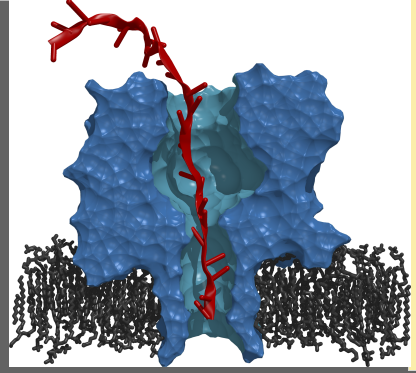
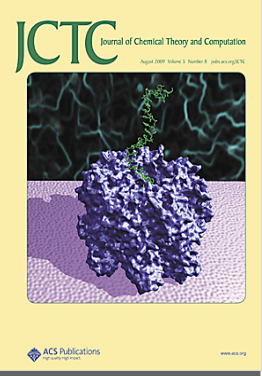
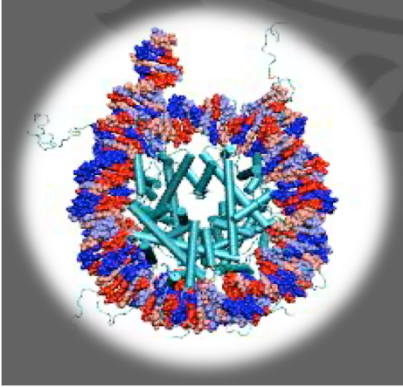
RUTGERS
THE STATE UNIVERSITY
OF NEW JERSEY

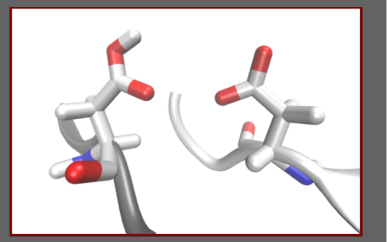




Towards Exascale Through Many Simulations: Abstractions and Tools

Matteo Turilli, Shantenu Jha
http://radical.rutgers.edu





Outline

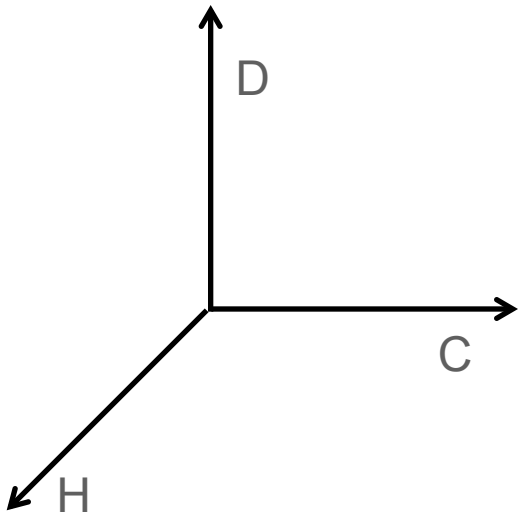
- Exascale program and why many-simulations matter.
- Many-simulations scenarios and examples: Ensemble and Replica-exchange.
- Engineering challenges to support many-simulations scenarios.
- RADICAL models and tools for many-simulation applications.
- The present: P*, BigJob and SAGA-python and how they are used to support ensembles and replica-exchange applications.
- The future: W*, F* and I* and towards and middleware for exascale.

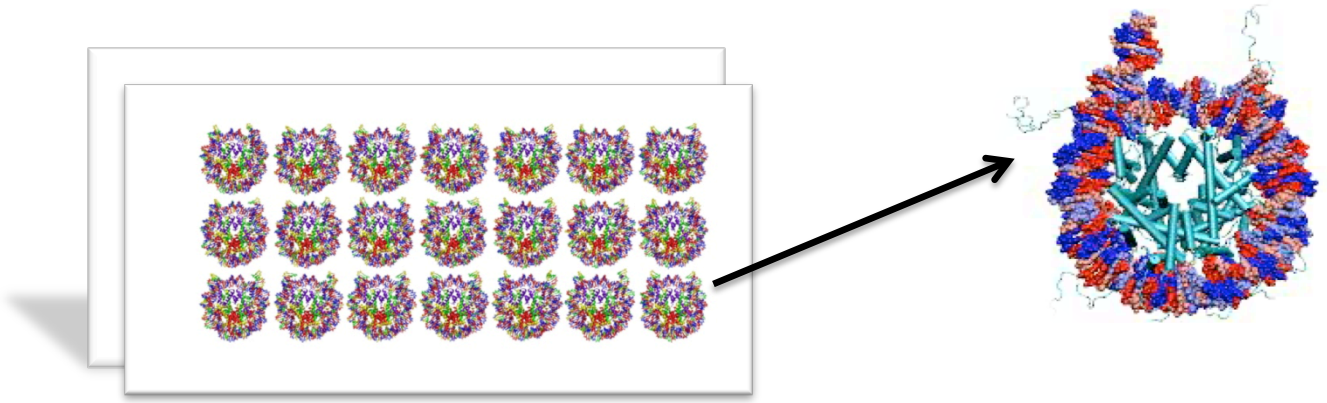
Exascale Computing: View from that side

- US Exascale program alive and active:
  - Exascale programs across agencies DoE/NNSA, DoE, NSF.
- Less tolerance for HPC as an activity of the elite and exclusive club:
  - Even historically HPC oriented agencies (DoE) are looking to support the long-tail of science via more inclusive models of computing.
- Increasing amount of effort spent on modeling and simulation of applications and system.

Many Simulations Pathway to Exascale?

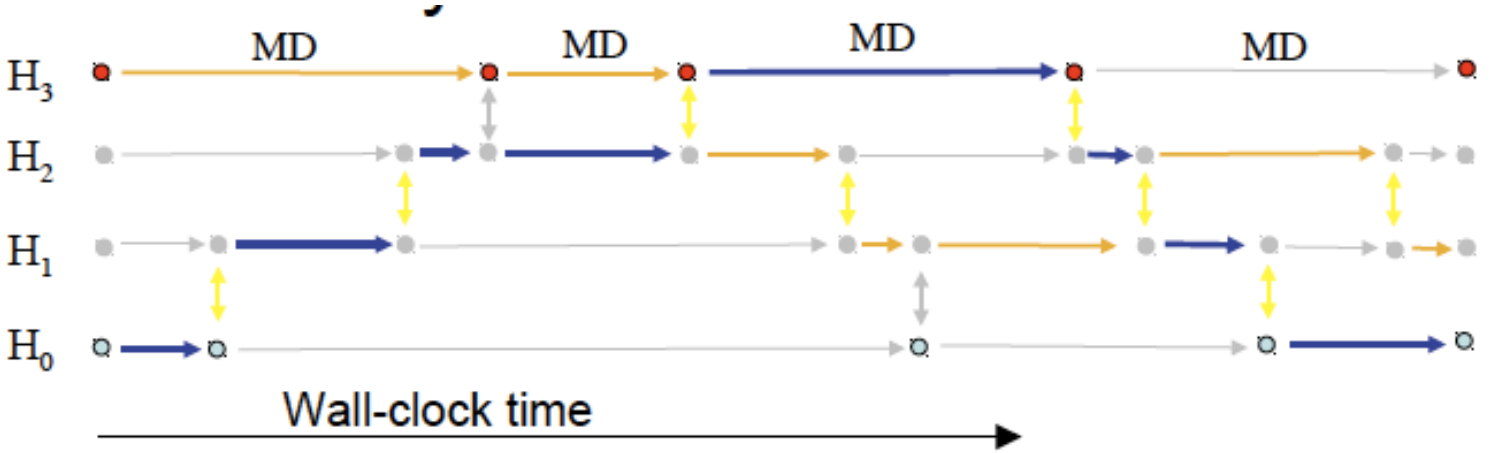
- Problems in computational science *naturally* amenable to "many simulations" model of computing:
  - Many free energy calculations, enhanced sampling problems.
  - Many multi-physics simulations are also multi components.
  - Starting from uncoupled heterogeneous tasks, varying levels of coordination and dependency can be gradually added and "tuned".
- A Single "Application" is broken into many smaller simulations.
- Nature of the simulations:
  - Homogeneous/Heterogeneous (H).
- Coupling between simulations:
  - Coordination (C).
  - Dependencies (D).





Scalable Online Comparative Genomics of Mononucleosomes.

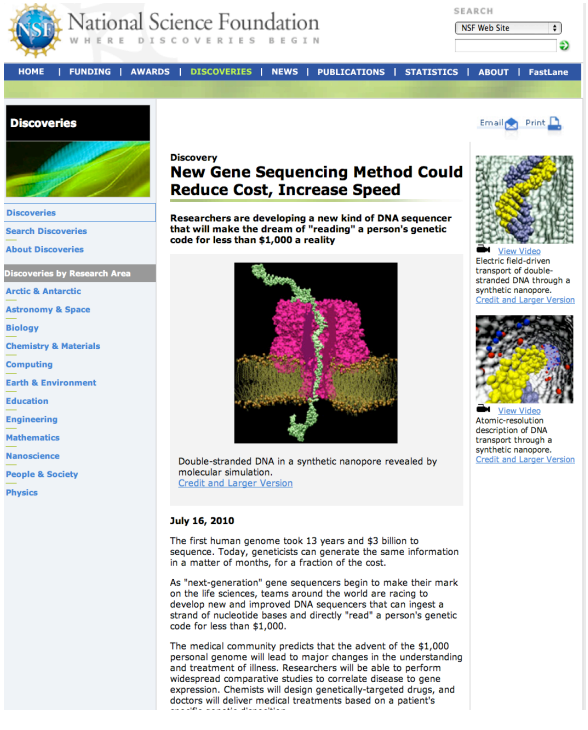
"Scalable Online Comparative Genomics of Mononucleosomes: A BigJob" , *Proceedings of Conference on Exsede*, 2013.

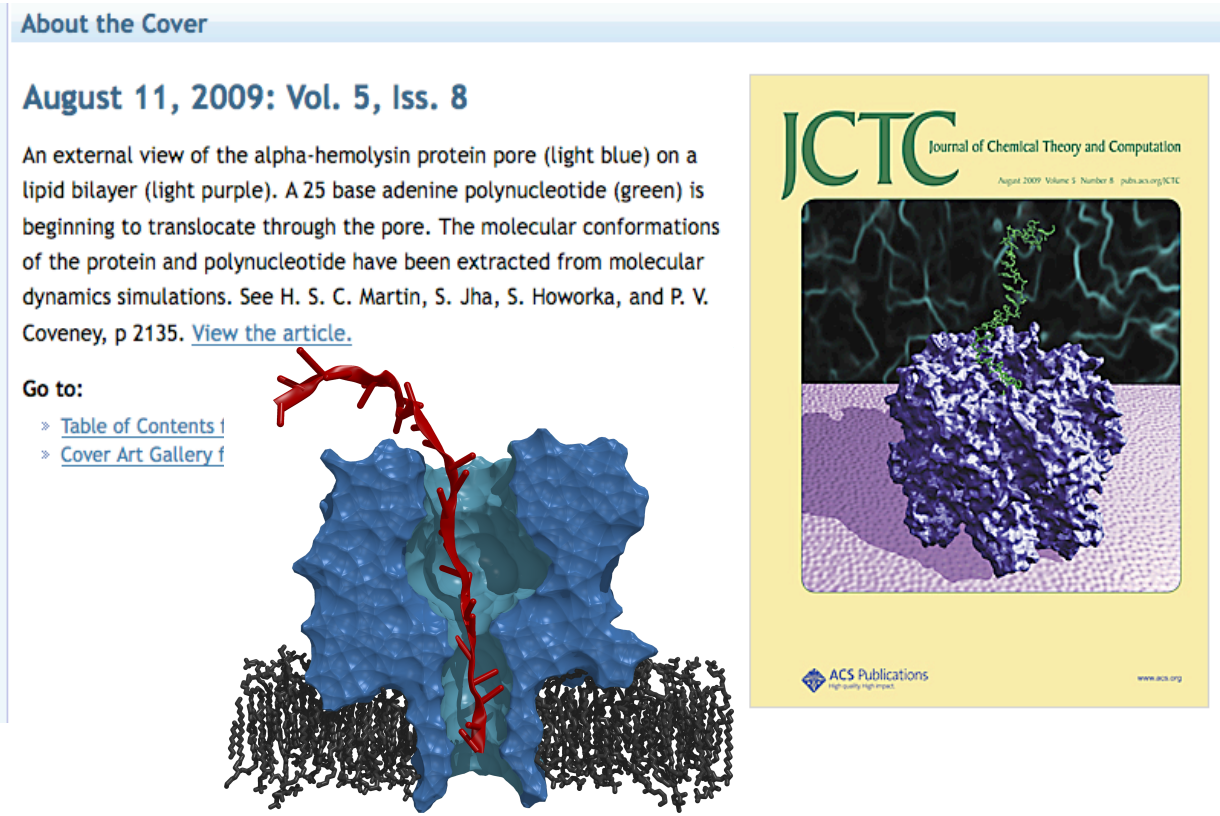


Asynchronous Replica-Exchange: Advanced Algorithms for enhanced sampling.

"A Framework for Flexible and Scalable Replica-Exchange on Production Distributed CI", *Proceedings of Conference on Exsede*, 2013.

A Pore Man's View of the TeraGrid/XSEDE





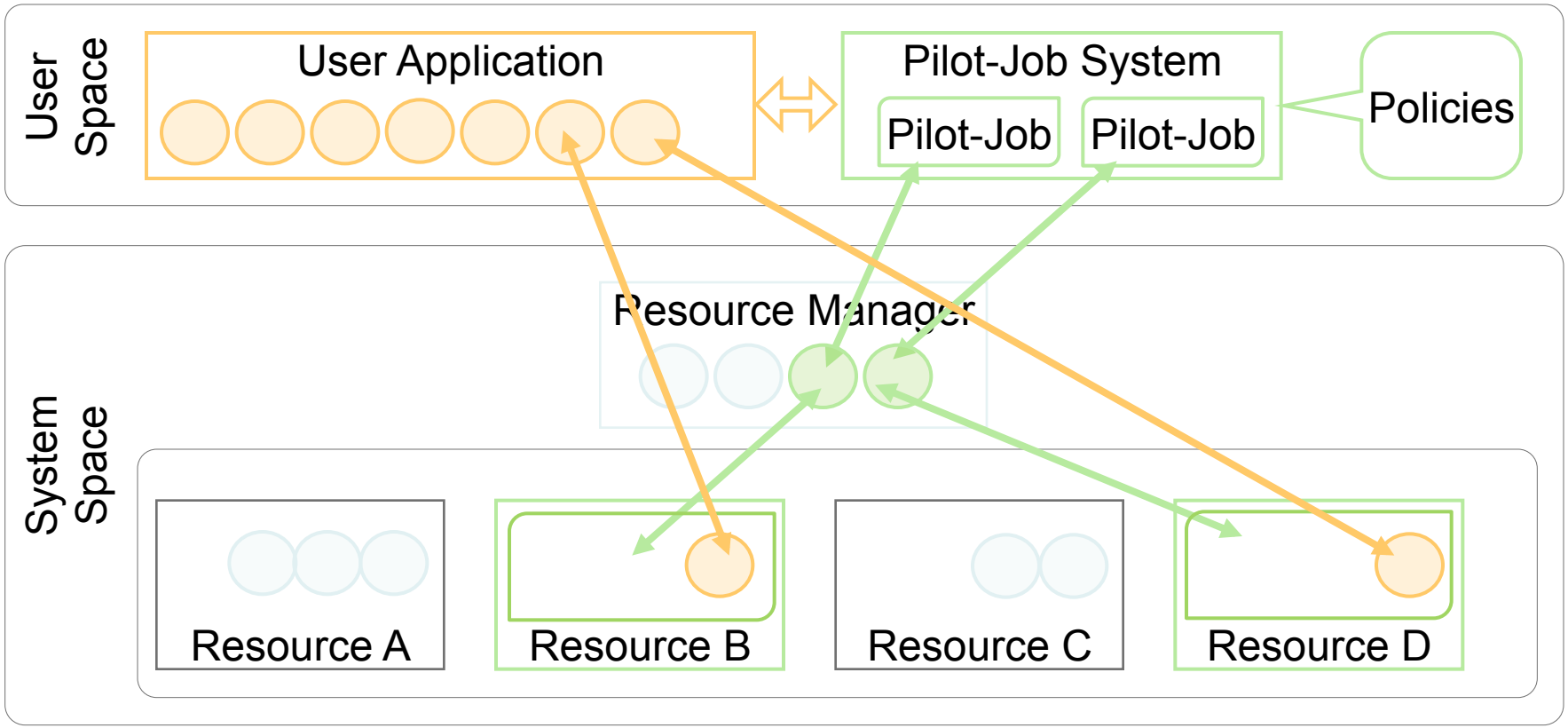
- **2005-09**: Tried running many simulations on many supercomputers. Did not work (well)!
- Why? What has changed?

Challenges of Many Simulations at Exascale

- Many of the “traditional” extreme scale computing:
  - Adaptive application formulation arising from non-deterministic application behavior requires flexible composition;
  - Dynamic resource utilization: Growing/shrinking resource pool.
- Democratization of HPC/Exascale Computing:
  - Capture different modes of extreme-scale computing:
    - Couple exaflops with exabyte: both simulation and analysis.
    - Integrate multiple O(100) PF computing: exascale as an aggregated capability.
- Abstractions:
  - Decouple workload and resource management.
  - Provide capabilities in an extensible and interoperable way by ensuring flexibility with performance, and hiding complexity yet exposing simple interfaces.
- Need to provide these abstractions as well engineered tools:
  - Employ CI best practices and software sustainability.

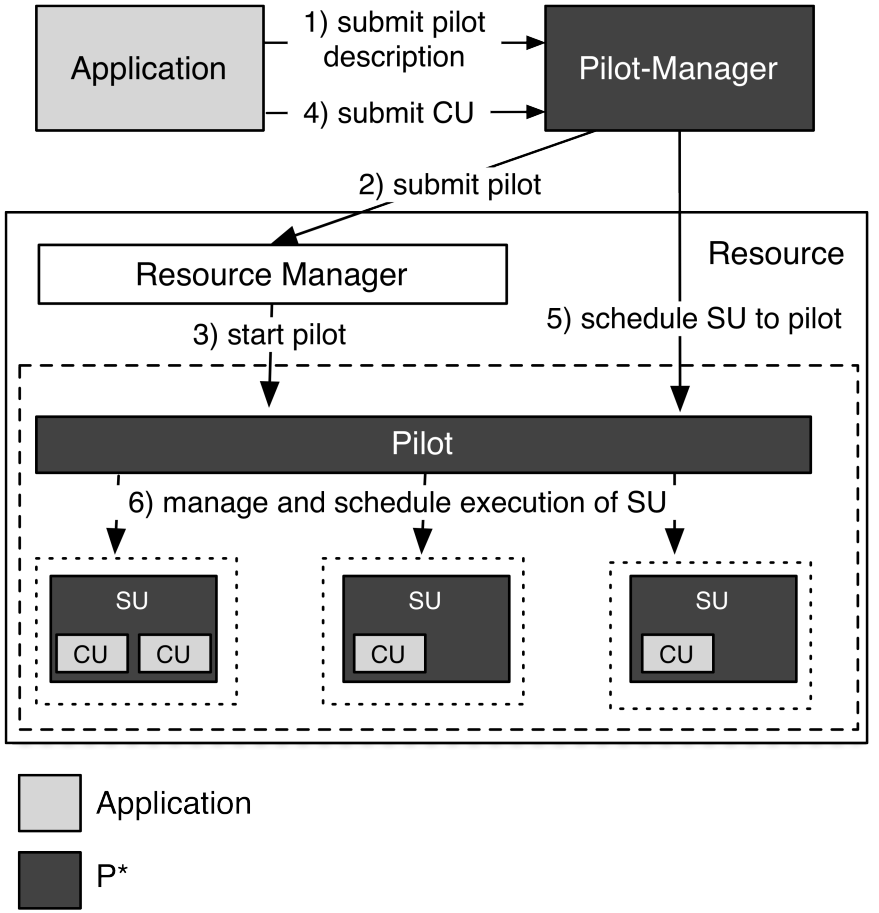
Pilot Abstractions

Working definition: Abstractions to generalize a placeholder job so to allow application-level control over the system scheduler via a scheduling overlay.



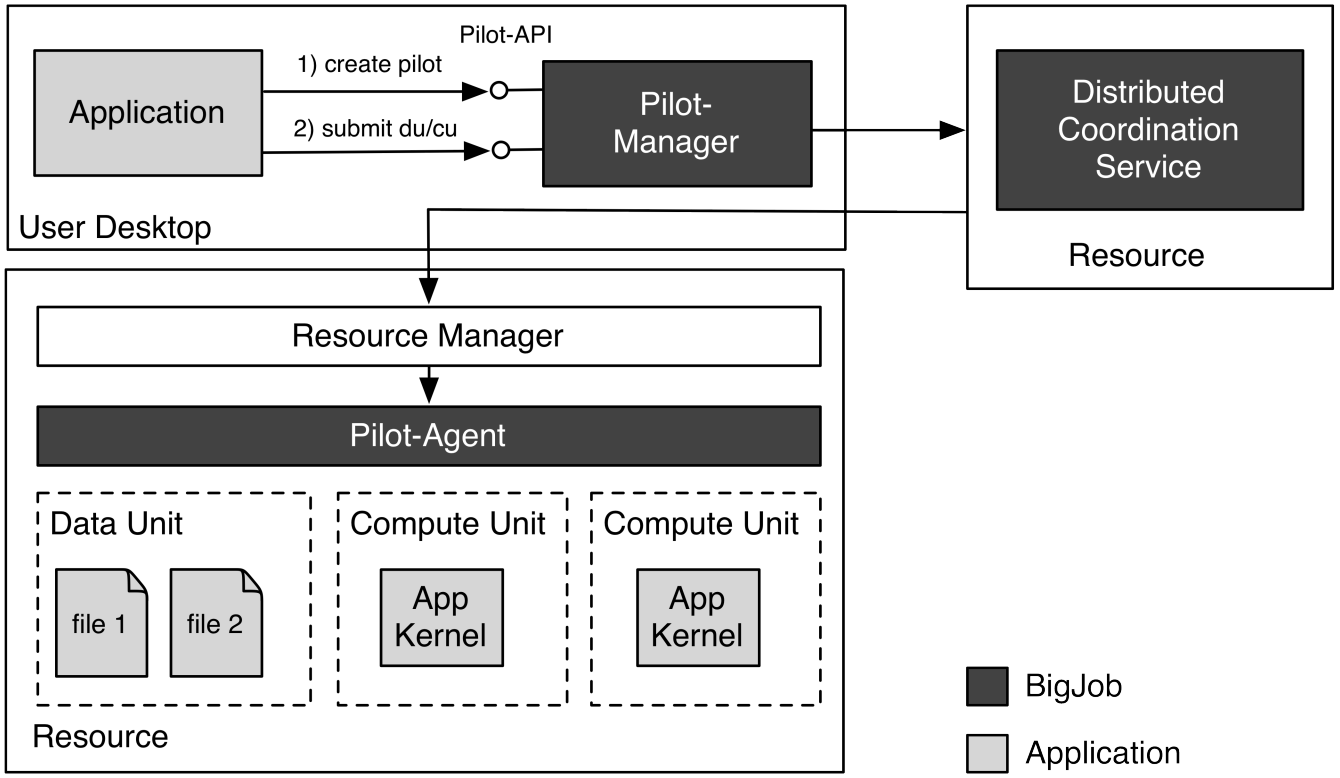
P* Model: Elements, Characteristics and API

- Elements:
  - Pilot-Compute (PC).
  - Pilot-Data (PD).
  - Compute Unit (CU).
  - Data Unit (DU).
  - Scheduling Unit (SU).
  - Pilot-Manager (PM).
- Characteristics:
  - Coordination.
  - Communication.
  - Scheduling.
- Pilot-API.



"P*: A Model of Pilot-Abstractions", *8th IEEE International Conference on e-Science 2012*, 2012

BigJob: Architecture



http://saga-project.github.io/BigJob/

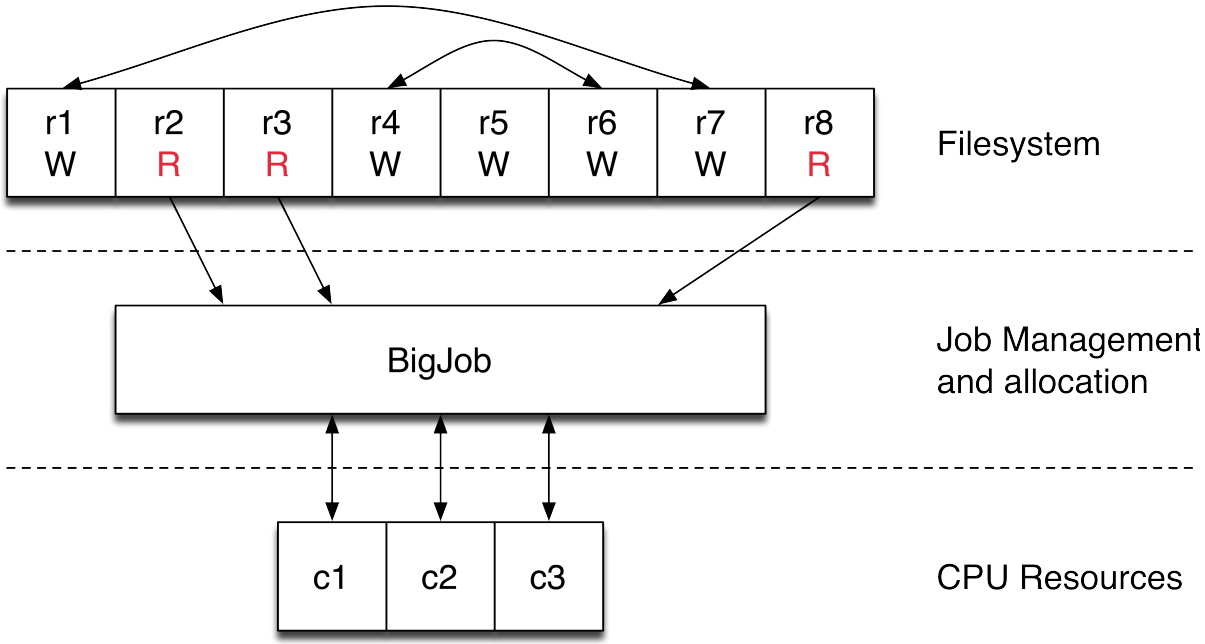
SAGA: Interoperability Layer for BigJob

- SAGA – Simple API for Distributed ("Grid") Applications:
  - Application level standardized (Open Grid Forum GFD.90) API.
  - Application is a broad term: "one person's application is another person's tool (building block)".
- SAGA-Python:
  - Native Python implementation of Open Grid Forum GFD.90.
  - Allows access to different middleware / services through a unified interface
  - Provides access via different backend plug-ins ("adaptors").
  - SAGA-Python provides both a common API, but also unified semantics across heterogeneous middleware:
    - Transparent Remote operations (SSH / GSISSH tunneling).
    - Asynchronous operations.
    - Callbacks.
    - Error Handling.

http://saga-project.github.io/saga-python/

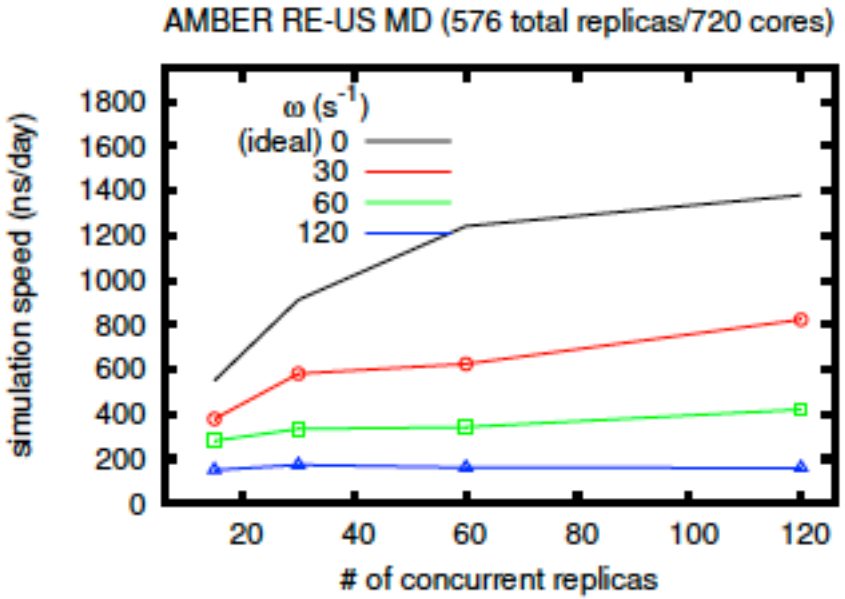
Async Replica-Exchange Package

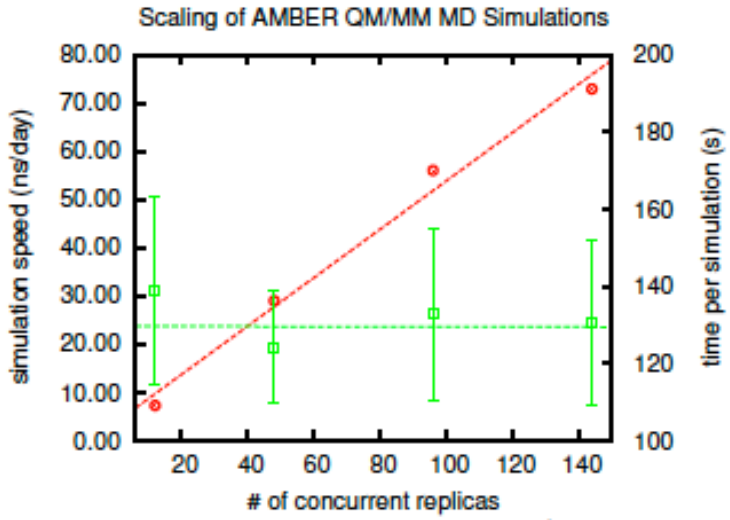
- Python package built to perform file-based asynchronous parallel replica exchange.
- Platform independent library.



https://github.com/saga-project/asyncre-bigjob

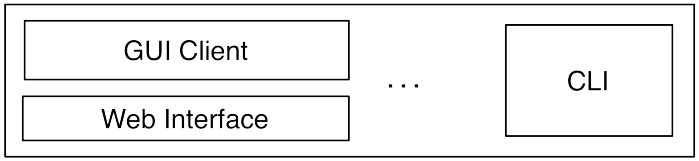
Async Replica-Exchange Package



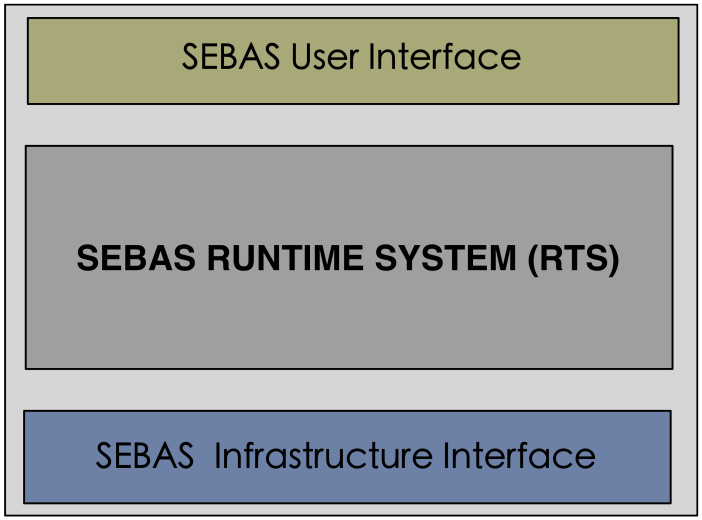


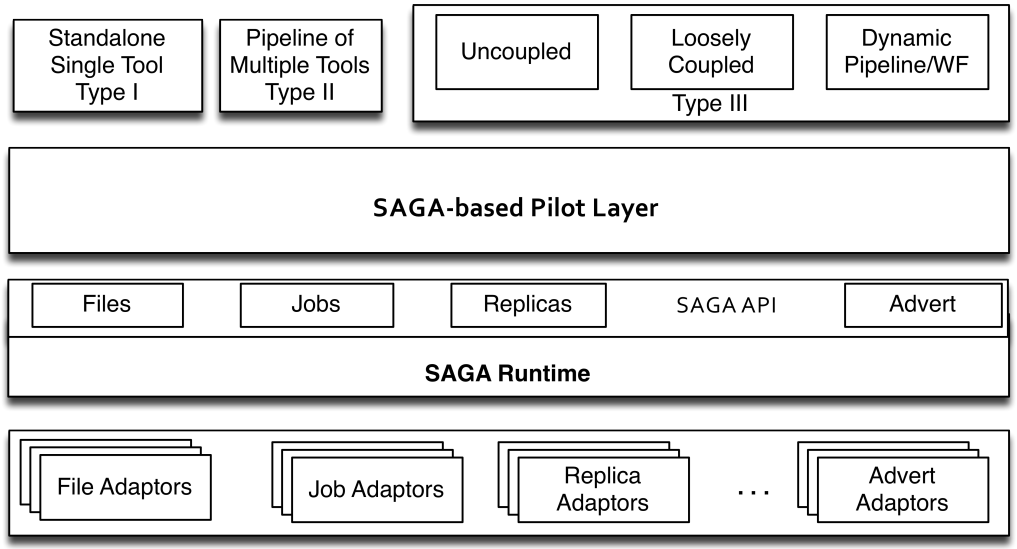
- Ideal performance considered to be zero coupling in this case.
- Diminished results due to coordination overheads.
- Scientists are free to choose the best tradeoff between simulation speed and number of concurrent replicas.
- BigJob-based Repex: Amongst the earliest QM/MM.

Scalable, Extensible HT Binding Energy Calculation



- Platform independent library.
- Suggestions for other libraries are welcome!





Compute and Data Infrastructure
e.g. UK-NGS/Hector, Campus resources, US-XSEDE, Clouds

In consultation with Peter Coveney and Charlie Laughton.

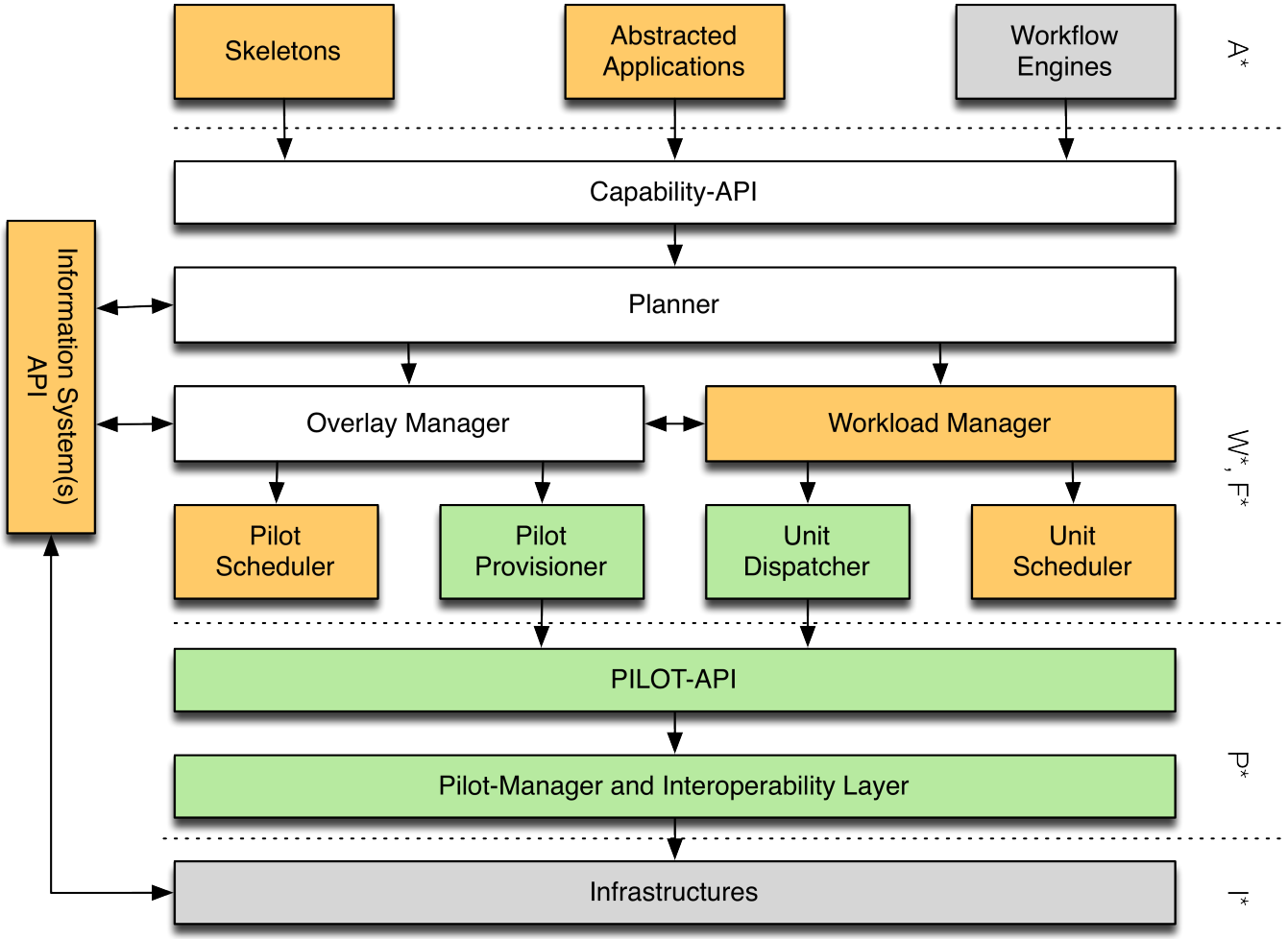
Mapping Simulations to Resources

1. Optimal Characterization – Finding the optimal workload characterization of an application:
   - Generality versus specificity.
   - Class of Applications versus Many application classes/types.
   - Different applications have different metrics.
2. Optimal Federation – Finding the optimal resource configurations for a given workload:
   - Irrespective of whether it is best representation of an application.
   - Resources availability at extreme scale is variable.
3. Optimal Mapping – Workload to Resource Mapping:
   - Static versus Dynamic resource mapping.

Towards an Exascale Middleware

- Workloads (W*)
  - Unified description: Workload representations from multiple applications are reduced into a minimal, common description.
  - Translation of the given workload into compute, data, and network units depending on the type and state of the available resources.
- Overlays and Resources (F*, I*)
  - Automation of the description, instantiation, and management of resource overlays for the given workload based on pilot abstractions.
  - Resource bundles: low-level resource aggregation.
  - Resource federation: high-level capabilities aggregation.
- Workloads and overlays
  - Multiple algorithms available to schedule pilots onto resources and units onto pilots.
  - Decoupling between workloads and overlay utilization: Workflows, minimizing overlay overheads and maximizing overlay utilization.

Overlay and Workload Manager (OWM)



Thank you

- SAGA-Python:
  - http://saga-project.github.io/saga-python/
- BigJob: An implementation of P*
  - http://saga-project.github.io/BigJob/
- RADICAL:
  - http://radical.rutgers.edu/
- Publications:
  - http://radical.rutgers/edu/publications
- Tutorials:
  - https://github.com/saga-project/tutorials/wiki/XSEDE13