RADICAL-Cybertools: Middleware Building Blocks NSF's Cyberinfrastructure Ecosystem (NSF 1931512)

Motivation: Middleware Building Block

Motivation: Sophisticated and scalable workflows have become essential for advances in computational science. In spite of the many successes of workflow systems, there is an absence of a reasoning framework for end-users to determine **which** systems to use, **when** and **why**. Workflows are increasingly a manifestation of the algorithmic and methodological advances; workflow users and workflow system developers are often the same. Workflow systems must be easily extensible so as to support diverse functionality and the proverbial "last mile customization".

We advance the science of workflows and prevent workflow system "vendor lockin" by formulating a **building blocks** approach to middleware for workflow systems grounded on four design principles of **self-sufficiency**, **interoperability**, **composability**, and **extensibility**. A building block has: (i) one or more modules implementing functionalities to operate on a set of explicitly defined entities; and (ii) two well-defined and stable interfaces, one for input and one for output.

Properties of building blocks

- **Self-sufficiency:** design does not depend on the specificity of other building blocks
- **Interoperability:** can be used in diverse system architectures without semantic modifications
- **Composability:** its interfaces enable communication and coordination with other building blocks
- **Extensibility:** its functionalities and entities can be extended to support new requirements or capabilities

Primary Functional Levels

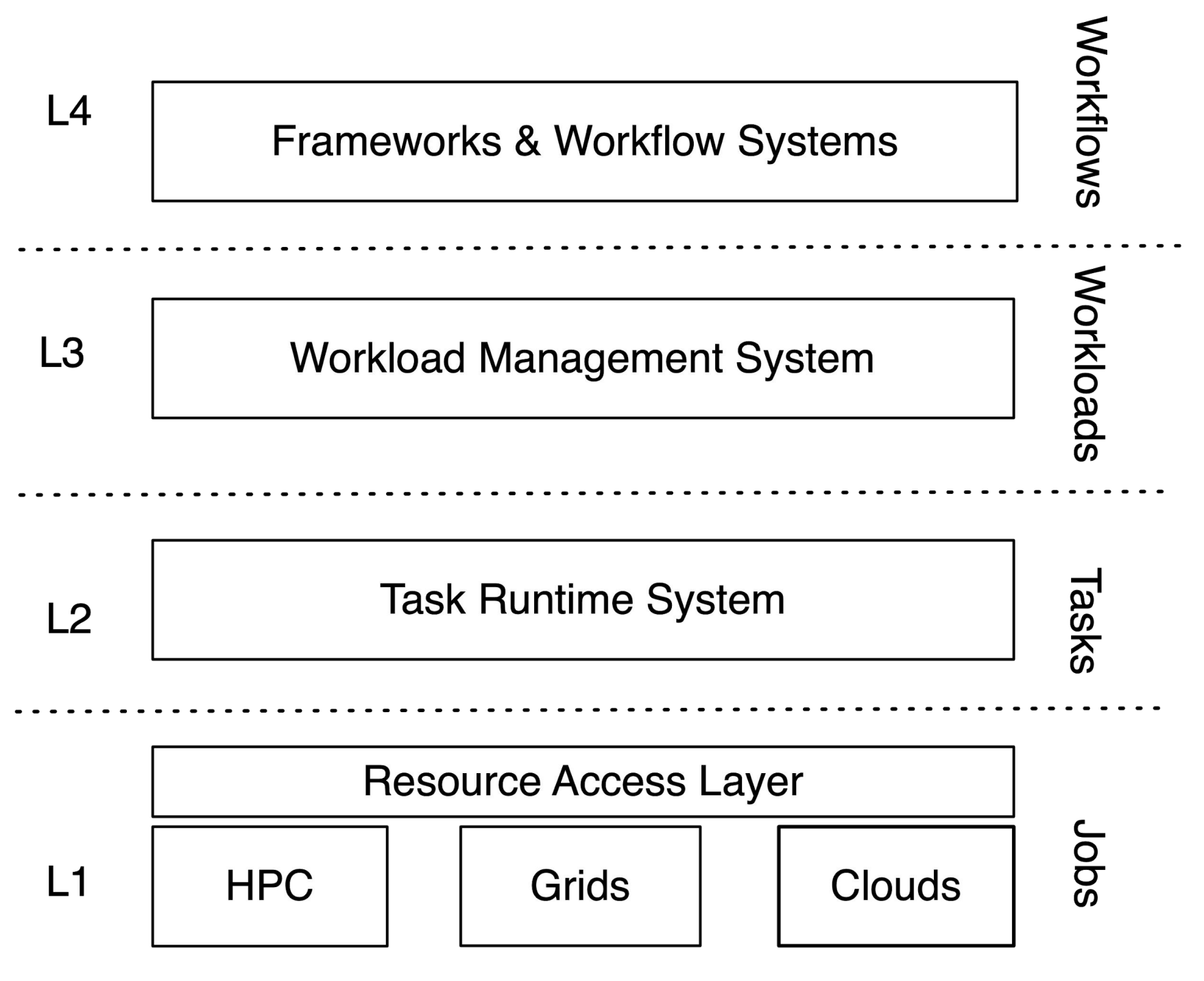
The diagram below describes the primary functional requirements for workflow systems. **RADICAL-Cybertools** are designed and implemented in accordance with the building block approach, span functional levels:

(L4) Workflow and Application Description: Requirements and semantics of applications and workflows.

(L3) Workload Management System: Applications devoid of semantic context are expressed as workloads.

(L2) Task Runtime System (TRS): Execution of the tasks of a workload.

(L1) Resource Access Layer: Capabilities, availability and interfaces required by the tasks to be executed.

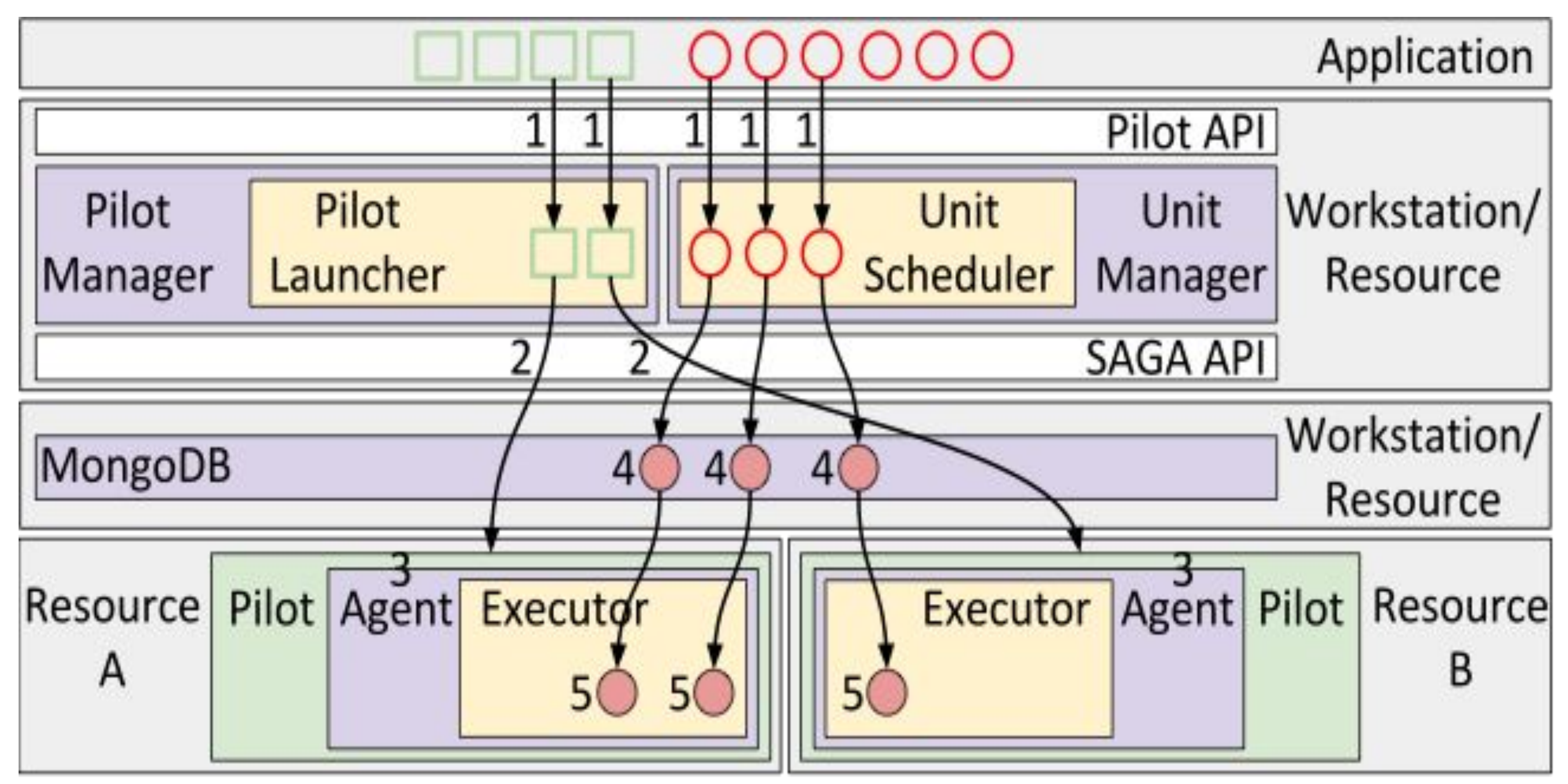


(L2-L1) Interface to Resource

RADICAL-SAGA ***(Simple API for Grid Applications):*** Provides an interoperability layer that lowers the complexity of using distributed infrastructure whilst enhancing sustainability of distribut- ed applications, services, and tools in the form of a Python API. By abstracting away the heterogeneity of the underlying systems, RADICAL-SAGA **simplifies access** to many distributed cyberinfrastruc-tures such as XSEDE and OSG.

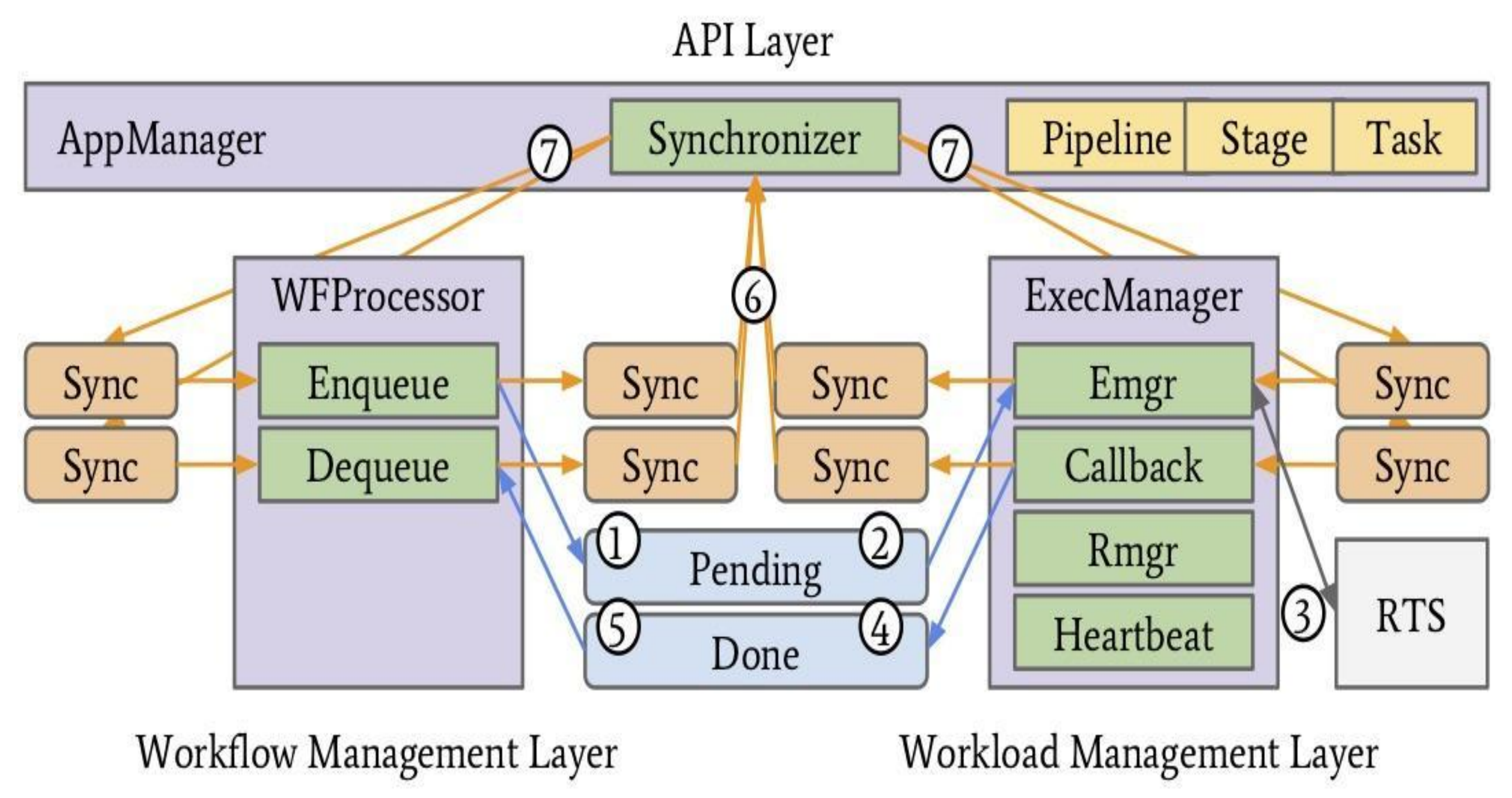
(L2) Task Runtime Management

RADICAL-Pilot: Scalable pilot system for the simple and versatile execution of concurrent and distributed many-task applications on clusters, grids, and clouds. RADICAL-Pilot offers users a lightweight Python API to handle a variety of workloads—including MPI, multiprocess, multithreaded, CPU, and GPU tasks—and scheduling O(10k) tasks while marshalling O(10k) distributed cores.



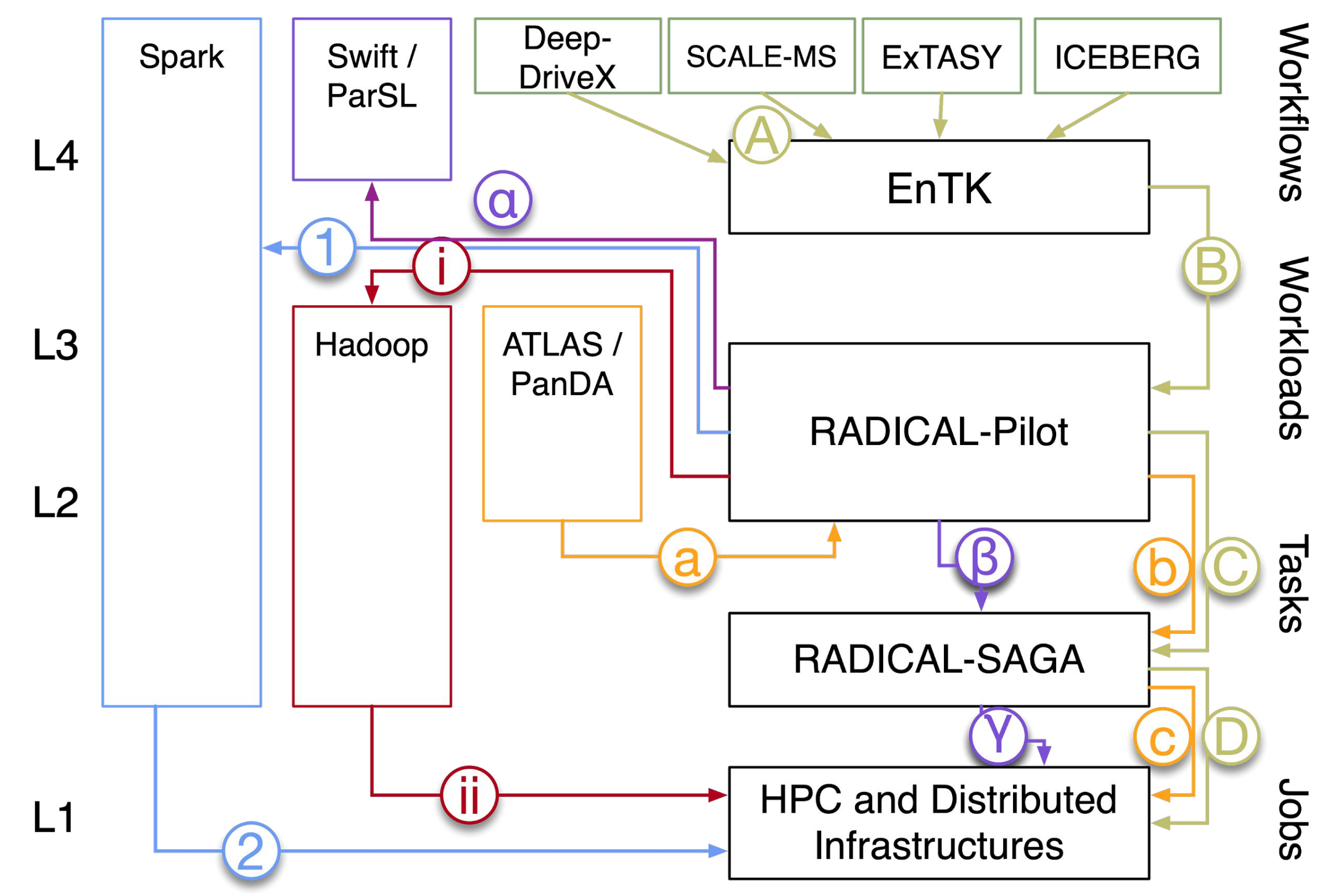
(L3-L4) Workflow / Workload Management

Ensemble Toolkit (EnTK): Provides the ability to execute flexible combinations of **ensemble-based workflows.** It promotes "ensembles" to a first class entity, by taking charge of where and how the ensemble workload is executed. EnTK exposes the pipeline-stage-task (PST) programming model & interface to express ensemble based workflows.



(L4) Workflow Management Systems

ExTASY: Enables advanced sampling of complex macromolecules using molecular dynamics.
DeepDriveX: Enables scalable and concurrent execution of Machine Learning (ML) and HPC workloads (X) on HPC Platforms.
ICEBERG: Enables scalable image analysis on HPC. It allows the integration of image analysis frameworks and algorithms.
SCALE-MS: Enables the concurrent execution of adaptive ensemble algorithms. It uses RADICAL-Pilot for workload execution.



RADICAL-Cybertools support multiple points of integration, "unifying" conceptual reasoning across otherwise different tools and systems

Integration with other Tools

SeisFlows

- Supports seismic inversion workflows on HPC machines, at scale
- We integrated **SeisFlow**
  - with **RADICAL-SAGA** (L1) to execute compute jobs
  - with **RADICAL-EnTK** (L3) to orchestrate tasks and data staging

ATLAS (Panda and Harvester)

- **PanDA** is a WMS designed to support the distributed execution of workflows via pilots.
- We integrated **Panda** and **RADICAL-Pilot** to improve its scaling on large HPC resources, and integrated **Harvester** and **RADICAL-Pilot** to provide scalable task execution on HPC machines

Swift /ParSL

- **Swift** is a language and a runtime system to execute workflows.
- We integrated **Swift** with **RADICAL-WLMS** (L3) to execute workloads concurrently on HPC and HTC resources.

Fireworks

- Fireworks is a system that enables material science workflows
- We integrate **Fireworks** and **RADICAL-Pilot** (L2) to improve its scaling on HPC resources

For more information: **http://radical.rutgers.edu**