SAGA-Python

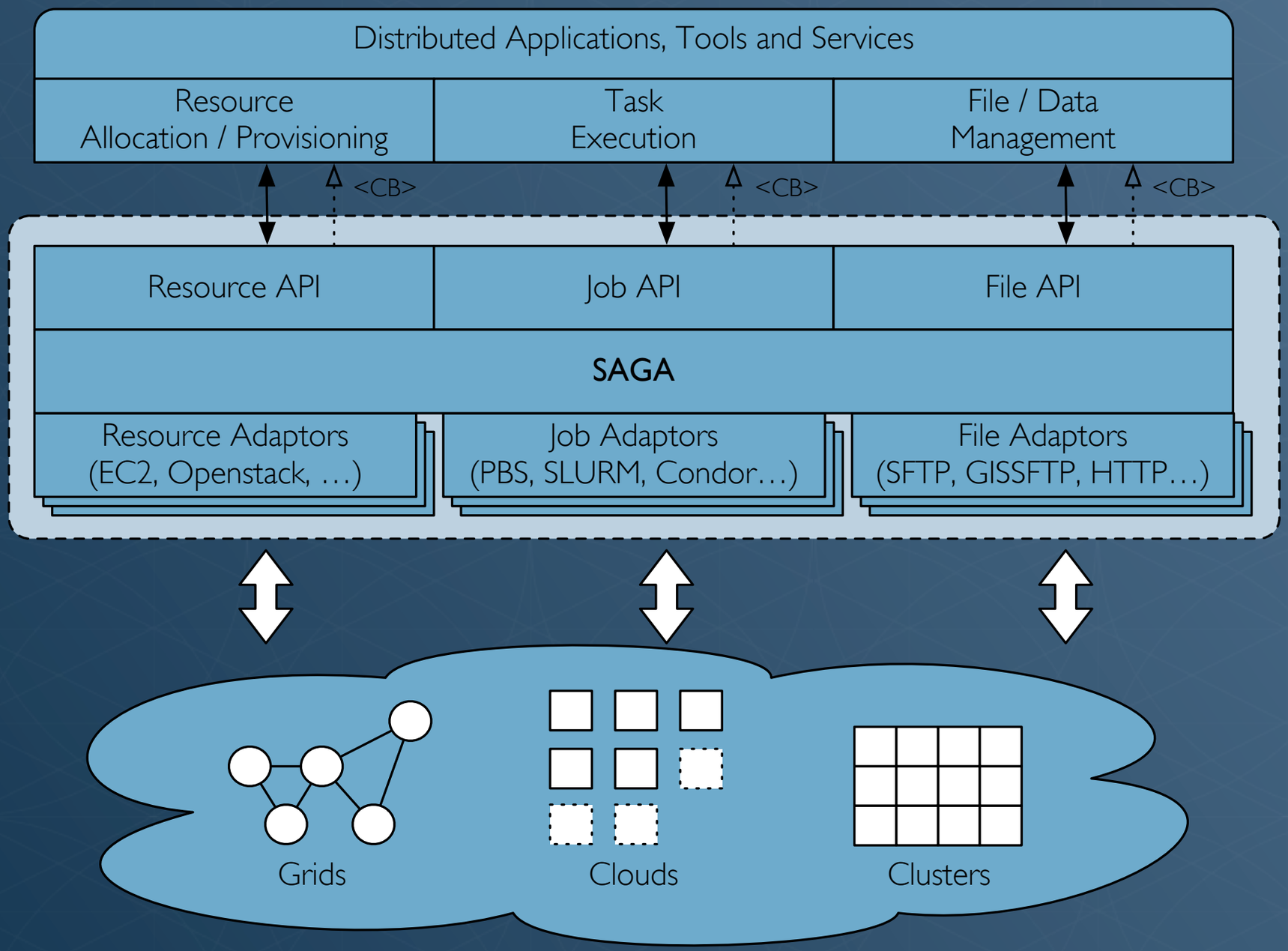
A Light-Weight Python Access Layer for Distributed Computing Infrastructure

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

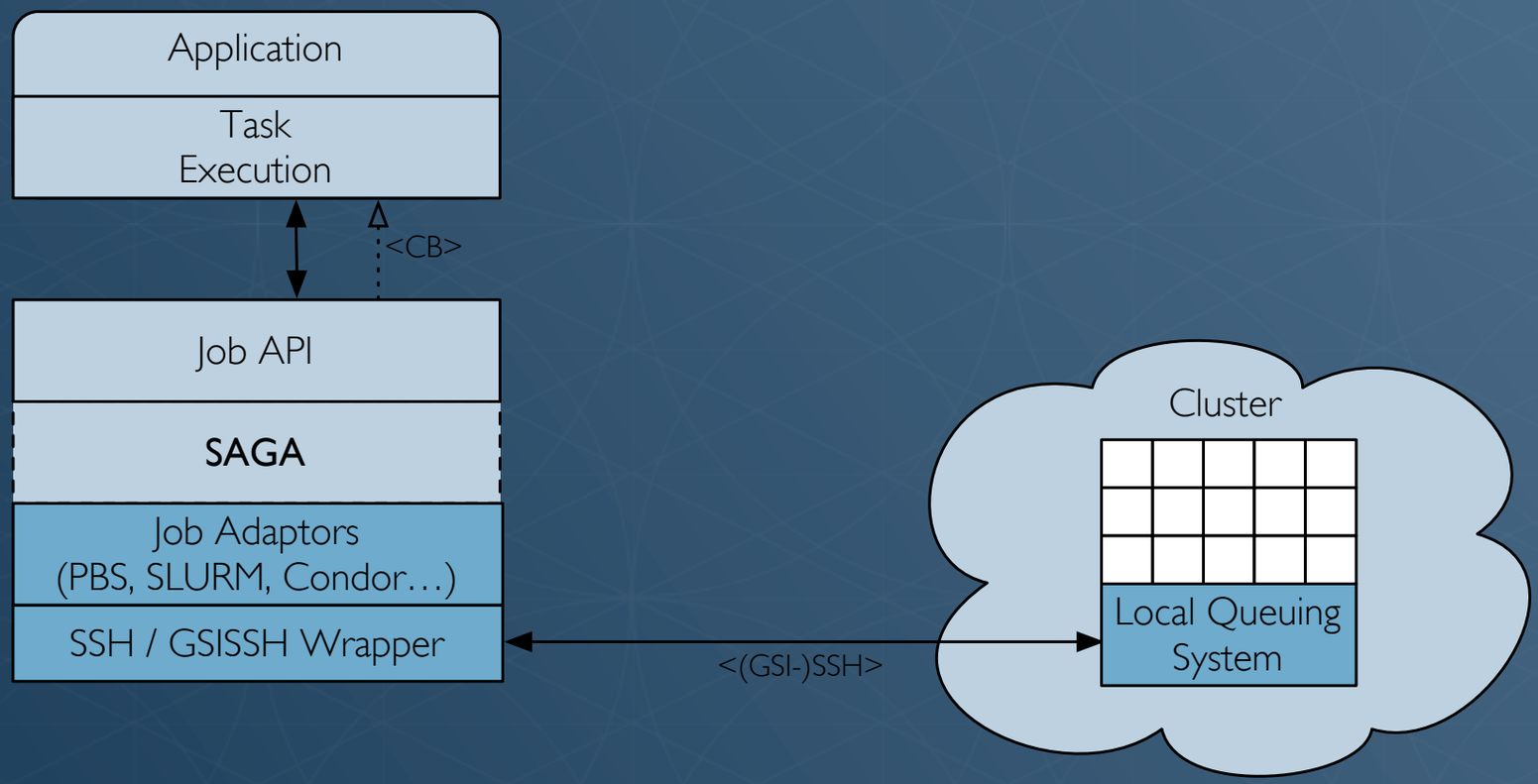
INTRODUCTION

- SAGA - A Simple API for Distributed ("Grid") Applications
- Native, open source (MIT license) Python implementation of Open Grid Forum GFD.90 interface standard
- Unified interface for resource allocation, job, file and data management in a distributed environment
- Unified semantics across heterogeneous middleware
  - Transparent remote operations, asynchronous operations / callbacks, error handling

OVERVIEW

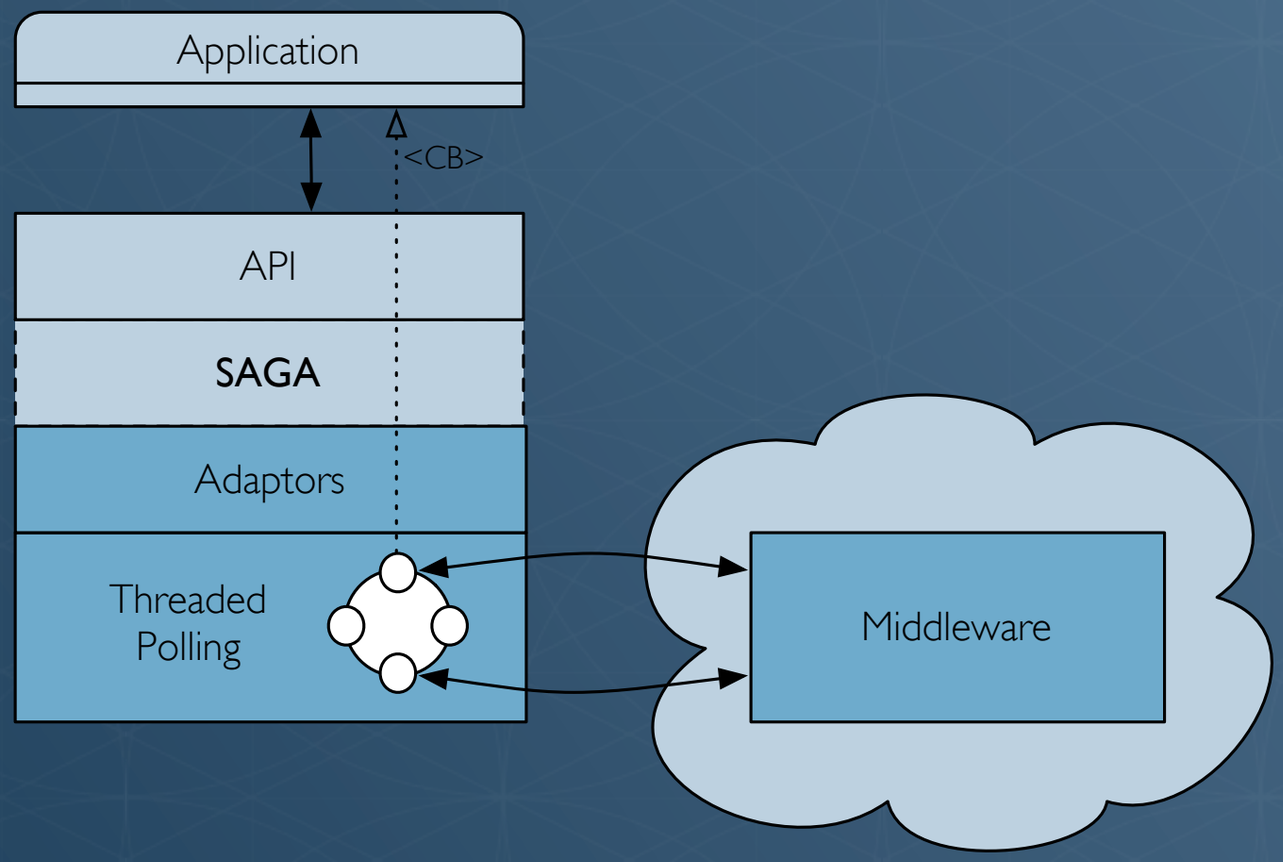

Distributed Applications, Tools and Services
Resource Allocation / Provisioning
Task Execution
File / Data Management
<CB>
<CB>
<CB>
Resource API
Job API
File API
SAGA
Resource Adaptors (EC2, Openstack, …)
Job Adaptors (PBS, SLURM, Condor…)
File Adaptors (SFTP, GISSFTP, HTTP…)
Grids
Clouds
Clusters

TRANSPARENT REMOTE OPERATIONS



- Fast and optimized (GSI-)SSH command transport wrapper can be used by adaptors to access otherwise 'local-only' services, like many queuing systems

TRANSPARENT CALLBACKS



- Callbacks and asynchronous operations are important concepts in distributed applications but are not supported by all middleware systems
- SAGA-Python moves application-level polling into the adaptor layer and exposes a clean callback interface through the API

AVAILABLE ADAPTORS

- Job Submission Systems
  - SSH, GSISSH, Condor, Condor-G, PBS(-Pro), TORQUE, SGE, SLURM. Under development: LSF
- File / Data Management
  - SFTP, GSIFTP, HTTP, HTTPS. Under development: iRODS, Globus Online
- Resource Management / Clouds
  - Amazon EC2, Openstack ('libcloud'-based)

CODE EXAMPLE

```
def job_state_change_cb(src_obj, fire_on, value):
    print "Callback: job state changed to '%s'\n" % value
    return True

def application():
    ec2_ctx = saga.Context('EC2')
    ec2_ctx.user_id = os.environ['EC2_ID']
    ec2_ctx.user_key = os.environ['EC2_KEY']

    ssh_ctx = saga.Context('SSH')
    ssh_ctx.user_id = 'ubuntu'
    ssh_ctx.user_key = os.environ['EC2_SSH_KEYPAIR']

    session = saga.Session(False)
    session.contexts.append(ec2_ctx)
    session.contexts.append(ssh_ctx)

    try:
        # Launch a VM on Amazon EC 2
        rm = saga.resource.Manager("ec2://aws.amazon.com/",
            session=session)

        cd = saga.resource.ComputeDescription()
        cd.image = 'ami-0256b16b'
        cd.template = 'Small Instance'

        cr = rm.acquire(cd)
        cr.wait(saga.resource.ACTIVE)
                                                    |-->
```

```
        # Copy a file to the VM (via SFTP)
        file = saga.filesystem.file('file://localhost/data/e_coli')
        file.copy(cr.access+'/home/ubuntu/bowtie/run01/')

        # Run a task on the VM (via SSH)
        js = saga.job.Service(cr.access, session=session)
        jd = saga.job.Description()
        jd.executable = '/opt/apps/bowtie/bin/bowtie'
        jd.arguments = ['-a', '-m 3', '-v 2 e_coli',
                        '-c ATGCATCATGCGCCAT']
        jd.working_directory = '/home/ubuntu/bowtie/run01/'

        job = js.create_job(jd)
        # register a callback with the job
        job.add_callback(saga.STATE, job_state_change_cb)

        # Run the job and wait for it to complete
        job.run()
        job.wait()

    except saga.SagaException, ex:
        # Catch all saga exceptions
        print "Error: (%s) %s " % (ex.type, (str(ex)))

    finally:
        # shut down the VM
         cr.destroy()
```

RESOURCES

Website

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

GitHub Code Repository

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

Google Groups

https://groups.google.com/forum/#!forum/saga-users

https://groups.google.com/forum/#!forum/saga-devel

Twitter

https://twitter.com/SAGAGridAPI