

#### A laissez-faire SAGA Implementation

Bliss Intro | 11/02/11 | Ole Weidner

### Preamble

- Bliss is <u>not</u> a replacement for SAGA C++ and the Python Bindings and is <u>not</u> striving to be
- I wrote it to get *my own* research in dynamic and autonomic distributed computing and applications off the ground

## Design Goals

- Pure Python Implementation (2.4 +)
- Pythonic API Feeling
- Simple Installation and Deployment
- Simple Adaptor Development
- Demand-Driven v.s. Standard-Driven
- An Agile Playground for New Ideas!

### Metrics

	SAGA C++	Python Bindings	A Few Adaptors*
Total Physical Source Lines of Code (SLOC)	87,693 (+66,000 Ext.)	6,203	13,551
Estimated Development Effort ( in Person-Years )	39.54	1.36	2.94
Estimated Development Cost (in US-\$)	\$5,341,072	\$183,600	\$397,261

Generated with SLOCCount: http://www.dwheeler.com/sloccount/

\* Condor, Globus & SSH

Linux 1.0.0 (March 1994) had a SLOC count of 176,250

http://en.wikipedia.org/wiki/Linux\_kernel

Industry average: about 15-50 errors per 1000 lines of delivered code

Code Complete, Microsoft Press Redmond, WA, USA, 2004

# Status Quo

- GFD.90 Base Classes a.k.a. Look & Feel
  - Object, Url, Session, Context, Exceptions
- Lightweight Runtime & Plug-In Mechanism
- Logging Facilities (yes: SAGA\_VERBOSE)
- SAGA Job Package (Synchronous)
- Proof-of-Concept Local fork:// Job Adaptor

# Compatibility

• Focus on *Pythonic* API (Naming, Properties, ...) but trying to maintain compatibility with the SAGA Python Bindings, e.g.

```
# separate namespace - can be renamed
import bliss.saga as saga

saga.url = saga.Url # classes can be renamed, too
saga.job.description = saga.job.Description

jd = saga.job.description()
jd.set_attribute("Executable", "/usr/bin/bfast") # old-style
jd.arguments = ['xx', 'yy'] # new-style (preferred!)

js = saga.job.Service("fork://localhost")
job = js.create_job(jd)
```

### Installation

- Bliss is in the Python Package Index (*PyPI*): <a href="http://pypi.python.org/pypi/bliss">http://pypi.python.org/pypi/bliss</a>
- Installation via easy\_install or pip:

```
$> easy_install bliss
```

 Or bootstrap in user-space on a remote machine including easy\_install & virtualenv:

```
$> curl -fsSLk https://raw.github.com/gist/1321016 > \
   pystrap.sh && /bin/sh pystrap.sh -lbliss
```

### Contribute

- Yes, Please! It's a Communal Playground...
- Give Feedback / Make Suggestions:
   <a href="https://github.com/oweidner/bliss/issues">https://github.com/oweidner/bliss/issues</a>
- Start Developing: Fork Bliss on GitHub https://github.com/oweidner/bliss

# Roadmap

- A Globus GRAM Adaptor
- A PBS via SSH Adaptor
- Tackling Thread-Safety / Metrics Interface?
- A Minimal File Package?

### Resources

- Preliminary API Documentation:
   <a href="http://oweidner.github.com/bliss/apidoc/">http://oweidner.github.com/bliss/apidoc/</a>
- Wiki & Documentation:
   <a href="https://github.com/oweidner/bliss/wiki">https://github.com/oweidner/bliss/wiki</a>
- The *User-Space Bootstrapping Script*: https://gist.github.com/1321016
- These Slides: <a href="https://github.com/oweidner/bliss/blob/docs/bliss\_intro.pdf?raw=true">https://github.com/oweidner/bliss/blob/docs/bliss\_intro.pdf?raw=true</a>