General

- Contribute to other work packages
- Workpackages overview

Contribute to other work packages

The control programs will be inserted into the ADMIRE components and applications at control points. Control points are places in the system that implement reconfiguration or scheduling actions [15]. The control points can be used for implementing application-local or system-wide policies.

WP3

3.1 Base mechanisms for malleability

- mechanisms for malleability of compute and io resources
- malleability protocol for expanding/shrinking ad hoc storage system
- runtime mechanisms to allow malleability overall
- API for programmers to indicate that the application can respond to external malleation requests

JGU will design malleability interfaces for the ad hoc storage system in three directions:

1. The ad hoc file system can be expanded or shrunk while it is running. Users can still access the ad hoc file system during this process, albeit with some performance penalties due to the relocation process. The entire ad hoc file system can also be re-located to an entirely different set of nodes and node numbers. This is beneficial when another job's application must access intermediate data but cannot be run on the same set of nodes (due to batch scheduling challenges). Further, this saves intermediate results being stored to the back-end parallel file system and then being reread in another job, causing unnecessary network traffic.
2. The ad hoc file system offers an API to throttle file system operations, that is, metadata operations and read/write operations. This can impact the overall CPU utilization of the file system and can be beneficial with regards to application performance when the application and the ad hoc storage system are co-located on the same nodes. Moreover, throttling I/O operations can reduce network interference of other applications.
3. The ad hoc file system offers a rich configuration interface to enable or disable certain file system features or file system protocols. This can include disabling certain metadata fields, e.g., permission bits, or relaxing file system protocols, e.g., the file creation process, with the goal to increase performance when a feature is not required by the application.

3.2 Scheduling algorithms and policies

- develop algorithms to maximize system throughput and minimize response times of individual jobs
- expand or shrink jobs and improve I/O and computation balance

JGU will contribute their knowledge concerning Quality of Service (QoS) to the decisions of the scheduling algorithms and policies. As the I/O bandwidth to the global parallel file system is limited, and a single application can severely degrade the overall system performance, it is necessary to fairly distribute the bandwidth on the users while taking priorities into account (some users have a higher priority than others and are privileged to more bandwidth). Taking the bandwidth information into account, e.g., via the QoS Lustre extensions developed by JGU and DDN, the scheduler can make informed scheduling decisions based on the current bandwidth usage of the system. Other Slurm plugins, defined and implemented in WP 4, allow users to ask for the required bandwidth when allocating a batch job, which the I/O scheduler can also use for their scheduling policies.

3.3

JGU will implement the designed interfaces into the ad hoc storage system and provide a corresponding API as a control point, allowing the malleability features outlined in T3.1

3.4

JGU will design a new Slurm plugin that extends today's Slurm API so that malleability features in T3.1 can be controlled through Slurm directly. For instance, the Slurm plugin could allow resizing the ad hoc file system or re-locate it to an entirely new set of compute nodes without further user actions. Moreover, JGU will consult with UC3M and TUDA with the goal to provide an API that can integrate the malleability protocols/policies with the malleable runtime and the scheduling policies into the Slurm plugin. With the help of PSNC system and application-centric metrics are evaluated in a real-world environment.

WP4

4.1 definitions of APIs, QoS metrics

- clients can give hints about their I/O requirements to the I/O scheduler when asking for compute resources
- They define which data is accessed, lifetime of data (should it be in gkfs or pfs), if other workflows reuse the data

- We define QoS metrics, such as how many I/O operations per second (of a specific size, say 1 MB), and how resources are prioritized to users
- These can be used to describe the QoS requirements for the job
- We define a syntax to convey the usage of the ad hoc file system
- User API to request data movement between storage tiers
- Add addiitonal user code if the data should be transformed in any way before storing it on the backend FS, e.g. compression.

JGU will lead this task due to their knowledge of the I/O requirements on their ad hoc file systems.

JGU will define the required APIs for the batch scheduler with the project partners so that users can convey their I/O requirements to the I/O scheduler. Example interfaces in the context of ad hoc file systems are

1. paths to the input data and paths where the output data should be placed within the PFS;
2. how long the data should be available on the ad hoc file system, that is, should the ad hoc file system be scheduled within or outside the boundaries of the batch job;
3. if other jobs need access to the data within the ad hoc file system. In this case, the ad hoc file system can run on a dedicated set of nodes, or the following jobs are scheduled to the same nodes of the node-local ad hoc file system. Nevertheless, I/O requirements can also include information about the data placement and distribution beneficial to the users' application.

Further, we define QoS metrics which allow insights on the used bandwidth of a user application. For instance, this can be based on the token-based system of Lustre's QoS extension that JGU and DDN developed in the past in which a user is allowed `x` amount of RPCs per second with each RPC being worth 1 MiB, regardless of wheather a user's I/O request uses the full megabyte of each RPC. This allows us to achieve insights into an application's behavior and allows users to ask for `x` amount of required bandwidth as a newly added the batch scheduler API. Based on the current usage and priorities (some users have a higher priority than others), the resources of the back-end storage system are fairly distributed. Such QoS metrics or, in general, the workload utilization should expand to the ad hoc file system so that users can make informed decisions on the ad hoc file system efficiency when running their applications.

Lastly, an API is defined for the batch scheduler, allowing users to ask for stage-in/out processes between storage tiers, e.g., PFS and ad hoc file system. In addition, the API should include ways to include custom intermediate code while moving the data between tiers. For instance, the job's output data could be processed and compressed before storing it on the PFS.

4.2 Scheduling algorithms and policies

- develop I/O scheduling algorithms and benchmarks to evaluate them.
- the current state of the system is used to drive scheduling decisions.
- Goal is to minimize storage backend congestion, enforce QoS constraits for each job, and reuse data on local storage to minimize data movement and leverage on locality.

- Algorithms to predict I/O behaviors and user/application behaviors with machine learning and other techniques
- Goal is to minimize waiting times

JGU will offer methods to reduce congestion by coordinating ad-hoc file systems and the storage back-end in three ways:

1. We'll define and implement optimized data movement strategies to minimize reading and writing data from and to the PFS, e.g., when staging-in/out data.
2. We enforce the QoS requirements in the batch scheduler defined in 4.1 by leveraging on the Lustre QoS extensions.
3. We implement the interfaces defined in 4.1 so that data can stay within the realms of the ad hoc file system across multiple jobs if they operate on the same input data or rely on the intermediate results of the previous job. In cases the same amount of nodes cannot be used, the data should be transferred between the compute nodes instead of storing them on the PFS.

4.3 On site, in transfer data transformations

- Create control points inside I/O scheduler
- Goal is that users can add custom code that is executed while data is transferred on the network or on-site when the data is on the local storage
- on site data should be resued by rescheduling connected simulation and analysis tasks to the same nodes
- control points will be offered to other admire components and ad hoc storage systems so that applications can use the functionalities through them (????)

JGU will implement interfaces and tools allowing users to execute their custom code for data processing at the compute node (in-situ). These users can then execute user-defined scripts or significantly extend certain file system I/O operations. For example, a modular file system interface could allow custom code to be executed before the ad hoc file system writes the back-end data to disk. One possible use case is the on-the-fly encryption of sensitive data so that raw data is never stored on node-local storage devices, which other users have access to in later scheduled batch jobs.

Workpackages overview

Ad hoc storage system

Ad-hoc storage system. The continuously growing size of HPC systems increases the probability of congestion on the back-end file systems. Ad-hoc storage systems dynamically virtualise on-node storage into a fast storage volume that allows congestion on the back-end storage systems to be reduced and data locality to be improved [9]. ADMIRE will develop two active ad-hoc storage systems with QoS support, addressing I/O performance and scalabil- ity. (Objectives O1, O2, O3, and O6; KPIs 1-6). ADMIRE will provide a high-performance ad-hoc file system that is based on the prototype ad-hoc file system GekkoFS developed by JGU and BSC. GekkoFS can be used by applications as a burst-buffer to alleviate I/O peaks and checkpointing pressure and already provides scalability to more than 500 nodes and delivers more than 40,000,000 file creates per second. It ranked number 4 in the overall 10-node challenge of Nov. 2019 IO500, while using a much smaller storage backend than competing file systems. It also even ranked number 2 concerning metadata performance in the same challenge. GekkoFS will be extended in ADMIRE to support malleability, allowing the dynamic resizing of resources in coordination with the malleability management module, and by integrating reliability mechanisms to enable long-term usage. Exposing control points will allow balancing the computation and I/O performance by coordinating the ad-hoc file system with the job and I/O scheduler. Since not all applications in the HPC ecosystem rely on traditional POSIX I/O interfaces, ADMIRE will also apply these ideas to BSC’s object store dataClay [10], thus providing more generality to the project’s infrastructure. Sections 1.3.2.5 and 1.4.1 provide more details about the ad-hoc storage system, and the advances over the state-of-the-art.

List of work packages

Deliverables:

WP1: Project management (UC3M)

WP2: Ad hoc storage systems (JGU)

- 2.1: Definition of requirements for ad-hoc storage systems (JGU)
- 2.2: Design of the ad hoc storage system (JGU)

- 2.3: Short-lived ad hoc storage system (BSC)
- 2.4: Resiliece for ad hoc storage system (BSC)

WP3: Malleability management (TUDA)

- 3.1: Malleability requirements definition (UC3M)
- 3.2: Base mechanisms for malleability (UC3M)
- 3.3: Scheduling algorithms and policies (TUDA)
- 3.4: Malleable storage systems available (UC3M)
- 3.5: Slurm malleability plugin available (JGU)

WP4: I/O scheduler (BSC)

- 4.1: I/O Scheduler requirements and API (BSC)
- 4.2: Software to support I/O scheduling policies (BSC)
- 4.3: Final report on the I/O scheduler (BSC)

WP5: Sensing and profiling (DDN)

WP6: Intelligent controller (INRIA)

WP7: Application co-design (FZJ)

WP8: Dissemination and exploitation (PARATOOLS SAS)

Description of work (overview)

our envolvements

WP1: Project management (UC3M)

- 1.1: Project management and reporting (UC3M) **All**
  - **All** team leaders of the partners will contribute with individual details.
- 1.2: Scientific-technical coordination **(JGU)** UC3M, TUDA, BSC, DDN, INRIA, FZJ, PARA
  - **JGU** will **lead** the task, provide scientific-technical supervision at WP-level as head of STC, provide coordination between the STC and other project management bodies as

PC.
    - UC3M, **JGU**, TUDA, BSC, DDN, INRIA, FZJ, and PARA will provide scientific-technical coordination at WP-level as leaders of a WP.
- 1.3: Quality control and knowledge management (UC3M) **JGU**, TUDA, BSC, DDN, INRIA, FZ, PARA
    - **JGU**, TUDA, BSC, DDN, INRIA, FZJ, PARA will provide quality-control at WP-level as leaders of WPs.
    - **All** partners will contribute to the elaboration of a yearly internal progress reports and participate in the peer-review process.

WP2: Ad hoc storage systems **(JGU)**

- 2.1: Data layer semantics **(JGU)** BSC, DDN, CINI
    - **JGU defines** and performs the application evaluation, defines semantics, and implements test tools.
- 2.2: Short-lived ad-hoc storage systems (BSC) UC3M, **JGU**, DDN, CINI
    - **JGU** will be responsible for operating system integration and performance tuning.
- 2.3: Resilience mechanisms for ad-hoc storage systems (BSC) **JGU**, UC3M, DDN
    - **JGU** will implement changes required to the data distribution based on the correcting codes.

WP3: Malleability management (TUDA)

- 3.1: Base mechanisms for malleability (UC3M) BSC, CINI, **JGU**, TUDA
    - BSC and **JGU** will design the malleability interfaces of the ad-hoc storage systems developed in WP 2.
- 3.2: Scheduling algorithms and policies (TUDA) **JGU**, UC3M, INRIA
    - **JGU** will contribute the underlying QoS model for I/O.
- 3.3: Malleable storage systems (UC3M) UC3M, **JGU**, BSC
    - **JGU**, UC3M and BSC will add an API to the ad-hoc storage systems developed in WP2.
- 3.4: Integration of malleability features into the job scheduler **(JGU)** PSNC, TUDA, UC3M
    - **JGU** will **lead** the task, create a Slurm plugin skeleton, and integrate the malleability protocol developed in Task T3.1 with the ad-hoc file system.

WP4: I/O scheduler (BSC)

- 4.1: Definition of the I/O requirements, QoS metrics and data movement API **(JGU)** UC3M, BSC, TUDA, DDN, CINI, PSNC
    - **JGU** will **lead** the task due to their knowledge of I/O requirements on their ad-hoc storage systems.
- 4.2: I/O scheduling policies and algorithms (INRIA) UC3M, **JGU**, BSC, TUDA, DDN

    - **JGU** will offer methods to reduce congestion by coordinating ad-hoc file systems and the storage back-end.
- 4.3: In-situ/in-transit data transformations using low-power processors (MPG) **JGU**, BSC, CINI, PSNC
    - **JGU** will provide the tools and mechanisms to run in-situ code in the different ad-hoc storage systems.

WP5: Sensing and profiling (DDN)

- 5.1: Definition of the requirements and design of monitoring and profiling tools (UC3M) **JGU**, BSC, TUDA, DDN, PARA, CINI, PSNC
    - **JGU**, BSC, TUDA and DDN will add the requirements of parallel ad-hoc storage system, malleability, the I/O scheduler, and the back-end file systems requirements.
- 5.2: Scalable monitoring tool for exascale systems (DDN) UC3M, PARA, PSNC, **JGU**
    - PSNC and **JGU** will support the evaluation of LIME-based infrastructures.

WP6: Intelligent controller (INRIA)

- 6.1 Intelligent controller design (UC3M) BSC, **JGU**, INRIA, TUDA, FZJ, PSNC
    - **JGU** will provide the replicated log and control points in the ad-hoc file system.
- 6.2 Novel runtime analytics tools to tune I/O system behaviour (CINI) BSC, **JGU**, UC3M, FZJ, PSNC, INRIA
    - **JGU** will provide its experience on machine learning-based failure prediction in HPC centres.

WP7: Application co-design (FZJ)

JGU not part of

WP8: Dissemination and exploitation (PARATOOLS SAS)

- 8.1 Dissemination and training (CINI) **All**
    - **JGU**, TUDA, BSC, CINI, PARA and FZJ will participate in the organisation and execution of dissemination activities. All partners will contribute.
- 8.2 Exploitation and standardisation (PARA) **All**
    - **JGU** will coordinate the standardisation and interoperabilty activities.
    - **All** partners will participate in exploitation activities.
- 8.3 Communication activities (UC3M) **All**
    - **All** partners will contribute to the organization of the communication activities.

Description of work (detailed)

our envolvements

WP1: Project management (UC3M)

This work package includes the effective management of the project as described in Section 3.2, including monitoring of progress towards milestones and deliverables, evaluation of research results, and the proper dissemination of those results as described in Section 2.2.1. It also provides the overall co-ordination of activities, both financial and technical. Thus, this WP will ensure resource sharing and usage as well as overall smooth execution of the project activities and the organisation project meetings. The project coordinator will prepare reviewing meetings, ensure the flow of information to the partner teams, signal any delay in providing the requested contributions, and identify deviations from the workplans.

- 1.1: Project management and reporting (UC3M) **All** This task includes activities related to project management and reporting including: prepare a project management handbook, supervise the elaboration of a data management plan, supervise financial, legal, administrative and technical matters involved in the project, act as the main contact point for the European Commission, approve and submit the project deliverables, organise and chair the General Assembly (GA) meetings, develop a communication and reporting strategy, and maintain continuous communication with all project management bodies. Role of partners:
  - UC3M, as project coordinator, will be in charge of all activities of the task.
  - **All** team leaders of the partners will contribute with individual details.

  Task output:
  - Project handbook as deliverable D1.1.
  - A data management plan and its update.
  - Midterm and final reports.
  - Internal midterm review reports.
- 1.2: Scientific-technical coordination **(JGU)** UC3M, TUDA, BSC, DDN, INRIA, FZJ, PARA This task includes activities related to the technical coordination of the project: coordinate the scientific-technical activities across WPs, ensure regular communication across project members in monthly video-conferences, organize and chair the Scientific Technical Committee (STC) meetings, compile and distribute meeting minutes, monitor and evaluate the project performance, analyze risks and conduct contingency planning, monitor and justify the resource allocation. Role of partners:
  - **JGU** will lead the task, provide scientific-technical supervision at WP-level as head of STC, provide coordination between the STC and other project management bodies as PC.
  - UC3M, **JGU**, TUDA, BSC, DDN, INRIA, FZJ, and PARA will provide scientific-technical coordination at WP-level as leaders of a WP. Task output:
  - Participation in the elaboration of midterm and final reports and yearly internal progress reports.

- 1.3: Quality control and knowledge management (UC3M) **JGU**, TUDA, BSC, DDN, INRIA, FZ, PARA This task includes activities related to the quality control: elaboration of a quality management plan, elaboration of yearly internal progress reports used for quality supervision, quality supervision at work package and project levels, peer review of documents, software, and papers. Documentation and deliverables will be hierarchically organized via a knowledge base whose elements will be readable by all project partners. Write access, administration, and use of quality- control tools wil be available for those responsible. All WP leaders are expected to work in the activities of this task.
  Role of partners:
  - UC3M will lead the task, will supervise the quality control at project level, coordinate quality plan and reporting, and prepare and administer the quality-control tools.
  - **JGU**, TUDA, BSC, DDN, INRIA, FZJ, PARA will provide quality-control at WP-level as leaders of WPs.
  - All partners will contribute to the elaboration of a yearly internal progress reports and participate in the peer-review process.

  Task output:
  - Project handbook and Quality Management Plan.
  - Midterm and final reports.

WP2: Ad hoc storage systems **(JGU)**

This WP develops ad-hoc storage systems to efficiently use node-internal NVMe and persistent memory technologies to reduce the pressure on back-end storage systems. It simplifies ad-hoc storage development as much as possible by defining minimal storage semantics (Task 2.1) and also provides storage systems without resilience (Task 2.2). Nevertheless, also longer-running applications and workflows will be supported by including additional error correcting codes (Task 2.3). The storage systems developed in this work package will leverage GekkoFS and the dataClay object storage.

- 2.1: Data layer semantics **(JGU)** BSC, DDN, CINI The POSIX semantics hinders parallel file systems to scale to exascale. This task will define new semantics that overcome these scalability limitations and will also simplify parallel file system implementation. The input for the new semantics will be an evaluation of the POSIX usage of typical parallel, big-data, and AI applications. The result will be a set of semantics which will be the foundation for the development of ad-hoc file systems and object stores in the following tasks which can be optimised for different scenarios. The task will also develop test tools to characterise the file system semantics and to test the semantic assumptions of HPC applications.
  Role of partners:
  - **JGU** defines and performs the application evaluation, defines semantics, and implements test tools.
  - BSC defines application evaluation parameters and assembles required tracing and evaluation tools. It performs the evaluation on MareNostrum and co-develops new semantics.

    - DDN provides its field-experience with all major file systems to discuss new file system semantics.
    - CINI contributes to the co-design of the data layer design with the Intelligent Controller (WP 6). Task output:
    - Report on new file system semantics including compatibility test tools.
    - Evaluation report on the usage of file system semantics of typical HPC applications
- 2.2: Short-lived ad-hoc storage systems (BSC) UC3M, **JGU**, DDN, CINI The task will implement short-lived ad-hoc file systems and object stores, which do not have to survive node failures. The file systems will leverage the use of node-internal NVMe and persistent memory technologies to simplify the development and increase system performance. This short-lived storage systems can be used, e.g., when data is mostly read-only and can be easily recovered from the back-end file system.
  The ad-hoc file systems will support the semantics from T2.1 and will take GekkoFS as input. The implementations are based on building blocks, which can be developed independently from the file system and keep minimum global state. The data distributions will be based on randomised distributions coupled with locality hints. The ad-hoc file systems will cooperate with the back-end by providing a global namespace, e.g., by using Lustre stubs. Furthermore, data in the ad-hoc file system can be switched to read-only in the back-end. Control points provide I/O bandwidth management following the advise of the intelligent controller from WP 6. The object store will be based on the dataClay's active object store that enables the execution of user code in the storage system. This feature will be leveraged to optimise application performance, e.g., by asynchronously writing back the object to the back-end storage system as soon as it is not used anymore by the accessing application, so that other applications can already access it without the need to wait for the original application. Data objects in the ad-hoc object store can also be organised in the appropriate data structures to optimise the performance of the running application, regardless of the original organisation of the data in the back-end.
  Role of partners:
    - BSC will lead the task and participate in the ad-hoc storage systems development with a focus on metadata.
    - UCM3 will focus on the building blocks and data distribution.
    - **JGU** will be responsible for operating system integration and performance tuning.
    - DDN will link the Lustre HSM mechanisms with the ad-hoc file system implementations.
    - CINI will contribute to the co-design of the ad-hoc file system with the intelligent controller (WP 6).

  Task output:
    - Set of ad-hoc storage system implementations being optimized for the semantics developed in T2.1.
- 2.3: Resilience mechanisms for ad-hoc storage systems (BSC) **JGU**, UC3M, DDN The file systems and object stores developed in T2.2 and T3.3 already contain all necessary functionalities to deploy them as ad-hoc storage systems. This task includes error correction schemes to enable ad-hoc storage systems to run for longer time periods (months to years). It focuses on the development of two different reliability encodings: a simple mirroring scheme and additionally two-error correcting erasure codes based on

fast SSE/AVX encodings. The recovery implementation will simply replace broken nodes based on standard recovery schemes, but will also implement distributed parity calculations based on advanced data placement schemes introduced in Task T3.3.
Role of partners:
- BSC will lead the task and the integration of SSE/AVX encodings.
- UC3M will be responsible for development and integration of the replication approach.
- **JGU** will implement changes required to the data distribution based on the correcting codes.
- DDN will participate in the optimization of SSE/AVX codes.

Task output:
- Long-living ad-hoc storage systems supporting resilience mechanisms.

WP3: Malleability management (TUDA)

We will provide base mechanisms for the combined malleability of compute and I/O resources. Those mechanisms will be guided by new scheduling algorithms and policies, integrated into Slurm via a plugin, that are able to maximise throughput of the system by balancing computation and I/O. Moreover, we will add malleability to the ad-hoc storage systems developed in WP 2.

- 3.1: Base mechanisms for malleability (UC3M) BSC, CINI, **JGU**, TUDA This task will provide base mechanisms for the combined malleability of compute and I/O resources. First, we will design and implement a malleability protocol to negotiate the precise terms of an expand/shrink operation between the job and the ad-hoc storage system on the one hand, and the job scheduler on the other. The protocol will also support the job-initiated evolution of the resource set. Second, we will develop runtime mechanisms, by generalising FlexMPI, to provide malleability to applications and to execute them using the previous negotiation protocol. Finally, we will develop an API that allows programmers to insert scheduling points into their codes that indicate when the application can respond to external malleation requests.
  Role of partners:
  - UC3M will lead the task and implement the malleable runtime, including the scheduling-point API.
  - BSC and **JGU** will design the malleability interfaces of the ad-hoc storage systems developed in WP 2.
  - CINI will contribute to the co-design of the malleability manager with the intelligent controller in WP 6.
  - TUDA will design and implement the malleability protocol and contribute to the scheduling-point API.

  Task output:
  - Base components and extensions for the combined malleability of compute and IO resources.
- 3.2: Scheduling algorithms and policies (TUDA) **JGU**, UC3M, INRIA This task will develop scheduling algorithms and policies that are able to maximise throughput of the system and the response times of individual jobs by expanding or shrinking malleable jobs while

maintaining or even improving the balance between compute and I/O resources within their allocation. Moreover, we will define and implement several policies that will allow system administrators and users to specify and tradeoff optimisation objectives at the level of the system or individual jobs, respectively.
Role of partners:
  - TUDA will lead the task as well as design and implement the scheduling algorithms.
  - INRIA will provide its expertise in scheduling and work on the interaction with task T6.3.
  - **JGU** will contribute the underlying QoS model for I/O.
  - UC3M will design and implement I/O-aware malleability policies.

Task output:
  - Scheduling algorithms for compute and I/O under the assumption that a subset of the jobs is malleable. • Configurable policies for the above algorithms.

- 3.3: Malleable storage systems (UC3M) UC3M, **JGU**, BSC This task will add a malleability interface to the ad-hoc storage systems developed in WP 2 to invoked by the malleability protocol. Data resiliency will be supported through adapting the results from TaskT2.3 according to insights gained from .
Role of partners:
  - UC3M will lead the task and focus on the two-phase locality preserving data distribution.
  - **JGU**, UC3M and BSC will add an API to the ad-hoc storage systems developed in WP2.

Task output:
  - Malleable ad-hoc storage systems supporting configurable data layouts as well as locality preserving distributions.

- 3.4: Integration of malleability features into the job scheduler **(JGU)** PSNC, TUDA, UC3M In this task, we will integrate the scheduling algorithms and the malleability protocol created in Tasks T3.1 as a plugin into Slurm. In addition, the protocol will be integrated into all components Slurm needs to negotiate with, so that resulting changes can be forwarded, for example, to the I/O scheduler being developed in WP 4.
Role of partners:
  - **JGU** will lead the task, create a Slurm plugin skeleton, and integrate the malleability protocol developed in Task T3.1 with the ad-hoc file system.
  - UC3M will integrate the malleability protocol developed in T3.1 with the malleable runtime and integrate the policies from T3.2 into the plugin.
  - TUDA will integrate the malleability protocol and scheduling algorithms from T3.1 into the plugin.
  - PSNC will evaluate system- and application-centric metrics in a real environment.

Task output:
  - A Slurm plugin supporting malleability for compute and I/O resources plus the malleability protocol integrated with job runtimes and ad-hoc storage systems.

WP4: I/O scheduler (BSC)

Due to the continuous data traffic of ad-hoc storage systems and the complexity of the HPC storage hierarchy, I/O operations involving the shared back-end file system need to be coordinated to limit congestion while minimising batch job waiting times. This WP will develop an I/O Scheduler with control point support that coordinates inputs from the intelligent controller and the resource and malleability managers to provide QoS-aware data scheduling. Functionalities to support in-situ/in-transit data transformations will be provided, and using low-power processors for such tasks will be researched.

- 4.1: Definition of the I/O requirements, QoS metrics and data movement API **(JGU)** UC3M, BSC, TUDA, DDN, CINI, PSNC This task will define an API for applications to convey their I/O requirements to the I/O Scheduler when asking for resource allocations from the batch scheduler. Clients will be able to define, e.g., which data will be accessed by a batch job, the type of access (e.g., input, output, input/output), the expected lifetime (e.g. temporary/persistent) and the expected reuse profile (e.g., if jobs in a workflow will reuse it). QoS metrics and priorities will be defined to describe the QoS requirements of a job, as well as the syntax and semantics of channels to convey dynamic information about the usage of ad-hoc storage systems. The API will include command definitions for users to request data movement between tiers and support the deployment of user code to enable filtering and transformation of data.
  Role of partners:
  - **JGU** will lead the task due to their knowledge of I/O requirements on their ad-hoc storage systems.
  - DDN will contribute expertise on QoS metrics and algorithms, based on previous work on QoS for Lustre.
  - BSC will provide expertise on I/O requirements and data movement from projects IOSTACK/NEXTGenIO.
  - TUDA, UC3M, CINI and PSNC will contribute to the API definition and will design connections between the API and middleware components.

  Task output:
  - The task will generate deliverable D4.1 and provide API of the system.
- 4.2: I/O scheduling policies and algorithms (INRIA) UC3M, **JGU**, BSC, TUDA, DDN Task T4.2 will develop I/O scheduling algorithms and evaluate them using benchmarks. Information about the current state of the system will be used as input for I/O scheduling decisions, with the goals of minimising congestion to the storage back-end, enforcing QoS constraints defined for each job, and reusing data already in compute nodes to leverage locality. The algorithms will explore predictable and unpredictable I/O behaviours and will include classical congestion minimisation as well as machine learning approaches to predict user/application behaviour . Job waiting times should also be minimised as much as possible to improve HPC throughput. This task will collaborate with T5.2, T5.3, and T6.3 to integrate up-to-date system information and the outcome of decentralised decision making into I/O scheduling decisions. The I/O Scheduler will be based on the NORNS staging service developed by BSC and the QoS steering of the back-end file system will be based on QoS extensions for Lustre developed by DDN and JGU.
  Role of partners:

    - INRIA will lead the task and design the algorithmic system based on its expertise on I/O scheduling.
    - **JGU** will offer methods to reduce congestion by coordinating ad-hoc file systems and the storage back-end.
    - BSC will provide its expertise in coordinating in-transit components and computational tasks and I/O tasks.
    - DDN will leverage its QoS expertise for developing I/O scheduling algorithms.
    - TUDA will provide the malleability test programs used in WP3 as test cases for the scheduling algorithms.
    - UC3M will provide experience in ad-hoc storage scheduling algorithms and mechanisms to tune them.

  Task output:
    - The task will provide a set of policies for the I/O Scheduler that will be presented on Deliverable D4.3.
- 4.3: In-situ/in-transit data transformations using low-power processors (MPG) **JGU**, BSC, CINI, PSNC This task will create control points and functionalities inside the I/O Scheduler to allow users to deploy custom data code to be executed in-transit while data is transferred to compute nodes and/or in-situ if data already resides in a compute node. In-situ processing further allows data-intensive analysis tasks to co-exist (and be co-scheduled) with simulation tasks, reducing the amount of data that has to be moved between simulation and analysis tasks. The control points developed will also be offered to other ADMIRE components and especially to ad-hoc storage systems, so that applications can leverage these functionalities through them. This task will focus on heterogeneous node- and interconnect-architectures involving low power devices co-located with traditional CPUs/GPUs to reduce energy consumption for both data processing and data movement. The usage of accelerators from the European Processor Initiative, if available, will be investigated.

  Role of partners:
    - MPG will lead the task and specifically focus on enabling in-situ/in-transit data analysis and transformation.
    - BSC will offer NORNS as implementation skeleton as well as expertise on filtering from previous projects.
    - **JGU** will provide the tools and mechanisms to run in-situ code in the different ad-hoc storage systems.
    - CINI and PSNC will provide its experience in Data Stream and machine learning runtimes.

  Task output:
    - A collection of in-situ/in-transit data transformation filters and analysis tasks presented on Deliverable D4.3.

WP5: Sensing and profiling (DDN)

This WP will investigate and develop scalable monitoring (T5.2) and profiling tools (T5.3), including low-level instrumentation, data collection, mining and data-centric online performance analysis, able to scale to the exascale level. The WP will put emphasis not only on monitoring performance

metrics, but also on modelling applications I/O profiles that enable to predict the scaling behaviour of applications (T5.3). Starting from historic I/O profiles and user hints, ADMIRE will generate dynamically feedback for the controller and help it to understand the interplay between applications necessary for online optimisation.

- 5.1: Definition of the requirements and design of monitoring and profiling tools (UC3M) **JGU**, BSC, TUDA, DDN, PARA, CINI, PSNC The goal of T5.1 is to define a set of probs and their respective API in order to feed a family of performance models with the relevant information. As the model will be iteratively defined during the life time of the project, T5.1 will start with the most generic probes and then refine and develop more advanced features. One example being an initial measurement of the CPU load and the IO load and later wise a measurement of the overlap between CPU and I/O activities. The probe mechanism envisioned in T5.1 is hierarchical, in the sense that a rules-based language is proposed to trigger additional measurements.
  Role of partners
  - UC3M will lead the task and will evaluate various transport alternatives.
  - **JGU**, BSC, TUDA and DDN will add the requirements of parallel ad-hoc storage system, malleability, the I/O scheduler, and the back-end file systems requirements.
  - PARA and DDN will provide its expertise and tools to help the characterisation process.
  - CINI and PSNC will provide a characterisation of workflow and machine-learning applications and constraints of real HPC systems.

  Task output
  - Definition of API and transport requirements for the profiling and I/O monitoring tool.
- 5.2: Scalable monitoring tool for exascale systems (DDN) UC3M, PARA, PSNC, **JGU** This task will enable the development of active components through a dynamic monitoring tool that will implement the API defined in Task T5.1. We will implement gather and reduction mechanisms that will be used for both periodic aggregations and on-demand performance queries. Based on defined threshold filters, only deltas will be transmitted to limit the volume of data to be sent to the analytics. The notification of events will be done on top of the control plane by using the publish/subscribe functionality. Our solution will integrate support for file system monitoring based on the Lustre Intelligent Management Engine (LIME) developed by ADMIRE partner DDN and system monitoring using Paratools TAU as a base.
  Roles of partners
  - DDN will lead the task and extend its LIME tool. PARA will extend Paratools TAU system.
  - UC3M will implement the reduction, gather, and notification mechanisms.
  - PSNC and **JGU** will support the evaluation of LIME-based infrastructures.

  Task output
  - A system monitoring tool for dynamically aggregating/reducing performance metrics.
  - An event publication system detecting bottlenecks or other conditions defined by the system admin or applications hints.
- 5.3: Tools for application I/O profiling and modelling (PARA) UC3M,INRIA,PSNC,TUDA

- 5.4: Predictive performance (DDN) UC3M, INRIA, TUDA

WP6: Intelligent controller (INRIA)

This work package integrates and analyses cross-layer system data to dynamically and intelligently steer the system components. It will optimise at system-scale the data management and the I/O accesses of the running applications based on the input provided by the ecosystem (WP5) and will enforce policies (e.g., I/O scheduling (WP4)) through malleability (WP3) and I/O management (WP2). In order to take decision it will rely on machine learning techniques to predict resource usage and application behaviour.

- 6.1 Intelligent controller design (UC3M) BSC, **JGU**, INRIA, TUDA, FZJ, PSNC This task will enable component orchestration by developing a control plane as described in Section 1.3.2.4. The control service will be implemented starting from Clarisse. For achieving scalability, we will extend it to become a distributed control infrastructure that still provides a single system image: each application will have its own controller agent and a global coordination will be promoted through a distributed coordination protocol.
  The control plane will also manage a replicated log for resilience. An API will be designed to provide access to the control plane following the requirements of the ADMIRE components. The control plane will then steer the main active components including the job scheduler, back-end storage systems, and I/O middleware.
  Role of partners:
  - UC3M will lead the task and will participate in the development of the control plane, with the support of PSNC and FZJ.
  - INRIA will work on the API and the scalability in interaction with Task T6.3.
  - BSC will interact with Task T5.1 and provide the control points of the object storage systems.
  - **JGU** will provide the replicated log and control points in the ad-hoc file system.
  - TUDA will design the interface between the intelligent controller and the malleability manager. Task output:
  - A control plane for orchestrating system components.
  - An API for the control plane.
- 6.2 Novel runtime analytics tools to tune I/O system behaviour (CINI) BSC, **JGU**, UC3M, FZJ, PSNC, INRIA The mechanisms to improve I/O system behaviour and facilitate anticipatory decisions for resource allocations will be investigated in this task. This will be achieved based on the knowledge collected from the IO systems, application runs and the batch system. Additional information will be collected by the profiling and monitoring tools developed in WP 5 and the historical job records of previous runs. This will enable to handle critical stages such as the co-scheduling of many I/O intensive jobs that could cause I/O contention and under-utilisation of other resources, ultimately degrading performance. The strategy is to train model classifiers that can handle jobs that have heterogeneous I/O patterns, detect congestion, perform resource provisioning, perform anomalies detection that will help to identify and predict potential problems and failures of applications. It will provide the knowledge and a methodology at the basis of the decision-making process in Task T6.3.

Role of partners:
    - CINI will lead the task, define monitoring metrics and actions, and evaluate machine learning approaches.
    - INRIA and UC3M will work on machine learning for I/O congestion management.
    - BSC will add the required output metrics for the different ad-hoc storage systems and the I/O Scheduler.
    - **JGU** will provide its experience on machine learning-based failure prediction in HPC centres.
    - PSNC will contribute to the anomaly detection and will provide training data sets from its HPC centre.
    - FZJ will develop novel machine (deep) learning models to support decentralised scheduling decisionmaking and predict potential failure of jobs with large memory footprint. Task output:
    - The task will generate a tool to be integrated into the intelligent controller embedding one or more machine learning techniques able to steer system behaviour to tune I/O performance dynamically.
- 6.3 Novel techniques for storage and applications multi-criteria and distributed control (INRIA) UC3M, PSNC
- 6.4 Coordination language for high-level intelligent I/O (CINI) UC3M

WP7: Application co-design (FZJ)

JGU not part of

WP8: Dissemination and exploitation (PARATOOLS SAS)

This workpackage is responsible for the dissemination of research results, the creation of an exploitation plan, promoting open source releases of ADMIRE framework and improving public awareness of the project, as described in Section 2.2. We will set up a project website to include all public documents and publications related to the project. Technical workshops and showcases will be held every year, probably co-located with major conferences, to contact with research groups, industrial researchers, and associations (as HIPEAC).

- 8.1 Dissemination and training (CINI) **All** Dissemination activities aim at publicly disclosing the project development and results to interested stakeholders: European HPC initiatives, business partners, HPC infrastructure operators, research peers, user groups, standardisation bodies, and policy makers. The project results will be presented in top international conferences and journals, ADMIRE workshops, network meetings (e.g., HPC SummitWeek, HIPEAC ComputingWeek, PRACE Days, etc.), user group and application developers meetings (e.g., CoE for HPC), training events (e.g at TeraTec Forum), meetings with policy makers, discussion fora, and mailing lists. The ADMIRE consortium will prepare promotional material to raise awareness, present the results, and encouraging

exploitation (press, newsletters, videos, etc.) A web site will be created in the first month to publish all ADMIRE related information (See Section 2.2.1).
Role of partners:
  - CINI will lead the task, will supervise the participation in network and user group meetings, and will coordinate the organisation of training events.
  - UC3M will set up the web site, and coordinate the organisation of the ADMIRE workshops.
  - DDN will monitor the accomplishment of communication KPIs and support participation in tech. fairs.
  - **JGU**, TUDA, BSC, CINI, PARA and FZJ will participate in the organisation and execution of dissemination activities. All partners will contribute.

Task output:
  - Project web page.
  - Dissemination plan.
  - Scientific publications and workshops of the project.
  - Training events and participation in conferences and technology fairs.
- 8.2 Exploitation and standardisation (PARA) **All** The exploitation activities will focus on developing high-quality products (primarily open-source software), demonstrating the software, and protecting the results upon partner agreement. The consortium will provide active support for technology adoption by setting up users groups, collaborative platforms, and demonstrations in technology fairs (See Section 2.2.2). Project partners will get actively involved in MPI, POSIX, and languages working groups, to which some partners already belongs. The complementary extreme-scale technologies (e.g., Lustre, Slurm) will be monitored for insuring the interoperability with project results.
Role of partners:
  - PARA will lead the task, coordinate the organisation of software demonstration, advise the project participants on support procedures, and participate in the fairs and exploitation activities.
  - JGU will coordinate the standardisation and interoperabilty activities.
  - UC3M will set up collaborative platforms for documentation and support.
  - **All** partners will participate in exploitation activities.

Task output:
  - Exploitation and standardisation plan.
  - Open-source software in public repositories and collaborative platforms.
  - Creation of user groups.
  - Proposals/ideas for standardisation.
- 8.3 Communication activities (UC3M) **All** The communication activities target for reaching out the society as whole and specific audiences. Project partners will regularly produce news (regular press, magazines, web portals, videos, etc.) for raising awareness about the project among the society as a whole, industry, and policy-makers at all levels. For reaching a wide audience of journalists the communication units will leverage AlfaGalileo and EurekAlert two leading global main distribution channels of research news. Project representatives will give interviews to traditional mass-media channels or on-line portals for informing the public-at-large about the project impact and the positive role of the support of European Union. The partners will constantly disseminate the news and videos

about the project in the most popular social networks.
Role of partners:
- UC3M will lead the task, supervise production of news and videos, and will organize communication activities.
- DDN will monitor the accomplishment of communication KPIs.
- **All** partners will contribute to the organization of the communication activities.

Task output:
- Communication plan.
- Communication materials (videos, newletters, press releases, etc.).
- A networking strategy to promote solutions provided for commercial exploitation.

Technical objectives

- Objective 1 (O1): Enable the efficient use of new storage tiers by subjecting storage to HPC scheduling de- cisions and establishing a distributed control that, based on global monitoring, can dynamically adapt storage allocations to changing application demands.
- Objective2(O2):IncreaseapplicationthroughputofHPCsystemsbyleveragingtheperformance advantageof fast, node-local storage tiers through novel, European ad-hoc storage systems, and in-transit/in-situ processing facilities.
- Objective 3 (O3): Balance computation and data transfers by providing elastic mechanisms to dynamically adjust the ratio between the allocations of compute and storage resources.
- Objective 4 (O4): Reduce I/O interference via globally coordinated minimisation of data transfers between storage tiers, while conveying and enforcing end-to-end QoS needs.
- Objective 5 (O5): Provide tools to co-design applications and storage systems with the goal of minimising data movement, targeting different HPC architectures.
- Objective 6 (O6): Increase power-efficiency in data management operations by reducing data movement and adopting low-power storage and CPU technologies.