SAGA

A Simple API for Grid Applications

A Brief Introduction to SAGA

Andre Mezky, Shantenu Jha

SAGA

A Simple API for Grid Applications

All material from this tutorial can be found at:

https://svn.cct.lsu.edu/repos/saga-projects/tutorial/EGI-2011

General Information and Documentation

- General information
  - http://saga. cct.lsu.edu/
- Documentation:
  - http://saga.cct.lsu.edu/software/cpp/documentation
- API documentation
  - Python
    - http://static.saga.cct.lsu.edu/apidoc/python/latest/
  - C++
    - http://static.saga.cct.lsu.edu/apidoc/cpp/latest/
- Programmers Guide:
  - https://svn.cct.lsu.edu/repos/saga/core/trunk/docs/manuals/programming_guide/tex/saga-programming-guide.pdf

Distributed Applications Development Challenges

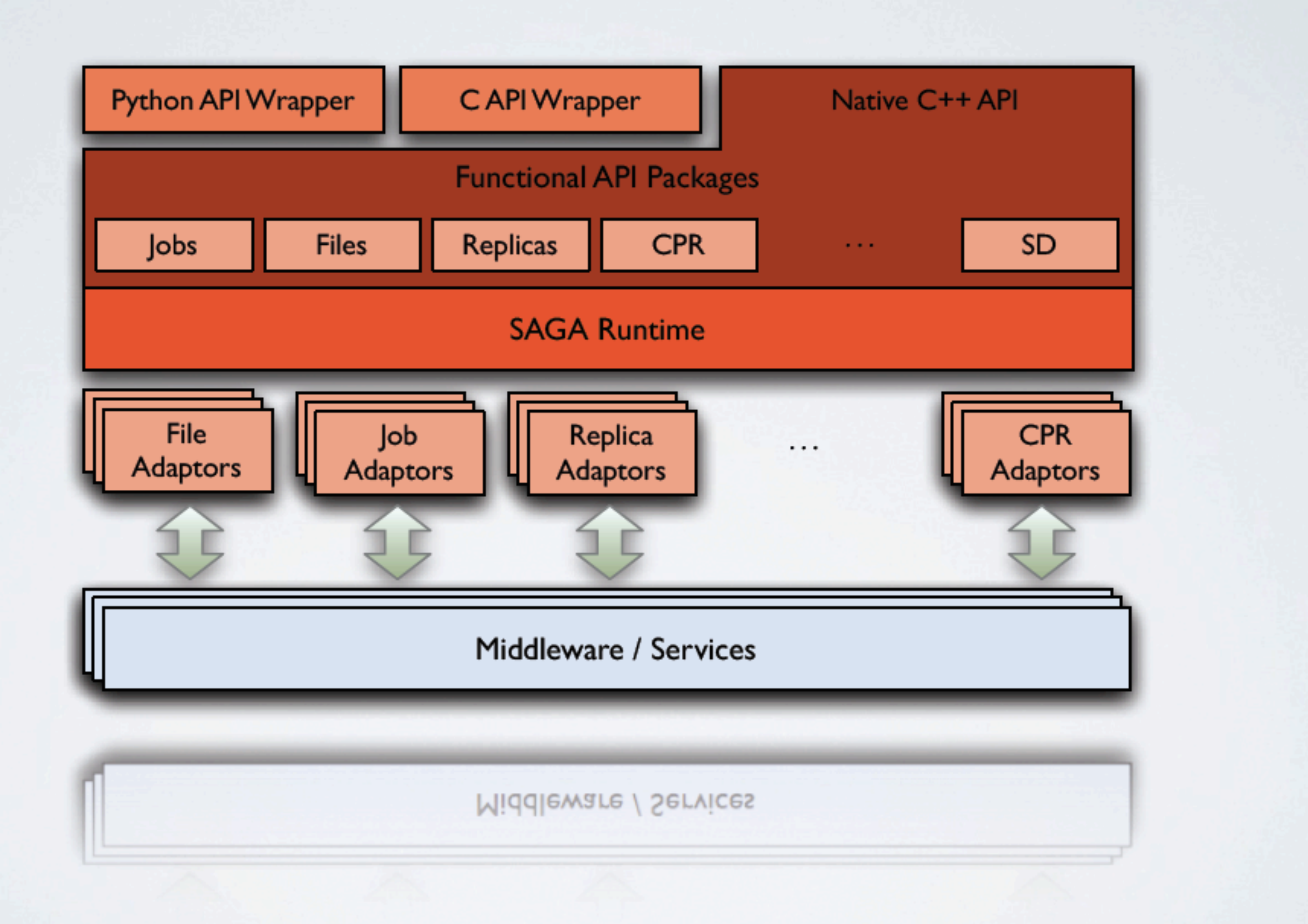
- Developing Distributed Applications is fundamentally hard:
  - Intrinsic:
    - Control/Coordination & execution over Heterogeneous sites
    - Complex Design point/Models of Distributed Applications,
      - Reasons for using distributed CI -- more than (peak) performance result
  - Extrinsic:
    - (Complex) Underlying infrastructure & its provisioning
    - Large number Programming systems, tools and environments
      - Lack of well-defined interfaces & abstractions
      - Interoperability and extensibility become difficult
- Number of "effective" distributed applications that utilize resources sequentially, concurrently or asynchronously is low
  - Distributed CI: Is the whole > than the sum of the parts?
- See: DPA Survey Paper:
  - http://www.cct.lsu.edu/~sjha/dpa_publications/dpa_surveypaper.pdf

SAGA: In a nutshell

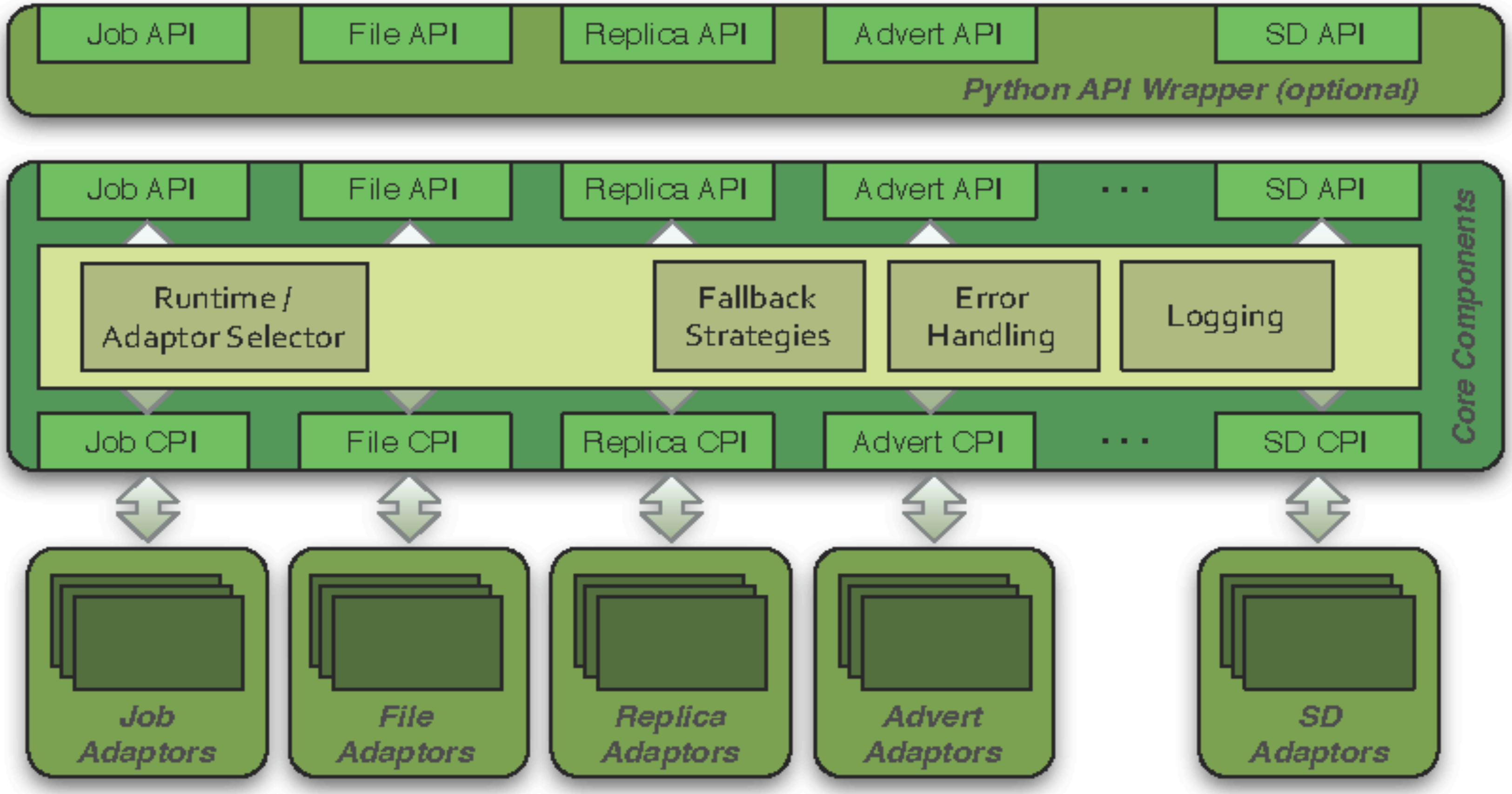
- There exists a lack of Programmatic approaches that:
  - Provide general-purpose, basic &common grid functionality for applications and thus hide underlying complexity, varying semantics..
  - The building blocks upon which to construct "consistent" higher-levels of functionality and abstractions
  - Meets the need for a Broad Spectrum of Application:
    - Simple scripts, Gateways, Smart Applications and Production Grade Tooling, Workflow…
- Simple, integrated, stable, uniform and high-level interface
  - Simple and Stable: 80:20 restricted scope and **Standard**
  - Integrated: Similar semantics & style across
  - Uniform: Same interface for different distributed systems
- SAGA: Provides Application* developers with units required to compose high-level functionality across (distinct) distributed systems

  (*) One Person's Application is another Person's Tool

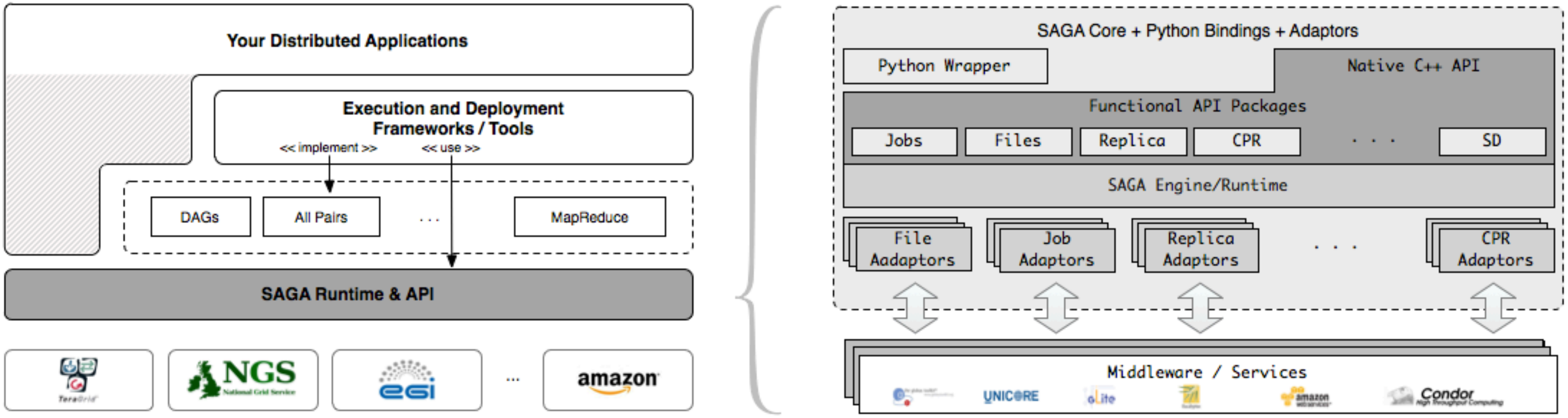
SAGA: In a thousand words



SAGA: Architecture



SAGA: The BigPicture



Your Distributed Applications

Execution and Deployment Frameworks / Tools

<< implement >>

<< use >>

DAGs

All Pairs

. . .

MapReduce

SAGA Runtime & API

TeraGrid

NGS National Grid Service

egi

...

amazon

SAGA Core + Python Bindings + Adaptors

Python Wrapper

Native C++ API

Functional API Packages

Jobs

Files

Replica

CPR

. . .

SD

SAGA Engine/Runtime

File Aadaptors

Job Adaptors

Replica Adaptors

. . .

CPR Adaptors

Middleware / Services

UNICORE

gLite

amazon web services

Condor High Throughput Computing

How is SAGA Used?

- SAGA is used to develop applications that are distributed by definition:
  - Simple extensions of "localized applications" (eg scripting)
    - MW applications, workers submitted to >8 back-ends
  - Novel Distributed Programming Models (eg Rep-Exch)
- SAGA: Build tools and implement abstractions that enable the execution of applications over distributed resources, *without modifying the applications*
  - Eg. Infrastructure Independent Pilot-Jobs
- SAGA: To provide uniform access layers to heterogeneous CI
  - Uniform access to EGI (ARC, gLite, Globus and Unicore/BES)
  - Simplify the building of tools and Gateways