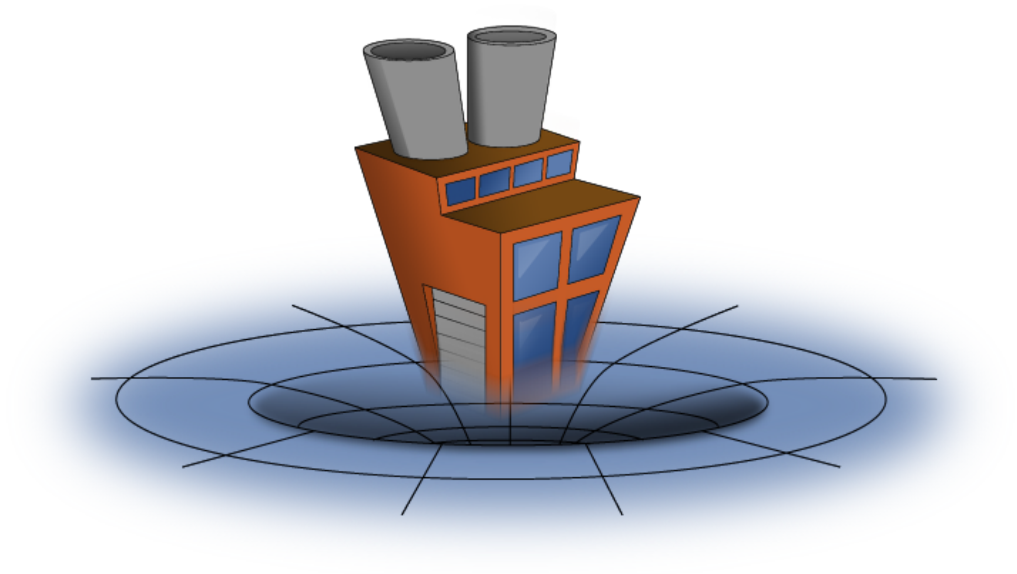


The Future of SimFactory

Ian Hinder

Collaborators: Barry Wardell and Erik Schnetter

Einstein Toolkit Workshop, Trento, June 2016

What is SimFactory?

- Tool to **manage simulations** with a uniform interface across different supercomputers
- Officially supported way to use the Einstein Toolkit
- http://simfactory.org

```
sim setup

sim build --thornlist
thornlists/mythorns.th

sim submit mysim --
parfile par/mysim.par
--procs 128 --walltime
12:00:00

[sim get mysim]

[sim archive mysim]
```

Current status

- **Works** (mostly)
- **130** open tickets (trac.einsteintoolkit.org)
- No new features in several years
- Many planned features not yet implemented

Why?

- Codebase very difficult to understand and work with
- Fixing problems takes a very long time
- Design lacks **modularity** and clarity

So?

- As a core component of the ET, it is important that SimFactory be **maintainable** and gain desired features
- **Refactor** vs **rewrite**?
- Refactoring?
  - Maintain **working system** during the process
  - Existing **bugfixes** are not lost
- Rewriting?
  - No **unit tests**;
  - Code very **overcomplicated** for the core features needed
  - Don't want to keep existing **large-scale structure**

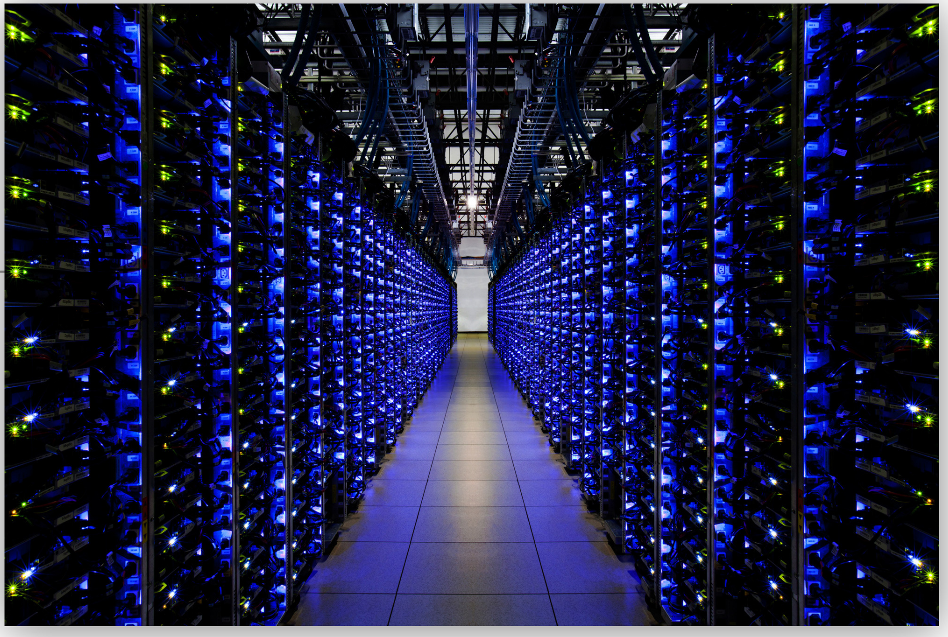
SimFactory 3

- Completely **new** implementation
- Same basic user interface
- Priorities:
  - Code cleanliness
  - Ease of use
  - Principle of **least surprise**

Status

- alpha-level at the moment
- Could be beta with a bit of work
- I use it in **production**

What works?



- **Machine** database
- **Submitting** Cactus runs to a queuing system
- Syncing/**remote** operation
- Command line interface: **same essential way** of using as SimFactory 2:

```
    sim <command> ...
```

- Multiple restarts: **output-XXXX** directory structure same as SimFactory 2; analysis codes should work fine
- Checkpoint/recovery
- Specification of **cores, nodes, processes** etc to run on

What's missing?

- Building Cactus (use SimFactory 2, or just "make")
- **What else** do you need?

New features

- Designed from the start to be "**application-agnostic**":
  - All Cactus-specific details in a single configuration file
- Configurable **termination conditions**:
  - **Automatic resubmission** if termination was due to walltime expiry; no more presubmission needed
  - Termination "**reasons**" (regexp of stdout), and "**actions**" all configurable
  - New commands: **pause/continue** (write termination file and checkpoint/recover)
- Clean code separation into Python **library** and **separate comand lineinterface**:
  - Can use **Python API** from other Python programs
  - Could create alternative interfaces (GUI, **web**, etc)

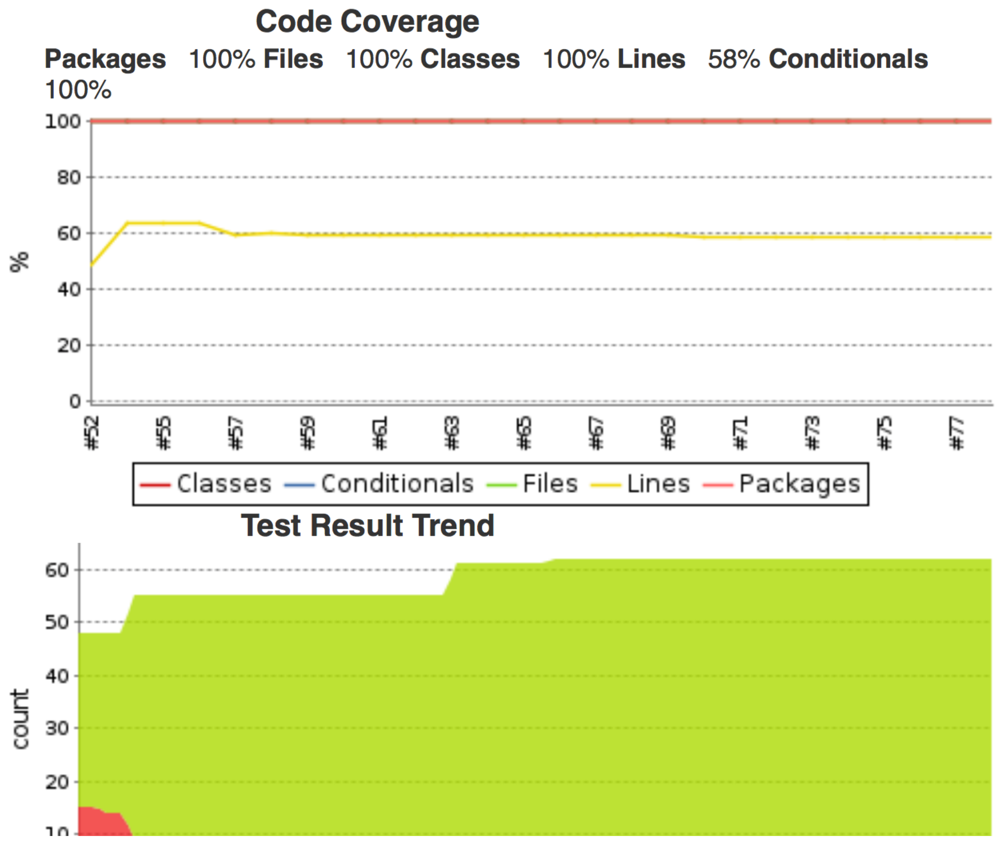
What has changed?

- Cleaned up command names and semantics
- Terminology for **nodes/processes/cores** etc rationalised
- **Single file** describing machine, incorporates:
  - machine definition
  - run script
  - submission script
  - +optionlist?

Software engineering

- Python 3 (easy to install if not available)
- **Unit tests** (currently 58% coverage) with continuous integration: cluster tests
- Object oriented; codebase modular with clear **separation** between classes

Jenkins on build.barrywardell.net



Future

- Is now **public**: https://bitbucket.org/ianhinder/simfactory3
- Probably not ready for widespread testing yet
- Could be made so with **some effort**
- Replace SimFactory 2 in the ET?
    - Some **work** still to do