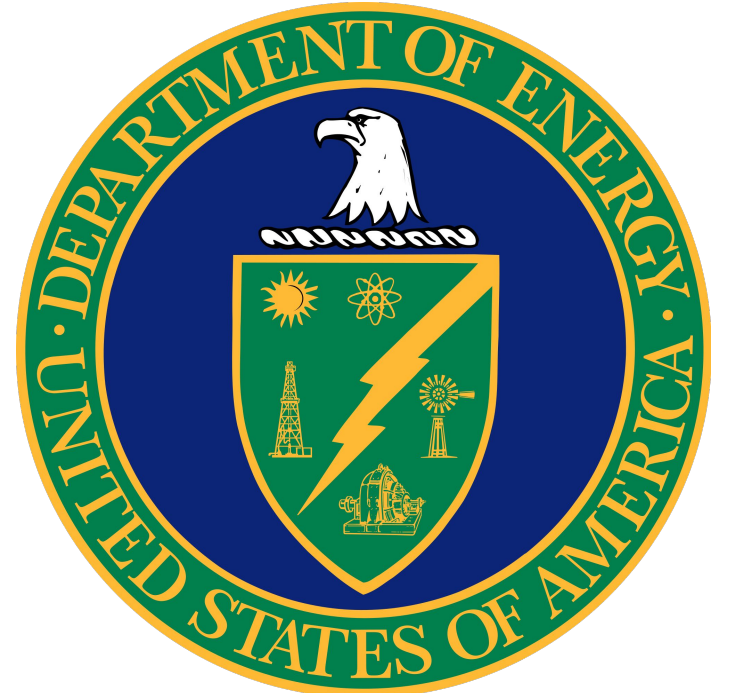


# RADICAL-Cybertools: Building Blocks for Workflow System Middleware



## Overview

**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

## RADICAL-Cybertools: An implementation of the Building Block Approach to Middleware

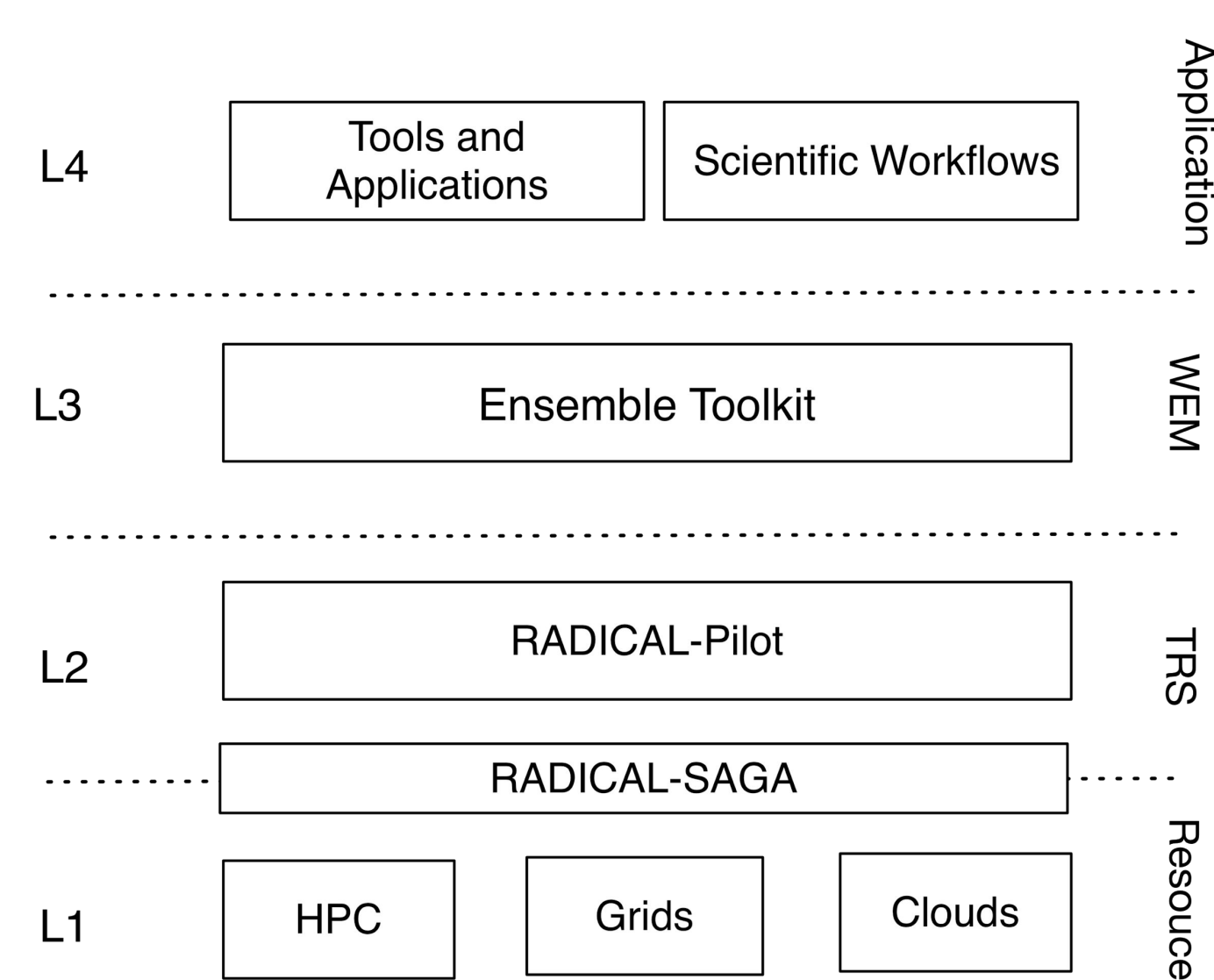
RADICAL-Cybertools are designed and implemented in accordance with the building block approach, spanning four functional levels:

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

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

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

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



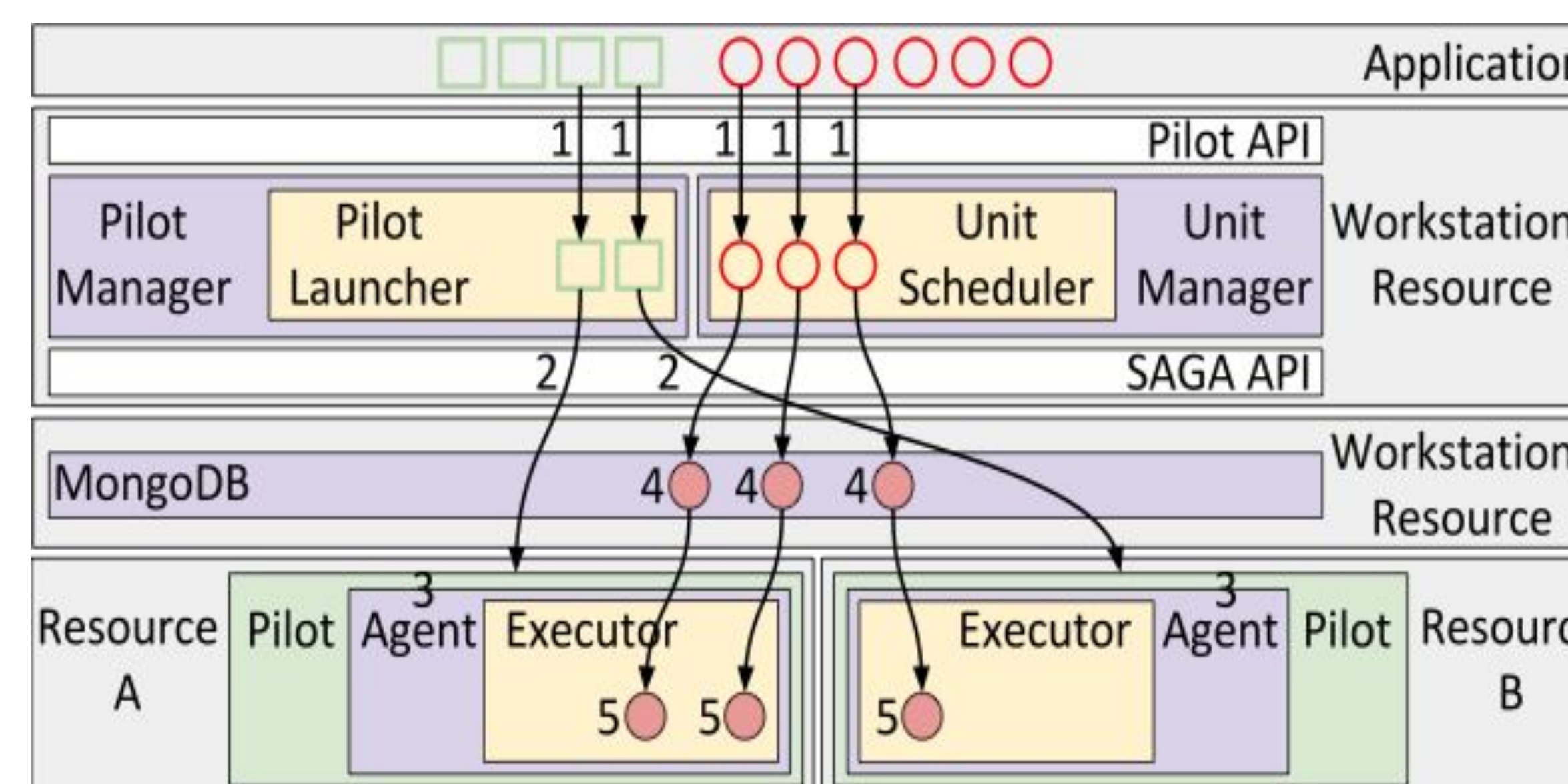
RADICAL-Cybertools are used at each level to support scalable, efficient and effective use of high-performance and distributed computing.

## (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 distributed 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 cyberinfrastructures such as XSEDE and OSG.

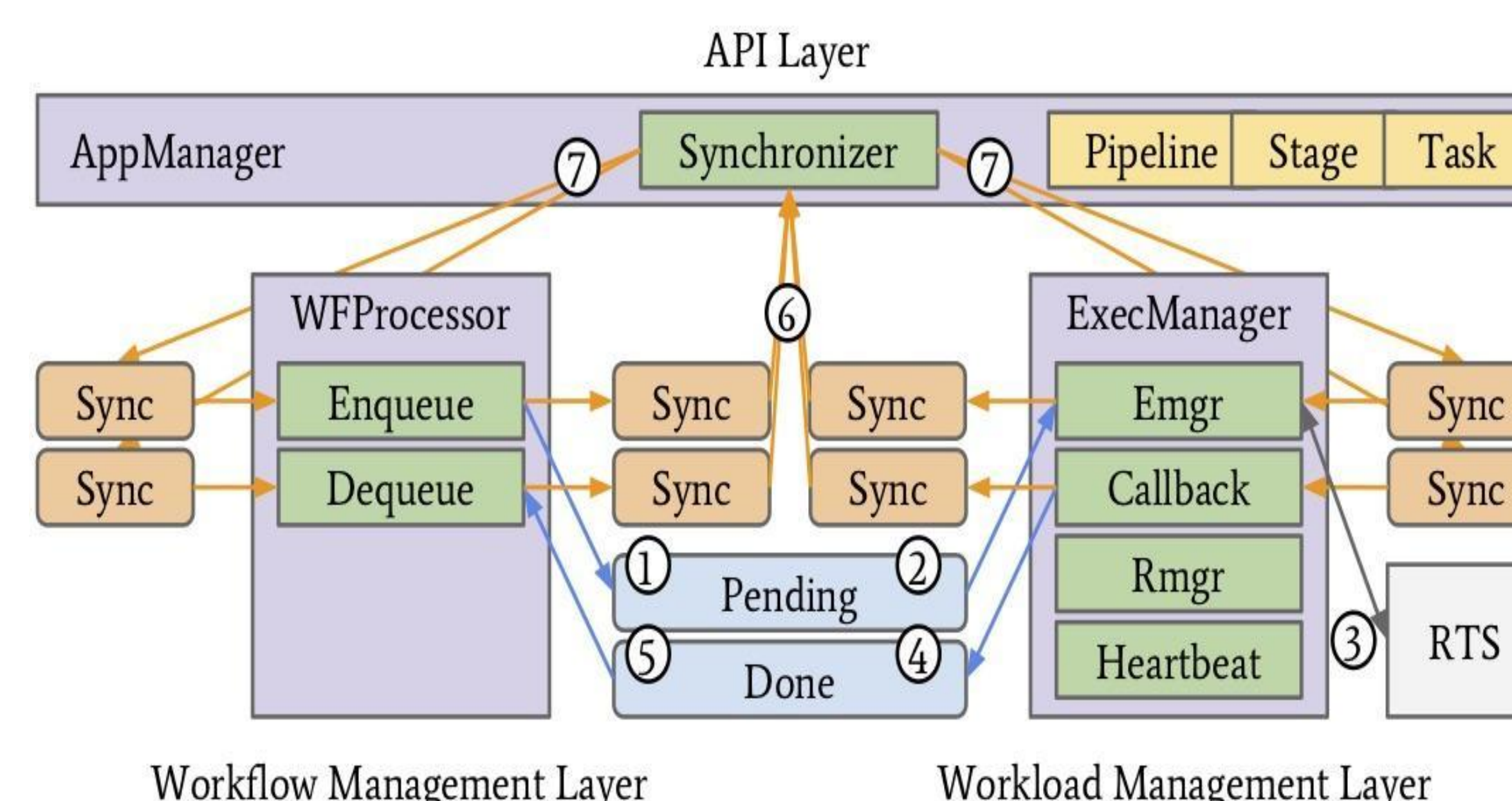
## (L2) Task Runtime Management

**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, multiprocessing, multithreaded, CPU, and GPU tasks—and scheduling O(10k) tasks while marshalling O(10k) distributed cores.



## (L3) Workload Management

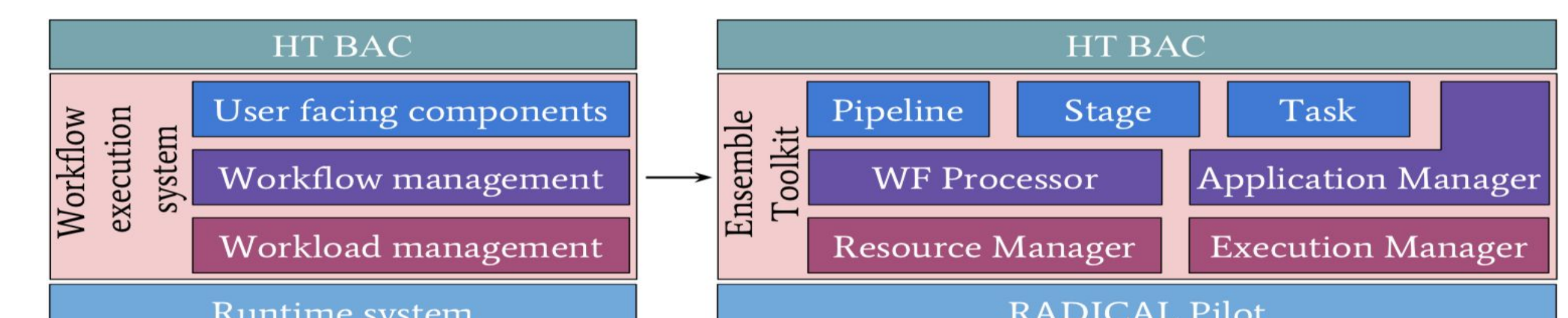
**Ensemble Toolkit:** Provides the ability to execute flexible combinations of **ensemble-based applications** on high-performance distributed computing resources. Ensemble Toolkit takes charge of where and how the workload is distributed: users only have to worry about what to run and when.



## (L4) Applications and Scientific Workflows

**ExtASY:** Enables sampling of complex macromolecules with molecular dynamics. It supports **high-performance** and **high-throughput** execution of molecular dynamic calculations, and analysis tools that provide runtime control over a simulation.

**HTBAC:** Enables the scalable, adaptive and automated calculation of the binding free energy on high-performance computing resources.



**RepEx:** Enables performing Replica Exchange simulations at a scale which is not attainable by stand-alone molecular dynamics applications. It uses RADICAL-Pilot for workload execution.

**ICEBERG:** Enables scalable image analysis on high-performance distributed computing for geoscience research. It provides a library based on extensible building-blocks that allows the integration of frameworks and algorithms seamlessly.

## Integration with existing systems

### 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.
- **Harvester** is a service which provides pilot and workload management to Panda
- 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

- **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