Examples
  - ptmalloc2, libevent, ROMIO

Patches

- "Small" patches do not require signed contribution agreements
  - Definition of "small" is relative and left up to common sense
  - Typos, small patches
  - Fixes to current functionality (not new functionality)
- "Large" patches do
  - New functionality (e.g., new components)

Legal / IP Questions

- When in doubt, ask
- When in doubt, ask
- When in doubt, ask

Open Source

Open Source Project

- The Open MPI code base is open source
  - Anyone *can* fork, but we discourage that
  - There are too many MPI's already
- Does not exclude closed source
  - Can distribute closed-source plugins
  - Do not need to distribute Open MPI itself

Community

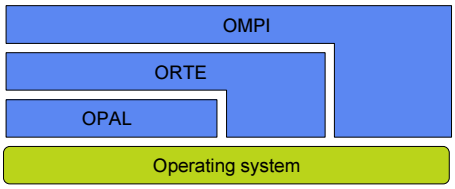
- Strong relationship with open source community
  - Open repository
  - Open mailing lists
  - Responsive to questions, problems
- Work with and for the HPC community

Top-Level Architecture

Three Main Code Sections

- Open MPI layer (OMPI)
  - Top-level MPI API and supporting logic
- Open Run-Time Environment (ORTE)
  - Interface to back-end run-time system
- Open Portability Access Layer (OPAL)
  - Utility code (lists, reference counting, etc.)
- Dependencies - not layers
  - OMPI → ORTE → OPAL
  - Strict abstraction barriers!

Three Main Code Sections



OPAL

- Lowest layer in Open MPI
- Much OS/system-system specific stuff
  - Assembly code
  - Processor / memory affinity
  - High-resolution timers
- "Glue" code
  - OBJ macros
  - Utility classes

ORTE

- Run-time environment support
  - Hook in to back-end resource managers, etc.
  - Process discovery, allocation, launch
  - I/O forwarding
  - Generally only provide functionality if back-end system does not
- General purpose registry
- Messaging (not high-performance)

OMPI

- All MPI semantics
  - Groups, communicators, datatypes, etc.
- Heavily optimized
  - Will be spending much of the workshop discussing this layer