The Message Passing Interface (MPI):

The New MPI 5.0 - Now with ABI Included!

BoF@ISC25 of the MPI Forum

CONNECTING THE DOTS

ISC High Performance

JUNE 10 – 13, 2025 | HAMBURG, GERMANY

Moderator: Martin Schulz, TUM/LRZ (Chair of the MPI Forum)

Speakers: Jeff Hammond, NVIDIA

Claudia Blaas-Schenner, TU Wien

Ryan Grant, Queens University

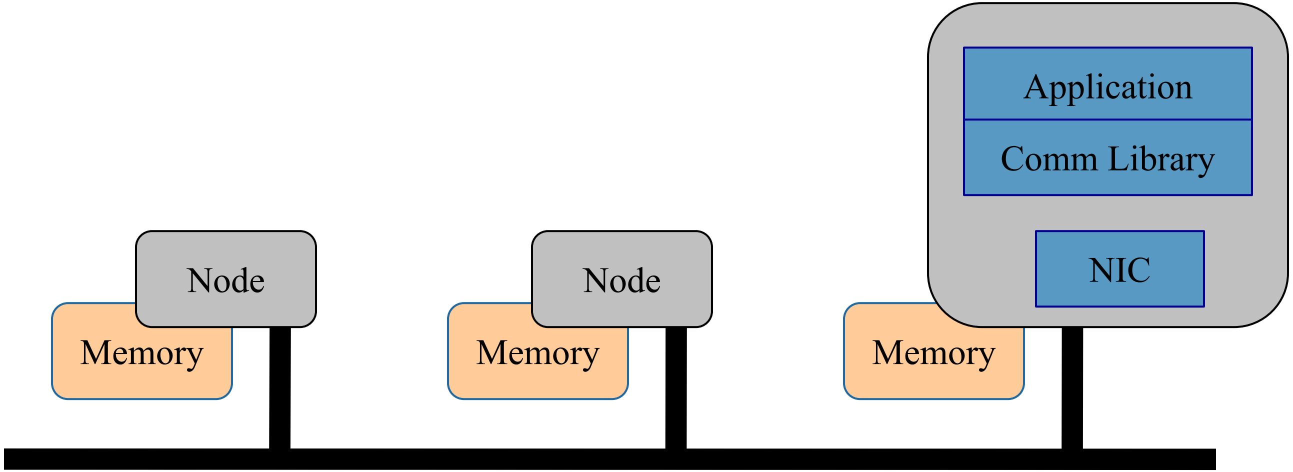
Marc-Andre Herrmanns, RWTH Aachen

MPI

The Message Passing Interface (MPI)

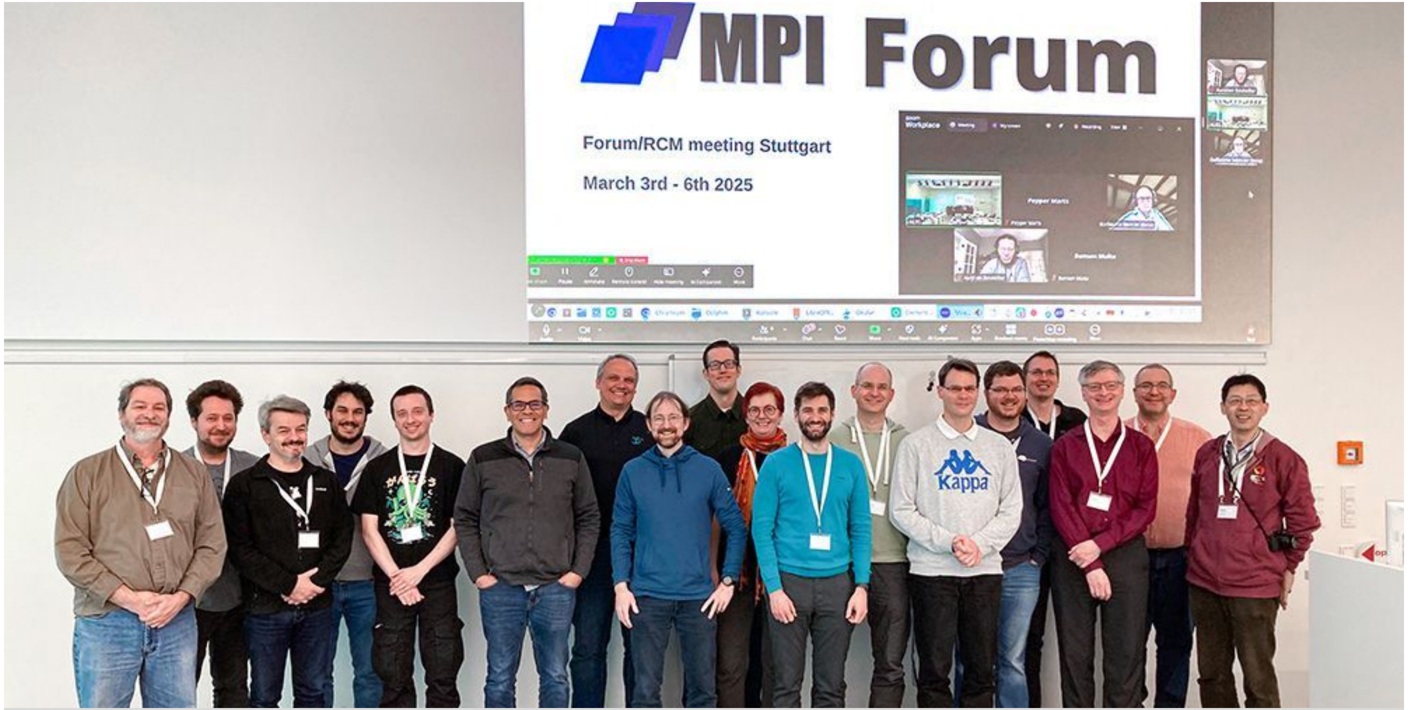
Designed in 1992, based on previous experiences with message passing libraries

- Based on the trend in the early 90ies towards shared memory architectures
- MPI 1.0 first ratified in 1994
- Started with simple point-to-point messaging and collectives
- Grew from there into broad functionality
- All documents at: http://www.mpi-forum.org/
- From the 25 year symposium in 2017: https://www.mcs.anl.gov/mpi-symposium/



MPI 5.0 Now Available!

On June 5th 2025 the MPI Forum ratified MPI 5.0



MPI Forum Meeting, March 2025 in Stuttgart @ HLRS

MPI 5.0 Now Available!

On June 5th 2025 the MPI Forum ratified MPI 5.0

Available at:
https://www.mpi-forum.org/docs/

Main new feature:
The MPI ABI

- + small updates
- + textual fixes

The MPI Forum Drives MPI

Standardization body for MPI

- Discusses additions and new directions
- Oversees the correctness and quality of the standard
- Represents MPI to the community
- Several working groups

Key Contacts: WG Chairs and Forum Officers

Application Binary Interface (ABI)
- Jeff Hammond and Lisandro Dalcin

Collective Communication, Topology, Communicators, Groups
- Tony Skjellum

Fault Tolerance
- Aurélien Bouteiller and Ignacio Laguna

Fortran
- Jeff Hammond, Purushotham Bangalore and Tony Skjellum

HW Topologies
- Guillaume Mercier

Hybrid and Accelerator Programming
- Jim Dinan

I/O
- Quincey Koziol

Languages
- Martin Ruefenacht

Remote Memory Access
- Joseph Schuchart

Sessions
- Howard Pritchard

Tools
- Marc-Andre Hermanns

MPI Forum Officers
- Chair: Martin Schulz
- Secretary: Wesley Bland
- Treasurer: Brian Smith
- Editor: Bill Gropp

The MPI Forum Drives MPI

Standardization body for MPI

- Discusses additions and new directions
- Oversees the correctness and quality of the standard
- Represents MPI to the community
- Several working groups

Open membership

- Any organization is welcome to participate
- Individuals have to “associate” themselves with one organization
- Voting rights depend on attendance
  - An organization has to be present two out of the last three meetings (incl. the current one) to be eligible to vote
- Votes are typically intended to be “close to unanimous”

The MPI Forum Drives MPI

Standardization body for MPI

- Discusses additions and new directions
- Oversees the correctness and quality of the standard
- Represents MPI to the community
- Several working groups

Open membership

- Any organization is welcome to participate
- Individuals have to "associate" themselves with one organization
- Voting rights depend on attendance
  - An organization has to be present two out of the last three meetings (incl. the current one) to be eligible to vote
- Votes are typically intended to be "close to unanimous"

Forum Meetings

- Typically 4x per year – 2x virtual and 2x hybrid (one with EuroMPI)
- Informal weekly meeting slot on Wednesday (as needed)
- Working group meetings organized per group

Join us:
www.mpi-forum.org

How Can You Participate?

1. Follow the MPI Forum website and git presence
   - Some parts are protected, don’t be shy to ask for access
2. Follow the MPI Forum email list(s)
   - Easy sign-up on the MPI Forum webpage
3. Provide feedback to the standard:
   - https://www.mpi-forum.org/comments/
4. Join a working group
   - All information on the website
   - Introduce yourself to the WG chair(s)
5. Introduce your own proposal to the WG
   - Start with discussions in the WG
   - Get feedback
   - Write concrete proposals
6. Volunteer for one of the chair positions

Join us:
www.mpi-forum.org

Why Should You Participate?

Why Should You Participate?

Centers/Users

Represent your user community

Support new features

Provide insights on usability

Catch wrong assumptions

Why Should You Participate?

Centers/Users

Represent your user community

Support new features

Provide insights on usability

Catch wrong assumptions

Drive development

Include innovations

Ensure portability

Ensure implementability

Develop prototypes

MPI Implementors

OPEN MPI

MPICH

MVAPICH

Why Should You Participate?

Centers/Users

Represent your user community

Support new features

Provide insights on usability

Catch wrong assumptions

Vendors

Ensure support for new hardware

Co-Design with SW developments

Help avoid mistakes

Understand your users

MPI

Drive development

Include innovations

Ensure portability

Ensure implementability

Develop prototypes

MPI Implementors

Open MPI

MPICH

MVAPICH

Why Should You Participate?

Centers/Users

- Represent your user community
- Support new features
- Provide insights on usability
- Catch wrong assumptions

Vendors

- Ensure support for new hardware
- Co-Design with SW developments
- Help avoid mistakes
- Understand your users

MPI

- Drive development
- Include innovations
- Ensure portability
- Ensure implementability
- Develop prototypes

MPI Implementors

OPEN MPI

MPICH

MVAPICH

- Develop new ideas and concepts
- Large community for feedback
- Ensure transition of research into long term practice
- Increase visibility of your work

HPC Researchers

The Message Passing Interface (MPI):

The New MPI 5.0 - Now with ABI Included!

CONNECTING THE DOTS

ISC High Performance

JUNE 10 – 13, 2025 | HAMBURG, GERMANY

Moderator: Martin Schulz, TUM/LRZ

Speakers: Jeff Hammond, NVIDIA
Claudia Blaas-Schenner, TU Wien
Ryan Grant, Queens University
Marc-Andre Herrmanns, RWTH Aachen

MPI

The Message Passing Interface (MPI):

The New MPI 5.0 - Now with ABI Included!

CONNECTING THE DOTS

ISC High Performance

JUNE 10 – 13, 2025 | HAMBURG, GERMANY

Jeff Hammond, NVIDIA

The new MPI ABI

MPI

MPI ABI Status Quo

MPI is an **API** standard, which defines the source code behavior in C (C++) and Fortran. The **compiled** representation of MPI features is implementation-defined.

If you **compile** with one of the following MPI families, you MUST **run** with the same.

1. MPICH / Intel MPI / MVAPICH / Cray MPI
2. Open MPI / NVIDIA HPC-X / Amazon MPI / IBM Spectrum MPI

Family 1 exists because there was a demand for interoperability with Intel MPI due to the prevalence of usage in ISV codes.

Family 2 is not guaranteed to be consistent, especially across major versions.

1 = https://www.mpich.org/abi/

API versus ABI

API

int MPI_Bcast(void * buffer, int count, MPI_Datatype d, int root, MPI_Comm c);

MPI_Datatype and MPI_Comm are unspecified types

ABI

typedef **struct ompi_datatype_t *** MPI_Datatype; // Open MPI family

typedef **int** MPI_Datatype; // MPICH family

Lots of other stuff like SO names, SO versioning, calling convention, etc.

https://dl.acm.org/doi/fullHtml/10.1145/3615318.3615319

Why?

Modern software use cases:

- Third-party **language** support, e.g. Python, Julia, Rust, etc.
- **Package** distribution, e.g. Spack, Apt, etc.
- **Tools** become implementation-agnostic
- **Containers**
- More efficient **testing** (build only once)

We can:

- Architectural reasons not to are gone
- Two platform ABIs cover >90% of HPC platforms

MPI ABI Packaging

- The header is abi/mpi.h
  - #include <mpi.h> still works - no code changes required to adopt ABI
  - The Forum should distribute a standard header for convenience
- The library is {lib}mpi_abi.ext
  - Implementations are instructed to use platform-specific SO versioning conventions
  - The Forum should distribute a standard SO for convenience
- The ABI is versioned independently from the API
  - ABI starts with 1.0
  - Backwards-compatible changes (e.g. new handle type) increment the minor version
  - Backwards-incompatible changes increment the major version

Now in MPI 5.0

- Single-feature ABI-only release. Chapter 20 is new. Appendix A is redone.
- Mukautuva, wi4mpi, and MPItrampoline can support this immediately.
- MPI ABI stubs repo: https://github.com/mpi-forum/mpi-abi-stubs
- MPICH has implemented the ABI already. Heavily tested by mpi4py.
- Open MPI is WIP: https://github.com/open-mpi/ompi/pull/13280

Diffusion: upstream -> release -> packaging, etc.

The Message Passing Interface (MPI):

The New MPI 5.0 - Now with ABI Included!

CONNECTING THE DOTS

ISC High Performance

JUNE 10 – 13, 2025 | HAMBURG, GERMANY

Ryan Grant, Queens University

Partitioned Communication

MPI

MPI Partitioned Communication: MPI 5.0 and Beyond

PRESENTER: DR. RYAN E. GRANT

STUDENT CREDIT: YILTAN TEMUCIN, AMIRREZA BARATI

COLLABORATORS: WHIT SCHONBEIN AND AHMAD AFSAHI

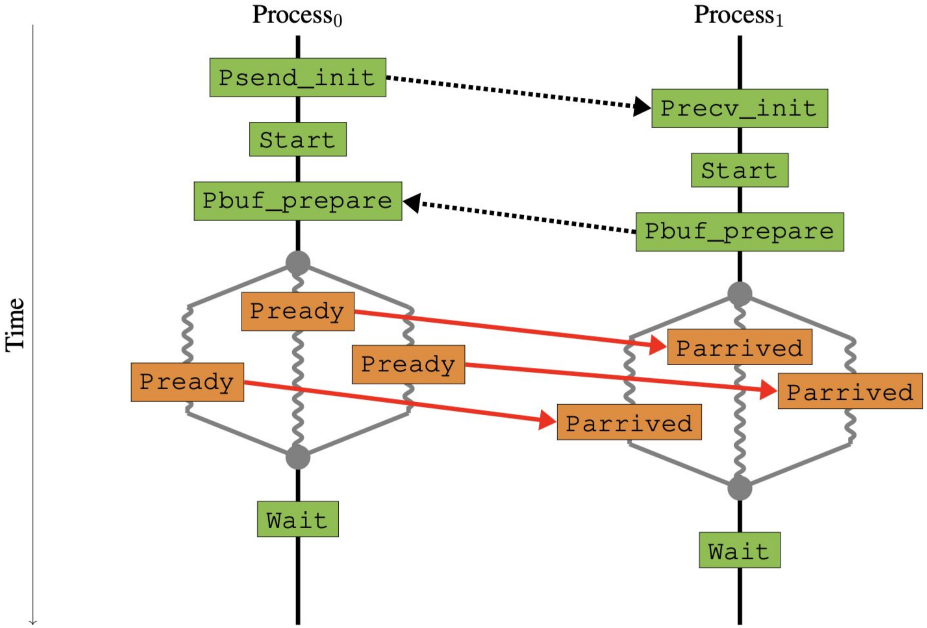
Intro to MPI Partitioned

- Decouple data movement from actors/threads from thread join/synchronization each communication
- Normal send/recv waits for threads to complete and then sends data
  - Why? Multi-threaded send/recv can have poor performance

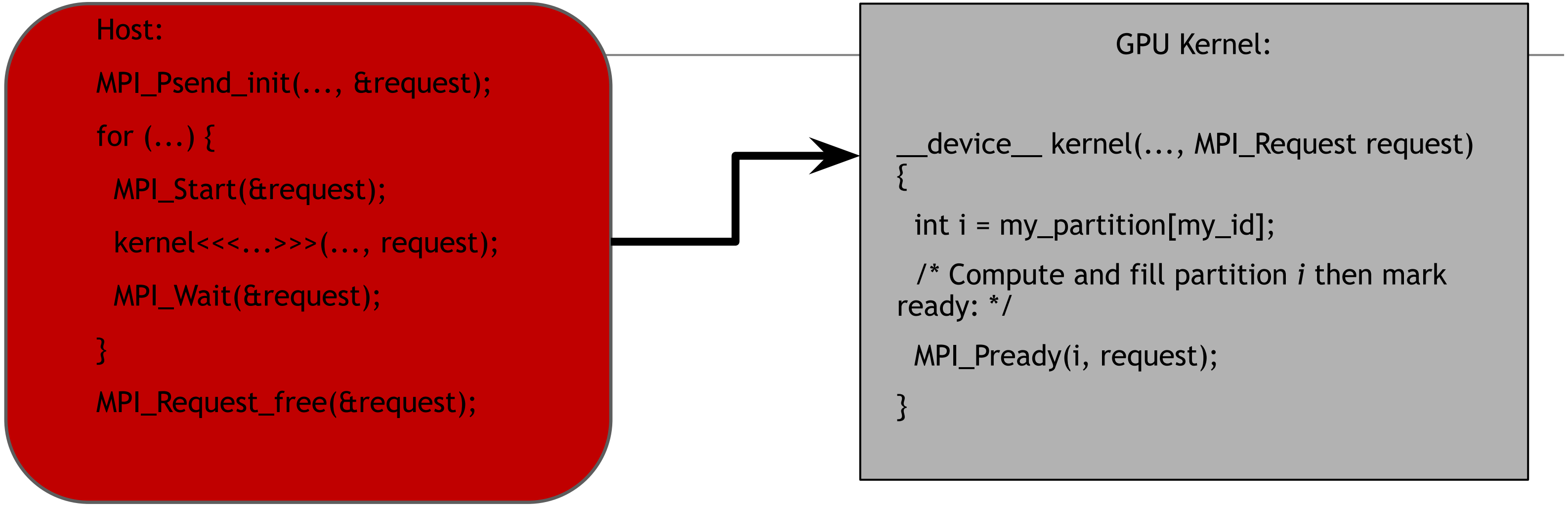
Basic Partitioned workflow

Actors (threads) call pready when their individual data becomes available to send

But how do we make sure the data proceeds in parallel?



Usage model - Kernel communication triggering



Host:

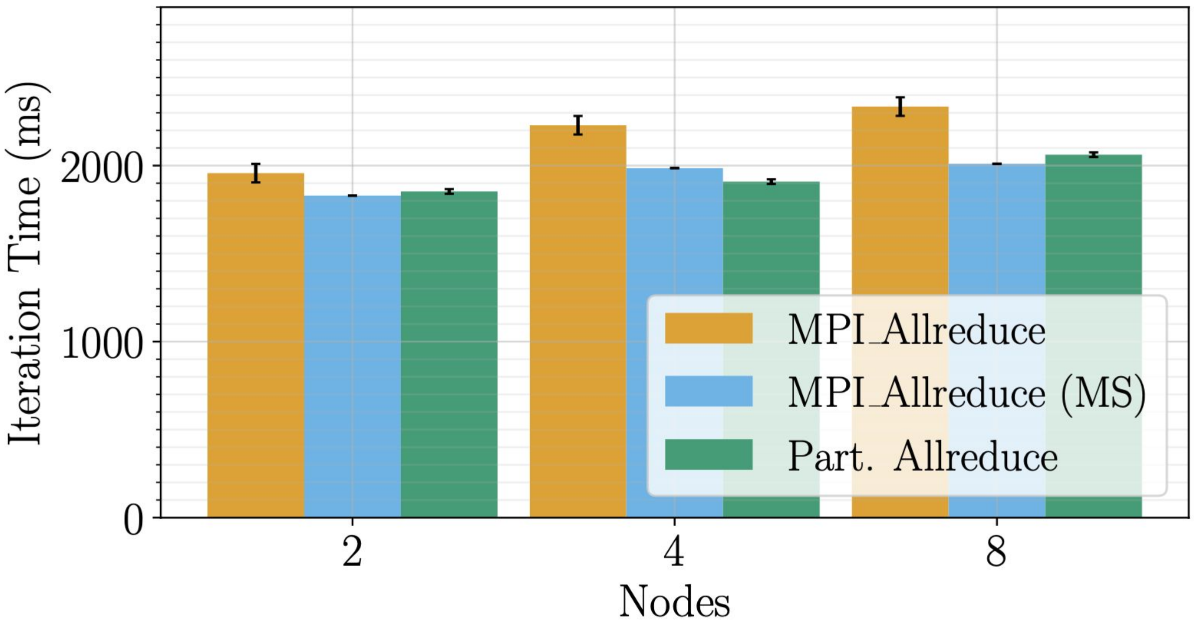
```
MPI_Psend_init(..., &request);
for (...) {
  MPI_Start(&request);
  kernel<<<...>>>(..., request);
  MPI_Wait(&request);
}
MPI_Request_free(&request);
```

GPU Kernel:

```
__device__ kernel(..., MPI_Request request)
{
  int i = my_partition[my_id];
  /* Compute and fill partition i then mark
ready: */
  MPI_Pready(i, request);
}
```

Note: CPU does communication setup and completion steps for MPI. Setup commands on NIC and poll for completion of entire operation. Kernel just indicates when NIC/MPI can send data. Ideally want to trigger communication from GPU to fire off when data is ready without communication setup/completion in kernel

Benefits training GPT – multipath with partitioned



Clearly using multiple paths makes performance better both hardware (blue) and software (green) approaches benefit over original allreduce

Note: hardware multi-spray can handle AI large volume traffic well

Takeaways

MPI partitioned communication is a great fit for multi-path networks

Need multiple send paths to make the most use of it

Results show 11.2% improvement over hardware multi-spray for pt2pt

Collectives also benefit with our approach at 3.05X vs 2.47X with hardware

The Message Passing Interface (MPI):

The New MPI 5.0 - Now with ABI Included!

CONNECTING THE DOTS

ISC High Performance

JUNE 10 – 13, 2025 | HAMBURG, GERMANY

Marc-Andre Herrmanns, RWTH Aachen

New MPI Tool Interfaces

MPI

Outlook on future tool interfaces

- QMPI
  - Successor of the PMPI interface
- Handle Introspection
  - Allow Debuggers interpret implementation specific data for handles
- MPI_T Unique Identifiers
  - Help matching MPI_T semantics across implementations
- MPI_T Entity Sets
  - Provide orientation for MPI implementors and tool developers

QMPI: next step for PMPI into the future

- Success of PMPI Interface
- Overcome PMPI limitations
  - allow for multiple tools to intercept calls to MPI at runtime
- Callback-driven
- User can influence interception order
- Similar in nature to PnMPI
- Status
  - Prototype available
  - Text drafted

Handle Introspection

- Generalized access to implementations-specific data
- Similar design to OMPD
  - Standarized API
  - Interface implemented by MPI library providers
- Allow for debuggers to rely on a standardized interface across MPI libraries
- MPI implementors also implement library to interpret/convert internal data to standardized data structures
- Status
  - Prototype in development
  - Interface drafted

MPI Tool Information Interface

Unique Identifiers

- Enable reliable identification of MPI_T entity semantics
  - Including updates/corrections to released semantics
- Support development of portable MPI_T tools
- Retain flexibility for MPI implementations to create or change behavior
- Status: API still in draft/discussion

Entity Sets

- Side-Document with specific definitions of one or more MPI_T entities
- Implementation/support remains optional
- Allow for definition of complex inter-entity relationships
- Status: List of entities in discussion

The Message Passing Interface (MPI):

The New MPI 5.0 - Now with ABI Included!

CONNECTING THE DOTS

ISC High Performance

JUNE 10 – 13, 2025 | HAMBURG, GERMANY

Discussion

New Directions for MPI 6.0

What is Next?

Implementations of the ABI available soon!

MPI Forum started working on MPI 6.0

- **Partitioned Communication**
- **New Tools Interfaces**

- **Support for Hybrid/Accelerated Computing**
  - Incl. bindings for GPUs
- **Dynamic resource management via MPI Sessions**
- **MPI Fault Tolerance**
- **Revamped support for MPI I/O and MPI RMA**
- **…**

MPI 5.0

We want to hear from you what you expect from MPI 6.0!