# SAGA: API Layers – Shell, Python, C++

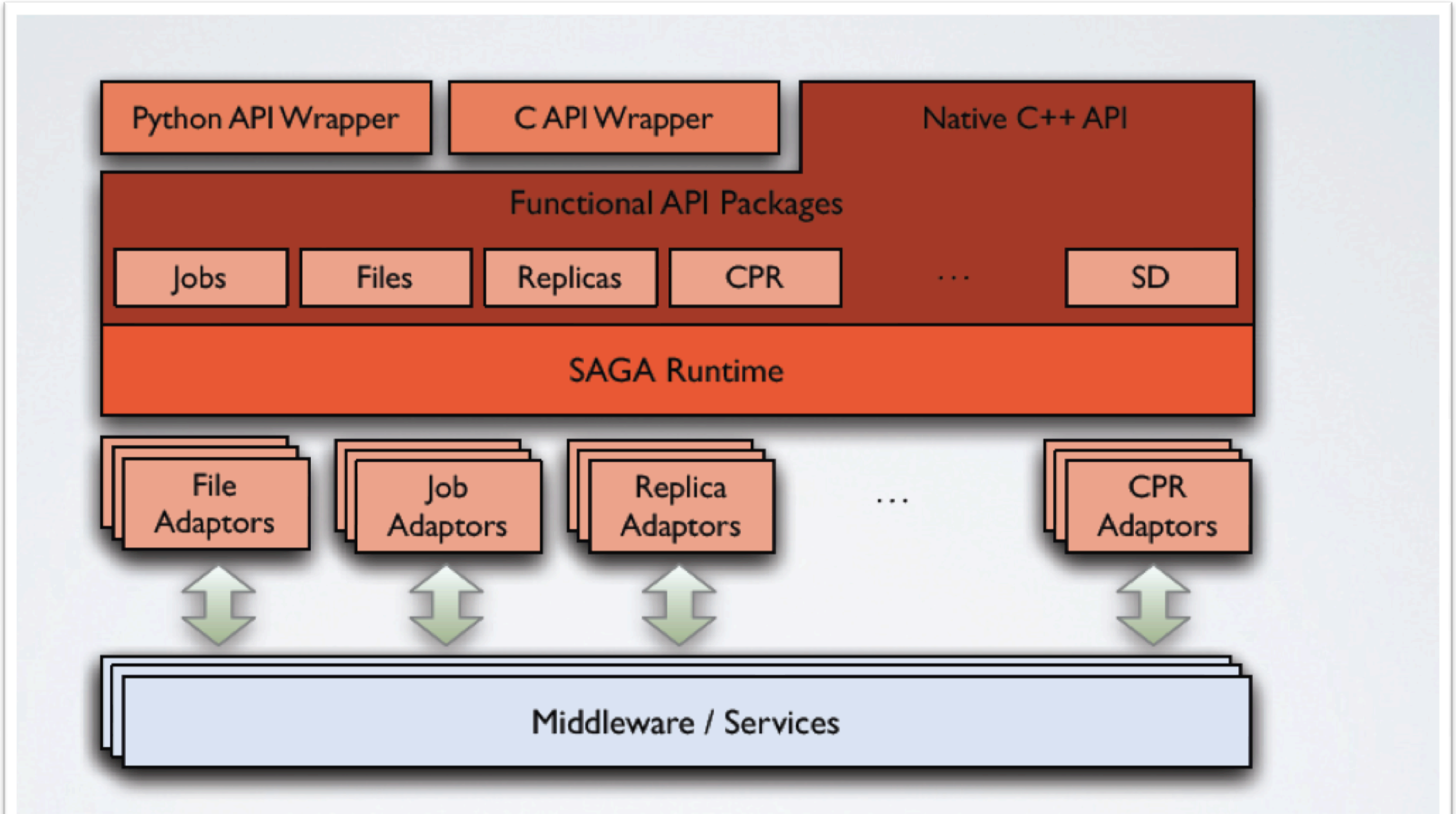
## Agenda

- SAGA command line tools
- SAGA Python API
- SAGA C++ API
- Examples

## Documentation

- General information
  - http://faust.cct.lsu.edu/trac/saga/wiki/... FIXME
- API documentation
  - Python
    - http://static.saga.cct.lsu.edu/apidoc/python/latest/
  - C++
    - http://static.saga.cct.lsu.edu/apidoc/cpp/latest/
- Programmers manual
  - http://static.saga.cct.lsu.edu/docs/programming_guide/html/saga-programming-guide.html

## SAGA: Architecture



| | Local Adaptors | Globus Adaptors | SSH Adaptors | . |
|---|---|---|---|---|
| | file://localhost/...<br>any://localhost/... | gram://remotehost/...<br>any://remotehost/... | ssh://remotehost/...<br>any://remotehost/... | |
| ell | saga-file copy src dest | | | |
| | saga-job run rm cmd | | | |
| ython | import saga.filesystem<br>dir.copy(src, dest) | | | |
| | import saga.job<br>js.run(cmd) | | | |
| ++ | using saga::filesystem::directory;<br>dir.copy(src, dest) | | | |
| | using saga::job::job; | | | |

## Command line tools

- SAGA comes with simple command line tools that allow t access basic package functionality.

- The source code is very simple and a great starting point explore the SAGA package APIs:

| | |
|---|---|
| saga-file | $SAGA_ROOT/saga/tools/clutils/file/ |
| saga-job | $SAGA_ROOT/saga/tools/clutils/job/ |
| saga-advert | $SAGA_ROOT/saga/tools/clutils/advert/ |
| saga-shell | $SAGA_ROOT/saga/tools/shell/ |

## Command line tools

- 'Shell bindings'
  - Package specific (file, job, advert, replica)
- SAGA shell
  - All in one solution
  - Filesystem navigation (filesystem, advert, replica)
  - Job launching
  - Scripting

## Command line tool: saga-file

- Supported protocols
  - Depends on SAGA adaptors
  - Also available: Globus GridFTP, Curl (subset), KFS, Amazon EC2, Opencloud (Sector/Sphere), Hadoop (HDFS)
- Supported commands:

| Command | Arguments |
|---|---|
| copy | <url from>  <url to> |
| move | <url from>  <url to> |
| remove | <url> |
| cat | <url> |
| ist_dir | <url> |

## Command line tool: saga-job

- Supported protocols
  - Depends on SAGA adaptors
  - Also available: Globus Gram, Condor, OMII-GridSAM, LSF, Amazo EC2, Opencloud (Sector/Sphere)
- Supported commands:

| Command | Arguments |
|---|---|
| run | <rm url> <command> <arguments> |
| submit | <rm url> <command> <arguments> |
| state | <rm url> <jobid> |
| suspend | <rm url> <jobid> |
| resume | <rm url> <jobid> |

## Command line tool: saga-advert

- What is it?
  - Central data store with
    - Hierachical keys
    - Attributes
  - Filesystem like structure
- Supported protocols
  - Depends on SAGA adaptors
  - Local adaptor:
    - Local backend: SQLite3
    - Remote backend: PostgreSQL
  - Also available: Hadoop H-Base, Hypertable

## Command line tool: saga-advert

| ommand | Arguments |
|---|---|
| st_directory | <advert-url> <pattern> |
| dd_directory<br>emove_directory | <advert-url> |
| dd_entry<br>emove_entry | <advert-url> |
| ore_string | <advert-url> <string> |
| etrieve_string | <advert-url> |
| st_attributes | <advert-url> |
| et_attribute | <advert-url> <key> <value> |
| emove_attribute | <advert-url> <key> |

## Command line tool: saga-shell

- All in one of all command line tools as mentioned earlier
- Keeps context in between commands
- Navigate (remote) filesystems (advert, replica too!)
- Launch (remote) jobs, uses io redirection to access in/out
- All commands are implemented using SAGA

## Command line tool: saga-shell

| ype | Commands |
|---|---|
| ile system navigation | pwd, ls, mv, cp, cd, mkdir, rmdir, touch, cat |
| ob package | run, suspend, resume, kill, status, ps |
| eplica | rep_find,  rep_list, rep_add, rep_remove, rep_update, rep_replicate |
| nvironment | setenv, getenv, env |
| ermissions | add_proxy, remove_proxy |

## Python API Example: File Package

- Copy a file

```
using saga
src = url(' … ')
dst = url(' … ')
f = filesystem.file(src, filesystem.ReadWrite)
f.copy(dst)
```

## Python API Example: File Package

- Get a directory file listing

```
using saga
src = url(' … ')
d = filesystem.directory(src)
names = d.list('*')
for name in names:
    ns = saga.name_space.entry(name)
    if ns.is_dir():                      print 'd ',  name
    elif ns.is_link():     print '->', ns.read_link()
    else:                                print '  ', name
```

## Python API Example: Job Package

- Submit a job
- FIXME

## Python API Example: Advert Package

- Create and modify an advert entry

```
# host A
using saga
name = url(' ... ')
e = advert.entry(name, advert.ReadWrite | advert.Create)
e.set_attribute('started', ' ... ' )

# host B
using saga
name = url(' ... ')
e = advert.entry(name)
started = e.get_attribute('started')
```

## C++ API Example: File Package

- Copy a file

```
saga::url src (' ... ');
saga::url dst (' ... ');
saga::file f(src, saga::filesystem::ReadWrite);
f.copy(dst);
```

## C++ API Example: File Package

- Get a directory file listing

```
saga::url src (' … ');
saga::directory d (src);
std::vector<std::string> names = d.list('*');

for (auto it = names.begin(); it != names.end(); ++it) {
        saga::name_space::entry ns (*it);
        if (ns.is_dir())                          cout << 'd ' <<  *it << '\n'
        else if (ns.is_link())           cout << '->' << ns.read_link() <<
'\n';
        else:                                     cout << '  ' << *it << '\n';
}
```

## C++ API Example : Job Package

- Submit a job
- FIXME

## C++ API Example: Advert Package

- Create and modify an advert entry

## Programmers Guide

- Set of very small and easy examples, on
  for each package/paradigm
  - file_copy, file_copy (async)
  - Error handling
  - Attributes
  - Stream (server/client)

## Example 1: hello_world

- Hello world
  - Launch 3 jobs on different machines
    - Execute “/bin/echo”
  - No job dependency
  - Each job returns its passed input argument
    - "Hello"
    - "distributed"
    - "world!"
  - Jobs are launched in parallel (in separate threads)
  - As soon as result is collected it's printed on loc console

## Example 1: hello_world

- Hello world
  - Arbitrary sequence of results
    - Optimally: "Hello distributed world!"
  - Demonstrates
    - How to launch a remote job using SAGA job_service
    - Pass arguments using the command line
    - Collect result by output redirection

- The source code can be found here (see ‘Example1’):
  - http://faust.cct.lsu.edu/trac/saga/wiki/ FIXME
  - The example uses localhost to spawn childs
  - For remote execution change HOST1, HOST2, HOST3 from "localhost" to "FIXME"

## Example 2: chaining_jobs

- Launch 3 jobs on 3 different machines
- Output of previous job is needed to launch next job
- Simple sequential execution, but SAGA style
- Demonstrates
  - How to launch a job using SAGA job_service
  - How to feed input to launched job
  - How to collect output
- Launched job: /usr/bin/bc
- Increment the number passed as the argument
  - Pass returned incremented number to next job

## Example 3: depending_jobs

- Coordinating information from advert service
- Launch a single job sequentially on a set of remote resources
  - Simulating checkpointing/relaunching on different resource (migration)
- Maintain a single result value in advert service
  - Gets written by one job, and read by the next
- Demonstrates
  - How to launch remote job using SAGA job, while maintaining environment
  - Assembling argument lists
- Result is left in advert service, but accessed afterwards