We must therefore expect that using combinations of different APIs at different system levels will be the only practical solution in the short to medium term. Although there remains room for improvement in individual programming models and their implementations, the main challenges lie in interoperability between APIs at the specification and implementation levels. In addition to interoperability among APIs, INTERTWinE tackles interoperability on a more general level with the help of the Directory/Cache service and the Resource Manager.

The main goal of the INTERTWinE Resource Manager is to coordinate access to CPU resources between different runtime systems and APIs to avoid both oversubscription and undersubscription situations. We have developed three APIs:

- **Interoperability API**: to invoke parallel kernels (e.g. a piece of code written in OpenMP, OmpSs, or StarPU) into a specific set of CPUs from one runtime system to another
- **A dynamic resource sharing API**: to transparently lend and borrow CPUs between parallel runtime systems to avoid utilization scenarios.
- **A task pause/resume API**: to resume the interoperability-based task with blocking message passing APIs such as MPI. Targeted API combinations: OpenMP/OmpSs, StarPU/

Resource Manager

Parallel Programming Models

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
<th>APIs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MPI</strong></td>
<td>Message-passing API, widely used for distributed memory parallel programming</td>
<td>Interoperability of MPI plus threads, Interaction with non-MPI components other than OS/like threads is implementation-dependent, Needed to strike a balance between thread safety and performance optimization. Solution: MPI endpoints, which communicates with a common interface with multiple tasks for each MPI process and MPI thread.</td>
</tr>
<tr>
<td><strong>GasPI</strong></td>
<td>Defines asynchronous, single-sided, and non-blocking communication primitives for a Partitioned Global Address Space (PGAS)</td>
<td>Interoperability of GasPI plus MPI, Allows for increased porting of existing MPI applications, Copies the parallel environment during its initialization, so keeping existing toolchains, including distribution and installation of the libraries, Allows to access data that were allocated in the MPI program without additional copy.</td>
</tr>
<tr>
<td><strong>OpenMP</strong></td>
<td>Parallel application program interface targeting Symmetric Multiprocessor systems</td>
<td>Interoperability with message passing libraries such as MPI and GasPI, Improves the performance and programmability of hybrid MPI/GASPI + OmpSs applications by coordinating task scheduling with the message passing engine.</td>
</tr>
<tr>
<td><strong>StarPU</strong></td>
<td>Runtime system that enables programmers to exploit CPUs and accelerator units available on a machine</td>
<td>Avoid oversubscription and undersubscription scenarios.</td>
</tr>
<tr>
<td><strong>PaRSEC</strong></td>
<td>Generic framework for architecture-aware scheduling and management of micro-tasks on distributed many-core heterogeneous architectures.</td>
<td>Interoperability of PaRSEC and OpenMP, Combination of PaRSEC-based codes (e.g. DPLASMA) with OpenMP applications (e.g. PLASMA) on nodes.</td>
</tr>
<tr>
<td><strong>TAU</strong></td>
<td>📢 Description: 📢 Computation with large-scale graphs (combinatorial computing) is crucial for Big Data analytics. While graph computations are often a source of poorly scalable parallel algorithms, due to their irregular nature and low computational intensity, many graph operations exhibit an embarrassingly parallel nature which can be efficiently parallelized using parallelization strategies borrowed directly from the computer science community.</td>
<td>C+MPI+OpenMP work sharing model interface for OpenMPI: application for the simulation of space-and-band dynamics during the interaction between the solar wind and the Earth magnetic field. Currently, the code is planned in two programming models: multi-threaded MPI version, with added OpenMPI runtime on top of it and OpenCL version with just multi-threaded MPI version. When adding OpenMPI version gives a better performance for the OpenCL task-oriented runtime overhauls the GasPI version shone the promise performance results.</td>
</tr>
</tbody>
</table>

Co-design Apps

- **Directory/Cache**
  - Resource Manager APIs to coordinate access to CPU resources
  - Allows to access data that were allocated in the MPI program without additional copy
  - Allows for increased porting of existing MPI applications

- **Resource Manager**
  - Provides an integer porting of existing MPI applications
  - Copies the parallel environment during its initialization, so keeping existing toolchains, including distribution and installation of the libraries
  - Allow to access data that were allocated in the MPI program without additional copy

Parallel Programming Models

**Directory/Cache**

**Resource Manager**

**Parallel Programming Models**

**MPI**

- **Message passing API**, widely used for distributed memory parallel programming
- **Resource Management**: MPI plus threads
- **Interoperability of MPI plus threads**: Interaction with non-MPI components other than OS/like threads is implementation-dependent
- **Need to strike a balance between thread safety vs. performance optimization**: Solution: MPI endpoints, which communicates with a common interface with multiple tasks for each MPI process and MPI thread

**GasPI**

- **GasPI plus MPI**: Allows for increased porting of existing MPI applications
- **Copies the parallel environment during its initialization**, so keeping existing toolchains, including distribution and installation of the libraries
- **Allows to access data that were allocated in the MPI program without additional copy**

**OpenMP**

- **Interoperability with message passing libraries such as MPI and GasPI**, Improves the performance and programmability of hybrid MPI/GASPI + OmpSs applications by coordinating task scheduling with the message passing engine

**StarPU**

- **Runtime system that enables programmers to exploit CPUs and accelerator units available on a machine**
- **Avoid oversubscription and undersubscription scenarios**

**PaRSEC**

- **Generic framework for architecture-aware scheduling and management of micro-tasks on distributed many-core heterogeneous architectures**
- **Task parallelism and provision of architecture-aware scheduling**

**TAU**

- **C+MPI+OpenMP work sharing model interface for OpenMPI**: application for the simulation of space-and-band dynamics during the interaction between the solar wind and the Earth magnetic field. Currently, the code is planned in two programming models: multi-threaded MPI version, with added OpenMPI runtime on top of it and OpenCL version with just multi-threaded MPI version. When adding OpenMPI version gives a better performance for the OpenCL task-oriented runtime overhauls the GasPI version shone the promise performance results.