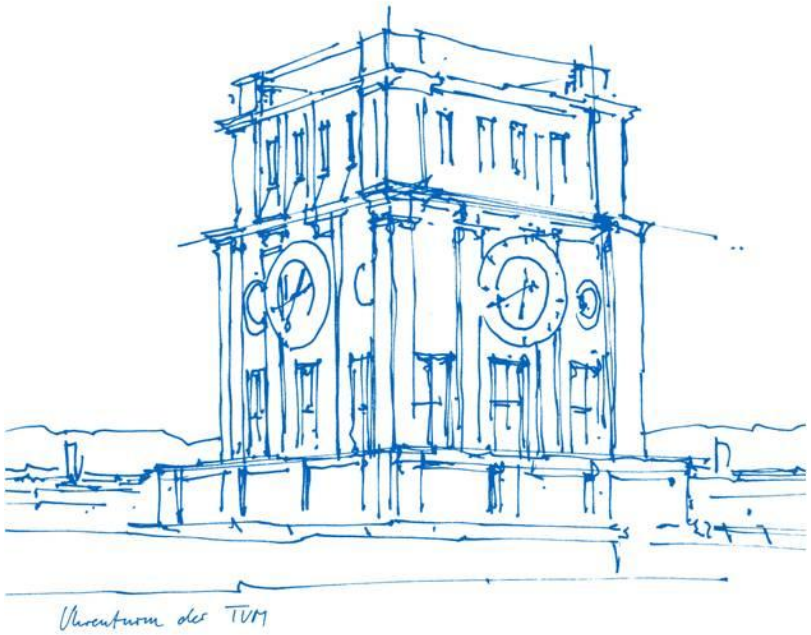
Dynamic Processes in MPI Sessions (Draft)

MPI Sessions WG

September 10th, 2025



Outline

- **Dynamic Resource Management: Definition and Motivation**
- Use Case Scenarios
- Limitations of current MPI specification
- Dynamic Processes with PSets

Dynamic Resource Management

- **Dynamically change resources (and number of procs) assigned to (MPI) applications at runtime**
- **Benefits:**
  - **Application Perspective:**
    - Tailor resources to (dynamically changing) resource requirements
    - Possibly enhance proactive fault tolerance mechanisms
  - **System Perspective:**
    - Improves scheduling flexibility and system metrics e.g. throughput & energy-efficiency
    - Address urgent/on-demand computing requirements
    - Enables reacting to system changes (e.g. carbon-efficiency, node failures, congestion, etc.)

Outline

- Dynamic Resource Management: Definition and Motivation
- **Use Case Scenarios**
- Limitations of current MPI specification
- Dynamic Processes with PSets

Use Case Scenarios: Scenario A (System Perspective)

Take away resources from jobs to meet external boundary conditions
Motivation:

- Short-notice power/energy cappings
- Avoid killing jobs

- **Requirements:**
  - Knowledge about jobs supporting shrinkage
  - Take away resources from Job A (without killing it)

Use Case Scenarios: Scenario B (System Perspective)

Swapping resources between applications

- **Motivation:**
  - Job B has higher efficiency then job A [1]
  - Improve locality of assigned resources (reduce latency, avoid Network Congestion, ...)
- **Requirements:**
  - Performance information of Job A and B
  - Take away resources from Job A
  - Assign resources to Job B

[1] M. Rodríguez-Gonzalo, D. E. Singh, J. G. Blas and J. Carretero, "Improving the Energy Efficiency of MPI Applications by Means of Malleability," *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, Heraklion, Greece, 2016, pp. 627-634, doi: 10.1109/PDP.2016.98.

Use Case Scenarios: Scenario C (App Perspective)

Adapting resources to varying application requirements

- **Motivation:**
  - Adaptive Mesh Refinement [1], In-Situ [2], Agent-based simulations [3], …
  - Phase-based applications, pilot-job workflows, …
- **Requirements:**
  - Job A indicates changing requirements to the system resource manager
  - Resource manager adds/removes resources to/from Job A

[1] Dominik Huber, Maximilian Streubel, Isaías Comprés, Martin Schulz, Martin Schreiber, and Howard Pritchard. 2022. **Towards Dynamic Resource Management with MPI Sessions and PMIx.** In Proceedings of the 29th European MPI Users' Group Meeting (EuroMPI/USA '22). Association for Computing Machinery, New York, NY, USA, 57–67.
[2] Ju, Y. *et al.* (2025). **Dynamic Resource Management for In-Situ Techniques Using MPI-Sessions.** In: Blaas-Schenner, C., Niethammer, C., Haas, T. (eds) Recent Advances in the Message Passing Interface. EuroPVM/MPI 2024. Lecture Notes in Computer Science, vol 15267. Springer, Cham.
[3] Singh DE, Olmedo Luceron C, Limia Sanchez A, Guzman Merino M, Duran Gonzalez C, Delgado-Sanz C, Gomez-Barroso D, Carretero J, Marinescu MC. **Evaluation of vaccination strategies for the metropolitan area of Madrid via agent-based simulation.** BMJ Open. 2022 Dec 9;12(12):e065937. doi: 10.1136/bmjopen-2022-065937.

Use Case Scenarios: Scenario D (App Perspective)

Changing resources in different application parts

- **Motivation:**
  - Coupled simulations, task-graphs / workflows, surrogate methods, …
- **Requirements:**
  - Job A indicates sub-job level structure to resource manager
  - Job A indicates requirements for sub-job level parts to resource manager
  - Resource manager adds/removes resources to/from application parts

Outline

- Dynamic Resource Management: Definition and Motivation
- Use Case Scenarios
- **Limitations of current MPI specification**
- Dynamic Processes with PSets

Motivation: Current lack of dynamicity in MPI

- **Current status in MPI (world model)**
  - Removing Processes:
    - MPI_Comm_disconnect (NOT possible on MPI_COMM_WORLD)
    - MPI_Abort (most implementations abort all processes in the job)
    - MPI_Finalize (is collective over MPI_COMM_WORLD)
  - Adding Processes:
    - MPI_Comm_spawn, MPI_Comm_accept/connect (blocking + no integration with RM)

Motivation: Current lack of dynamicity in MPI

- **Does not support Use Cases**
  - Scenario A+B: Can’t remove processes from MPI_COMM_WORLD
  - Scenario C: Spawn/Disconnect are on MPI communicator level (no feedback to RM)
  - Scenario D: RM does not know about different application parts

Motivation: Opportunities for dynamicity in MPI Sessions

- **Current status of MPI Sessions model**
  - MPI_Session_init is local
  - MPI_Session_finalize can be local if processes disconnected from all comms
  - No MPI_COMM_WORLD (processes are initially unconnected)
  - Processes can disconnect from all communicators
  - PSets represent sets of processes in the MPI runtime layer
  - Communicators can be created from PSets (via Groups)

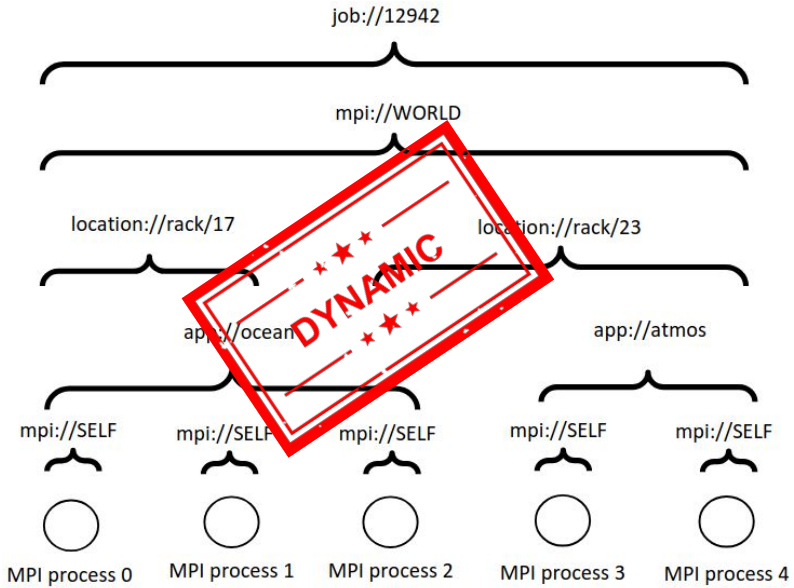
- So far no dynamic process management for MPI Sessions defined

Outline

- Dynamic Resource Management: Definition and Motivation
- Use Case Scenarios
- Limitations of current MPI specification
- **Dynamic Processes with PSets**

Introducing dynamicity to MPI Session Psets

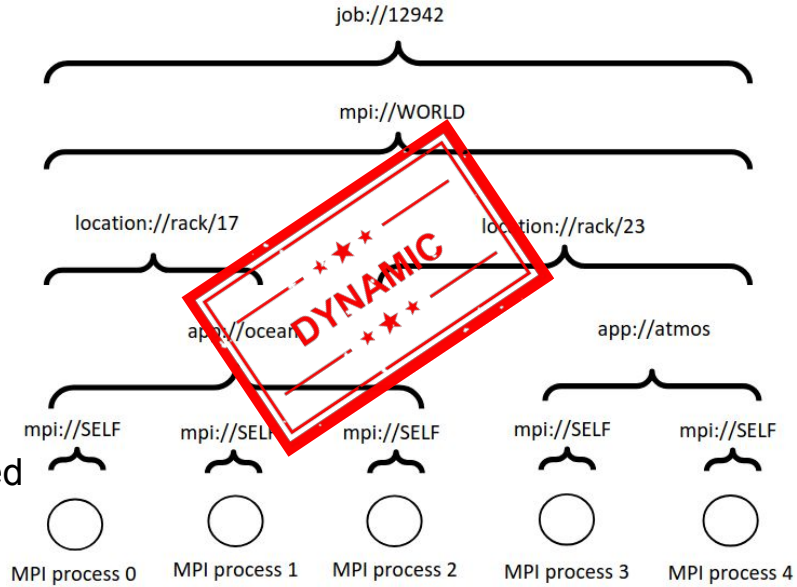
- **Goals:**
  - Cover diverse application types/patterns
  - Enable system-wide optimization by RM
  - Generic (beyond MPI)
- **Approach:**
  - Graph-based abstraction
    - Vertices = PSets
    - (Hyper)Edges = PSetOps
  - Cooperative optimization



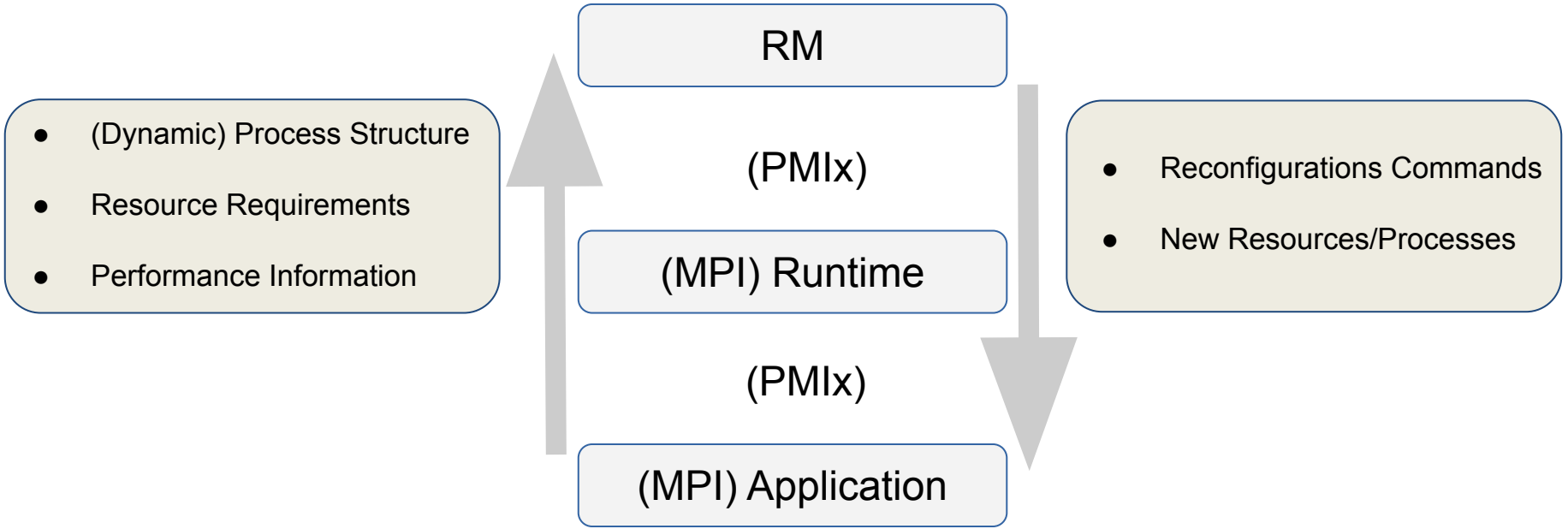
Introducing dynamicity to MPI Session Psets

- **Current Definition of PSets:**
  - Query list of PSet names (_get_nth_pset)
  - Contains only PSets where proc is included
  - List can only be extended

- **Extended Definition of PSets:**
  - PSets can exist beyond the list
  - Procs can use PSets where they are not included

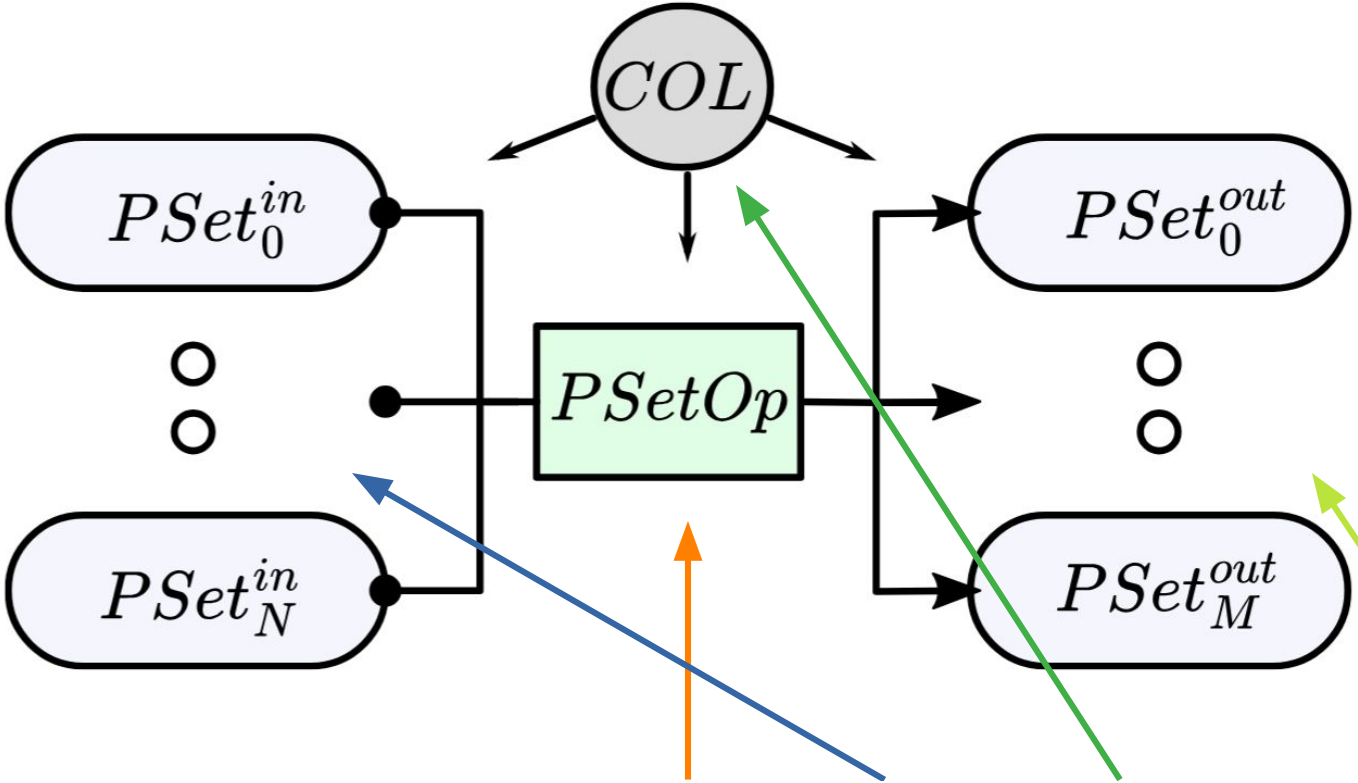


App-RM interaction across HPC software stack

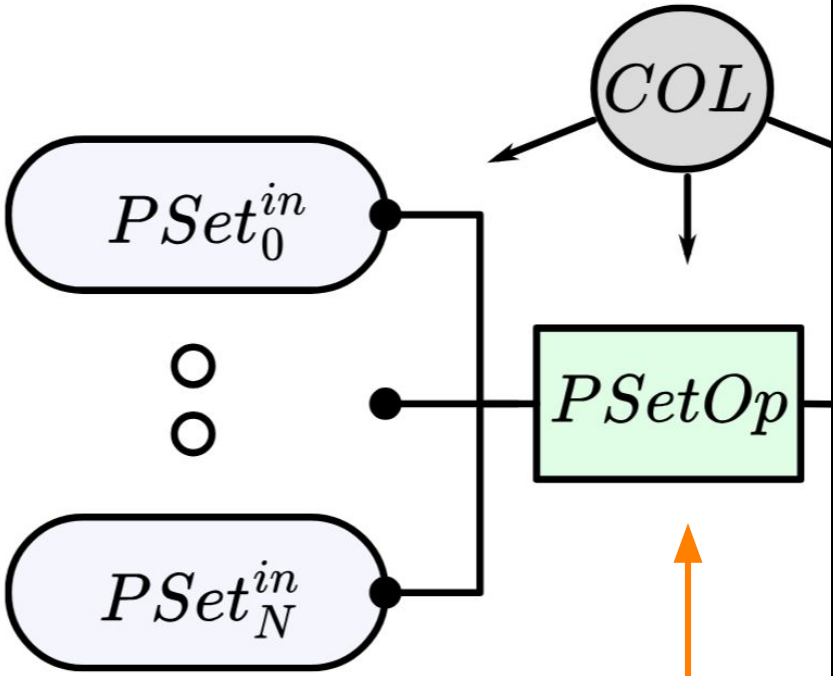


MPI_SESSION_IPSETOP(session, pset_op, input_psets, col_info, output_psets, output_size, retained, request)

IN	session	MPI Session handle (handle)
INOUT	pset_op	The PSet Operation (handle)
IN	input_psets	The list of input PSets of the operation (string)
IN	col_info	The optimization information expressed with the COL (handle)
OUT	output_psets	The list of output PSets of the operation (string)
IN	output_size	Maximum possible size of the output_psets string. '-1' indicates the buffer to be allocated by the MPI. (integer)
IN	retained	Flag indicating whether this setop is retained for repeated executions. (logical)
INOUT	request	Request to query status of operation (handle)



MPI_SESSION_IPSETOP(session, pset_op, input_psets, col_info, output_psets, output_size, retained, request)

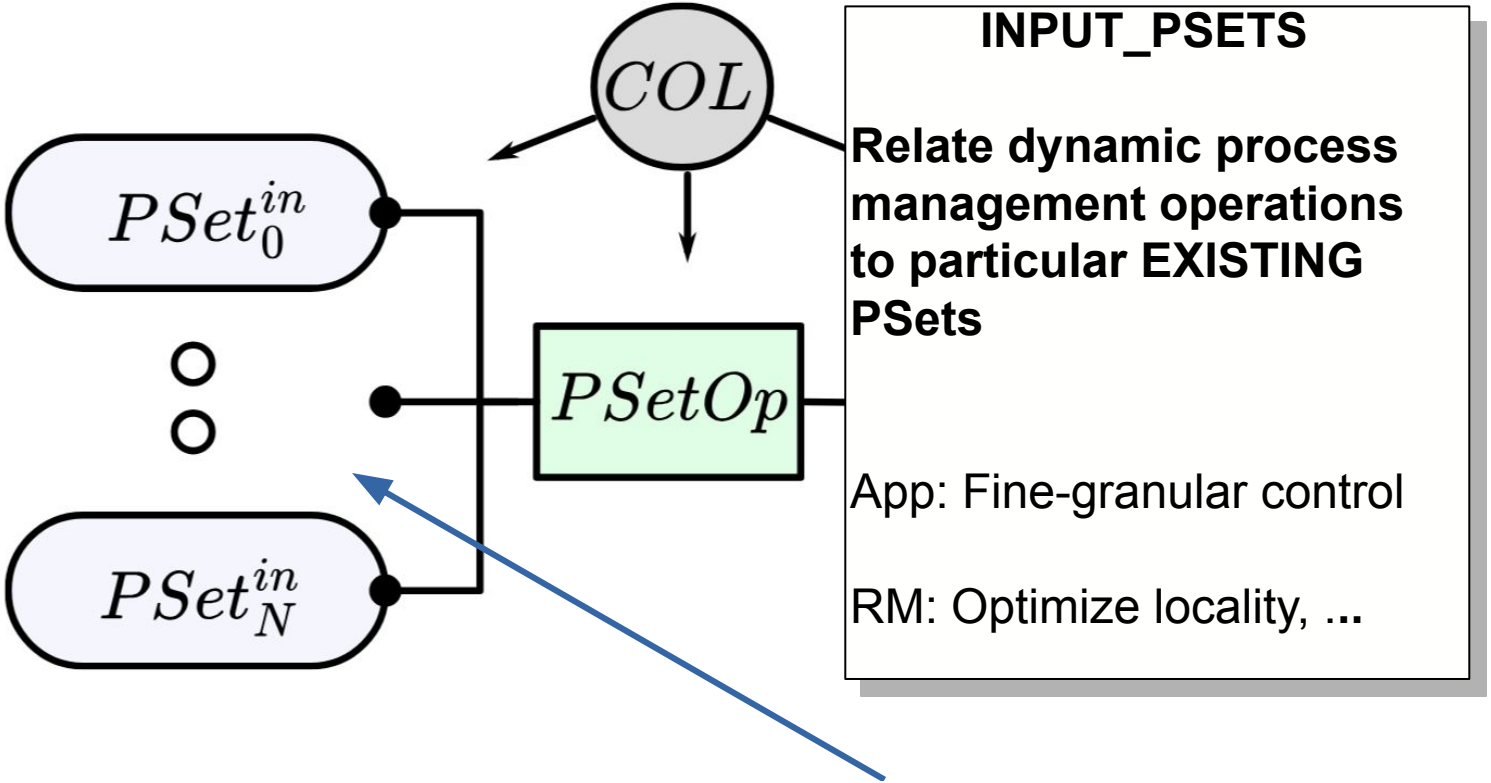


PSETOP

Examples:
- ADD: Task launch
- SUB: Task termination
- SPLIT: Sub tasks
- UNION: Merge tasks
- GROINK: Resize Task
- …

Expose application patterns as generic dynamic process management operations to the RM

MPI_SESSION_IPSETOP(session, pset_op, input_psets, col_info, output_psets, output_size, retained, request)

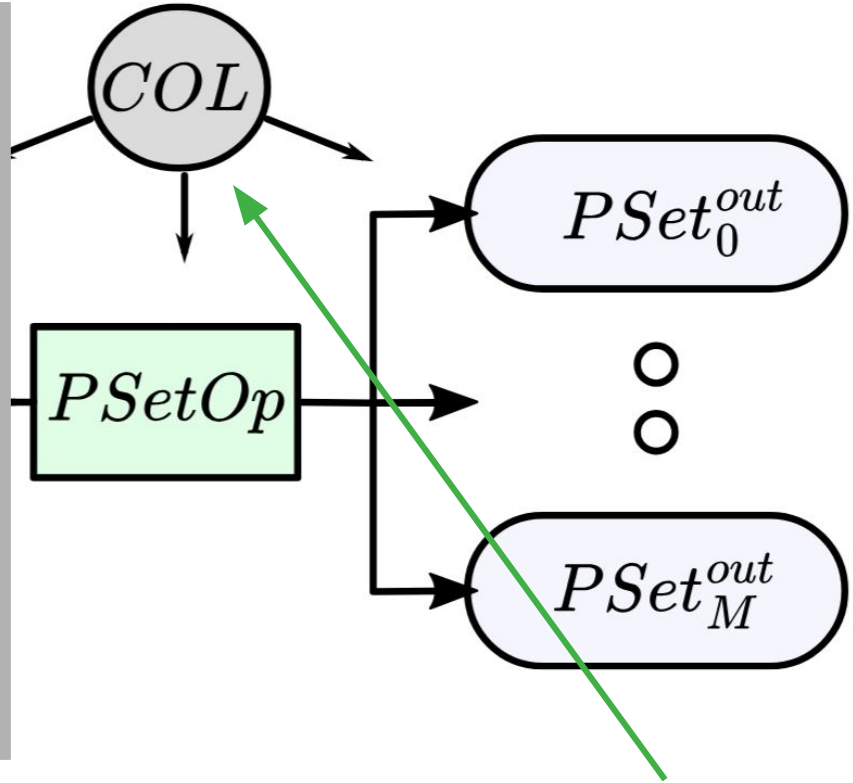


MPI_SESSION_IPSETOP(session, pset_op, input_psets, col_info, output_psets, output_size, retained, request)

COL (Collaborative Optimization Language)

Info for RM to decide how to instantiate output Psets:
- Performance Models
- Constraints
- Possible Mappings
- Executable/Environment?
  (= MPI_Comm_spawn info)

→ Different versions/ levels-of-complexity possible



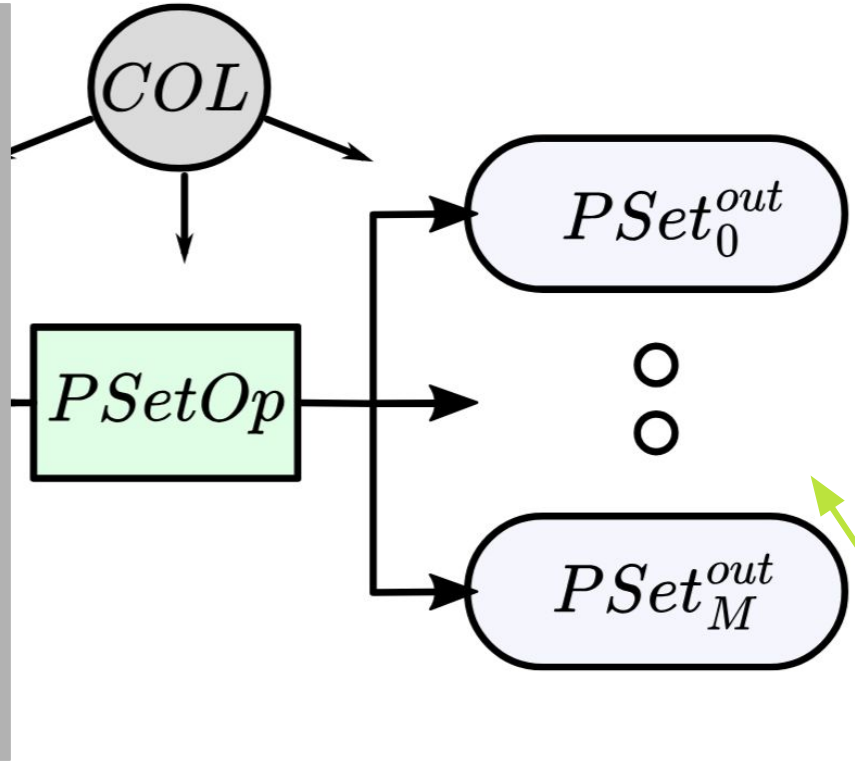
MPI_SESSION_IPSETOP(session, pset_op, input_psets, col_info, output_psets, output_size, retained, request)

OUTPUT_PSETS

The concrete RM decision of process reconfiguration

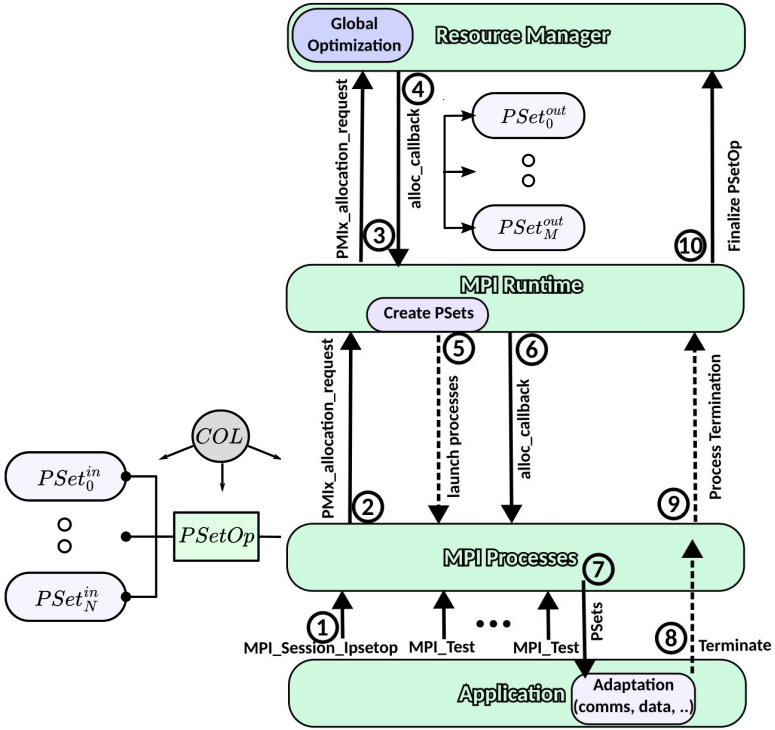
- Concrete association between Psets/Procs and system resources

- App needs to adapt accordingly (include new procs, terminate procs, data redistribution)



MPI_SESSION_IPSETOP(session, pset_op, input_psets, col_info, output_psets, output_size, persistent, request)

App-RM interaction across HPC software stack using DPP



Questions

- Is it easy enough to use?
- Are there any unsupported use cases?
- COL (Collaborative Optimization Language)
  - first version in MPI Standard or in supplementary document?
  - binary representation?
- What is the “MPI way” of returning a variable-length list of strings?
- …

Contact: domi.huber@tum.de

Related Publications

Dynamic Processes with PSets:

- Dominik Huber et al. (2025) **Dynamic Resource Management in HPC Systems Using Dynamic Processes with PSets.** (under review.)
- Zafer Bora Yılmazer, Dominik Huber, Arjun Parab, Amir Raoofy and Josef Weidendorfer, **Malleability in LAIK with MPI Dynamic Processes and PSets,** DynResHPC'25 workshop at EuroPAR 2025, (accepted)
- D. Huber, S. Iserte, M. Schreiber, A. J. Peña and M. Schulz, **Bridging the Gap Between Genericity and Programmability of Dynamic Resources in HPC**, *ISC High Performance 2025 Research Paper Proceedings (40th International Conference)*, Hamburg, Germany, 2025, pp. 1-11. https://ieeexplore.ieee.org/document/11018304
- Jonas Posner, Tim Ellersiek, Nick Bietendorf, Dominik Huber**,** Martin Schreiber, Martin Schulz**. (2025) Dynamic Resource Management: Comparison of Asynchronous Many-Task (AMT) and Dynamic Processes with PSets (DPP).** Asynchronous Many-Task Systems and Applications. WAMTA 2025. https://zenodo.org/records/14902214
- Yi Ju, Dominik Huber, Adalberto Perez, Philipp Ulbl, Stefano Markidis, Philipp Schlatter, Martin Schulz, Martin Schreiber, and Erwin Laure. (2025)**. Dynamic Resource Management for In-Situ Techniques Using MPI-Sessions**. In Recent Advances in the Message Passing Interface: 31st European MPI Users' Group Meeting, EuroMPI 2024, Perth, WA, Australia, September 25–27, 2024, Proceedings. Springer-Verlag, Berlin, Heidelberg, 105–120. https://doi.org/10.1007/978-3-031-73370-3_7
- Dominik Huber, Martin Schreiber, Martin Schulz, Howard Pritchard, Daniel Holmes. (2024) **Design Principles of Dynamic Resource Management for High-Performance Parallel Programming Models.** https://arxiv.org/abs/2403.17107
- Huber, D., Schreiber, M., Schulz, M. (2023). **A Case Study on PMIx-Usage for Dynamic Resource Management.** In: Bienz, A., Weiland, M., Baboulin, M., Kruse, C. (eds) High Performance Computing. ISC High Performance 2023. Lecture Notes in Computer Science, vol 13999. Springer, Cham. https://doi.org/10.1007/978-3-031-40843-4_4
- Dominik Huber, Maximilian Streubel, Isaías Comprés, Martin Schulz, Martin Schreiber, and Howard Pritchard. (2022) **Towards Dynamic Resource Management with MPI Sessions and PMIx.** Proceedings of the 29th European MPI Users' Group Meeting (EuroMPI/USA'22). Association for Computing Machinery, New York, NY, USA, 57–67. https://doi.org/10.1145/3555819.3555856
- Jan Fecht, Martin Schreiber, Martin Schulz, Howard Pritchard, and Daniel J. Holmes. 2022. **An Emulation Layer for Dynamic Resources with MPI Sessions.** In High Performance Computing. ISC High Performance 2022 International Workshops: Hamburg, Germany, May 29 – June 2, 2022, Revised Selected Papers. Springer-Verlag, Berlin, Heidelberg, 147–161. https://doi.org/10.1007/978-3-031-23220-6_10

State-of-the-art survey:

- A. Tarraf *et al*. (2024) **Malleability in Modern HPC Systems: Current Experiences, Challenges, and Future Opportunities.** In: *IEEE Transactions on Parallel and Distributed Systems*, vol. 35, no. 9, pp. 1551-1564, Sept. 2024. https://doi.org/10.1109/TPDS.2024.3406764

Backup Slides

PMIx Attributes to support PSet Operations

Allocation Directive:

```
PMIX_ALLOC_SETOP (pmix_alloc_directive_t) // Allocation request contains a Set
                                           Operation to be considered by the
                                            Resource Manager
```

Allocation Attributes:

```
PMIX_PSETOP_TYPE   “pmix.psetop.type”   (pmix_psetop_type_t) // The type of the
                                                               PSetOp
PMIX_PSETOP_INPUT  “pmix.psetop.input” (char *) // Comma-separated list of input
                                                  Pset names
PMIX_PSETOP_OUTPUT  “pmix.psetop.output” (char *) // Comma-separated list of
                                                    output PSet names
PMIX_PSETOP_COL    “pmix.psetop.col”      (tbd)  // COL object
```

Event Notification Codes: (used e.g. by RTE to notify RM when PSetOp finished)

```
PMIX_PSETOP_FINALIZED      -4242         // PSet Operation has been finalized
                                            (all procs started/terminated)
```

PMIx Attributes to support PSet Operations

PSet operation types (examples):

```
PMIX_PSETOP_NULL        (pmix_psetop_type_t) // Invalid PSet Operation
PMIX_PSETOP_ADD         (pmix_psetop_type_t) // Add a PSet with new processes
PMIX_PSETOP_SUB         (pmix_psetop_type_t) // Release processes in PSet
PMIX_PSETOP_GROINK      (pmix_psetop_type_t) // Add Grow/Shrink version of PSet
...
```