

## MPI versions

## MPI History

- Standardization started (1992)
- MPI-1 completed (1.0) (May 1994)
- Clarifications (1.1) (June 1995)
- MPI-2 (started: 1995, finished: 1997)
- MPI-2 book 1999
- MPICH 1.2.4 – partial implementation of some MPI-2 features

## MPI-2

- Dynamic process creation and management
- One sided communications
- Extended collective operations
- Parallel I/O
- Miscellany

## MPICH2

- all-new implementation of MPI
- designed to support research into high-performance implementations of MPI-1 and MPI-2 functionality.
- MPICH2 includes support for one-side communication, dynamic processes, intercommunicator collective operations, and expanded MPI-IO functionality.
- Clusters consisting of both single-processor and SMP nodes are supported.
  - With the exception of users requiring the communication of heterogeneous data, strongly encourage everyone to consider switching to MPICH2.
  - Researchers interested in using using MPICH as a base for their research into MPI implementations should definitely use MPICH2.
- current version of MPICH2 is 1.0.3, released on November 23, 2005

## LAM

- Current version is **LAM 7.1.2 (March 12, 2006)**
- **expected to be among the last - if not the last - release of LAM/MPI.**
- **MPI-1 Support**
  - complete implementation of the MPI 1.2 standard,
- **MPI-2 Support**

LAM/MPI includes support for large portions of the MPI-2 standard. A partial list of the most commonly used features LAM supports is provided below:

  - Process Creation and Management
  - One-sided Communication (partial implementation)
  - MPI I/O (using ROMIO)
  - MPI-2 Miscellany:
    - mpiexec
    - Thread Support (MPI\_THREAD\_SINGLE - MPI\_THREAD\_SERIALIZED)
    - User functions at termination
    - Language interoperability
  - C++ Bindings

## LAM features

- **Checkpoint/Restart**
  - MPI applications running under LAM/MPI can be checkpointed to disk and restarted at a later time.
  - LAM requires a 3rd party single-process checkpoint/restart toolkit for actually checkpointing and restarting a single MPI process - LAM takes care of the parallel coordination.
  - Currently, the [Berkeley Labs Checkpoint/Restart](#) package (Linux only) is supported. The infrastructure allows for [easy addition](#) of new checkpoint/restart packages.
- **Fast Job Startup**
  - LAM/MPI utilizes a small, user-level daemon for process control, output forwarding, and out-of-band communication.
  - The user-level daemon is started at the beginning of a session using `lambboot`, which can use `rsh/ssh`, `TM` (OpenPBS / PBS Pro), `SLURM`, or `BProc` to remotely start the daemons.
  - Although the time for `lambboot` to execute can be long on large platforms using `rsh/ssh`, the start-up time is amortized as applications are executed - `mpirun` does not use `rsh/ssh`, instead using the LAM daemons.
  - Even for very large number of nodes, MPI application startup is on the order of a couple of seconds.

## LAM features

- **High Performance Communication**
  - LAM/MPI provides a number of options for MPI communication with very little overhead.
  - The TCP communication system provides near-TCP stack bandwidth and latency, even at Gigabit Ethernet speeds.
  - Two shared memory communication channels are available, each using TCP for remote-node communication. LAM/MPI 7.0 and later support Myrinet networks using the GM interface. Using Myrinet provides significantly higher bandwidth and lower latency than TCP. LAM/MPI 7.1 also provides support for high-speed, low-latency Infiniband networks using the Mellanox Verbs Interface (VAPI).
- **Run-time Tuning and RPI Selection**
  - LAM/MPI has always supported a wide number of tuning parameters. Unfortunately, most could only be set at compile time, leading to painful application tuning.
  - With LAM/MPI 7.0, almost every parameter in LAM can be altered at run-time -- environment variables or flags to `mpirun` -- make tuning much simpler.
  - The addition of LAM's System Services Interface (SSI) allows RPI (the network transport used for MPI point-to-point communications) selection to be made at run-time rather than compile-time. Rather than recompiling LAM 4 times to decide which transport gives best performance for an application, all that is required is a single flag to `mpirun`.

## LAM features

- **SMP-Aware Collectives**

The use of clusters of SMP machines is a growing trend in the clustering world. With LAM 7.0, many common collective operations are optimized to take advantage of the higher communication speed between processes on the same machine. When using the SMP-aware collectives, performance increases can be seen with little or no changes in user applications. Be sure to read the LAM User's Guide for important information on exploiting the full potential of the SMP-aware collectives.
- **Integration with PBS**

PBS (either [OpenPBS](#) or [PBS Pro](#)) provides scheduling services for many of the high performance clusters in service today. By using the PBS-specific boot mechanisms, LAM is able to provide process accounting and job cleanup to MPI applications. As an added bonus to MPI users, `lambboot` execution time is drastically reduced when compared to `rsh/ssh`.
- **Integration with BProc**

The BProc distributed process space provides a single process space for an entire cluster. It also provides a number of mechanisms for starting applications not available on the compute nodes of a cluster. LAM's BProc support supports booting under the BProc environment, even when LAM is not installed on the compute nodes -- LAM will automatically migrate the required support out to the compute nodes. MPI applications still must be available on all compute nodes (although the `-s` option to `mpirun` eliminates this requirement).

## LAM features

- **Globus Enabled**  
LAM 7.0 includes beta support for execution in the Globus Grid environment. Be sure to read the release notes in the User's Guide for important restrictions on your Globus environment.
- **Extensible Component Architecture**  
LAM 7.0 is the first LAM release to include the System Services Interface (SSI), providing an extensible component architecture for LAM/MPI. Currently, "drop-in" modules are supported for booting the LAM run-time environment, MPI collectives, Checkpoint/Restart, and MPI transport (RPI). Selection of a component is a run-time decision, allowing for user selection of the modules that provide the best performance for a specific application.
- **Easy Application Debugging**  
Debugging with LAM/MPI is easy. Support for parallel debuggers such as the Distributed Data Debugging Tool and the Etnus TotalView parallel debugger allows users straight-forward debugging of even the most complicated MPI applications. LAM is also capable of starting standard single-process debuggers for quick debugging of a subset of processes (see our FAQ). A number of tools, including XMPI are available for communication debugging and analysis.
- **Interoperable MPI**  
LAM implements much of the Interoperable MPI (IMPI) standard, intended to allow an MPI application to execute over multiple MPI implementations. Use of IMPI allows users to obtain the best performance possible, even in a heterogeneous environment.

## Open MPI

- project combining technologies and resources from several other projects (FT-MPI, LA-MPI, LAM/MPI, and PACX-MPI) in order to build the best MPI library available
  - Fault Tolerant MPI (FT-MPI) is a full 1.2 MPI specification implementation that provides process level fault tolerance at the MPI API level. FT-MPI is built upon the fault tolerant HARNES runtime system.
  - LA-MPI was an implementation of the Message Passing Interface (MPI) motivated by a growing need for fault tolerance at the software level in large high-performance computing (HPC) systems.

## Open MPI

- PACX-MPI (**PA**rallel **C**omputer **eX**tension) enables seamless running of MPI-conforming parallel application on a **Computational Grid**, such as a cluster of **High-Performance Computers** like **MPPs**, connected through high-speed networks or even the Internet.
- Communication between MPI processes internal to an MPP is done with the vendor MPI, while communication to other members of the Metacomputer is done via the connecting network.

## OpenMPI objective

- Create a free, open source, peer-reviewed, production-quality complete MPI-2 implementation.
- Provide extremely high, competitive performance (latency, bandwidth, ...pick your favorite metric).
- Directly involve the HPC community with external development and feedback (vendors, 3rd party researchers, users, etc.).
- Provide a stable platform for 3rd party research and commercial development.
- Help prevent the "forking problem" common to other MPI projects.
- Support a wide variety of HPC platforms and environments.

## Open MPI – PACX/MPI

- The characteristics of such systems show two different levels of quality of communication:
  - Usage of the **optimized** vendor-MPI library: Internal operations are handled using the vendor-MPI environment on each system. This allows to fully exploit the capacity of the underlying communication subsystem in a portable manner.
  - Usage of communication daemons: on each system in the metacomputer two daemons take care of communication between systems. This allows to bundle communication and to avoid to have thousands of open connections between processes. In addition it allows to handle security issues centrally. The daemon nodes are implemented as additional, **local MPI processes**. Therefore, no additional TCP-communication between the application nodes and the daemons is necessary, which would needlessly increase the communication latency.

---

DiSCoV KENT STATE 23 March 2006

## Open MPI features

- Full MPI-2 standards conformance
- Thread safety and concurrency
- Dynamic process spawning
- High performance on all platforms
- Reliable and fast job management
- Network and process fault tolerance
- Support data and network heterogeneity
- Single library supports all networks
- Run-time instrumentation
- Many job schedulers supported

---

DiSCoV KENT STATE 23 March 2006

## Open MPI features (ctd.)

- Many OS's supported (32 and 64 bit)
- Production quality software
- Portable and maintainable
- Tunable by installers and end-users
- Extensive user and installer guides
- Internationalized error messages
- Component-based design, documented APIs
- CPAN-like tool for component management
- Active, responsive mailing list
- Open source license based on the BSD license

---

DiSCoV KENT STATE 23 March 2006

## Open MPI contributors

- Indiana University (LAM/MPI)
- University of Tennessee (FT-MPI)
- Los Alamos National Laboratory (LA-MPI)
- Sandia National Laboratories
- High Performance Computing Center at Stuttgart

---

DiSCoV KENT STATE 23 March 2006

## Open MPI Release status

- Currently at stable release 1.0.1 and alpha 1.0.2
  - all aspects of MPI-1 and MPI-2 except the MPI-2 one-sided operations API
- One-sided operations are available at the head of development and will be released as part of v1.1
- Network support
  - TCP / ethernet
  - Shared memory
  - Loopback (send-to-self)
  - Myrinet / GM
  - Myrinet / MX
  - Infiniband / OpenIB
  - Infiniband / mVAPI
  - Portals