

# rDBMS and script language future plans

L. Tomassetti

# Summary

- Web interface, Bookkeeping database for Monte Carlo simulations, grid job script  
⇒ **rDMBS choice**
- The new grid job script written in Python  
⇒ **Python version**

# rDBMS

- The production tools use MySQL as rDBMS for the Bookkeeping database
- Server version: 5.0.77 Source distribution [last version is 5.0.93, 5.0.x reached End of Product Lifecycle]

**End of Product Lifecycle.**

Active development for MySQL Database Server version 5.0 has ended. Oracle offers various support offerings which may be of interest. For details and more information, see the MySQL section of the Lifetime Support Policy for Oracle Technology Products (<http://www.oracle.com/us/support/lifetime-support/index.html>). Please consider upgrading to a recent version.

# rDBMS

- Production tools need some extra features:
  - Stored procedures (w/ triggers and events)
  - master/slave configuration
  - cluster?
- Meanwhile, we should move from 5.0.x

# rDBMS

- Stay with MySQL, two options:
  - 5.1.x (5.1.47 on SL6, 5.1.57 latest)
  - 5.5.x (MySQL cluster unavailable)
- Move to another product...
- The Oracle acquisition was eventually unconditionally approved by the European Commission on January 21, 2010.  
As part of the negotiations with the European Commission, Oracle committed that MySQL server will continue to use the dual-licensing strategy long used by MySQL AB with commercial and GPL versions available until at least 2015.

# rDBMS

- Alternatives:
  - Oracle (licensed, maintenance cost)
  - PostgreSQL (expertise @cnaf?)
  - ...?
- In addition, for some applications, we should investigate some NoSQL products (CouchDB, MongoDB, etc...)

# rDBMS

- Production tools (web interface) use a database abstraction class
- Submission scripts **will** use the same database abstraction class
- Job script (either BASH or Python) rely on a REST interface, which supports both MySQL and PostgreSQL

# rDBMS

- We propose to upgrade to MySQL 5.1.x in order to gain stored procedures and events (to be mainly used for pre/post-production processing)
- Try to deploy a master/slave configuration (submission/monitor) as soon as possible
- Investigate alternatives;  
it seems to be not so urgent



# Python








- The (grid) job script is being rewritten in Python (from BASH)
- More expressive, object-oriented, better configurable, ...
- Old (current) script is a (**working**) mess  
⇒ polish and rewrite

# Python


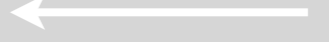
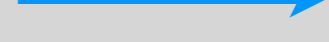

- The new script (just one) will serve both FullSim and FastSim (Grid only)
- Made of several classes (site, job, parser, logger, envvars, rest, ...)
- Requires a configuration file + runnum from the command line

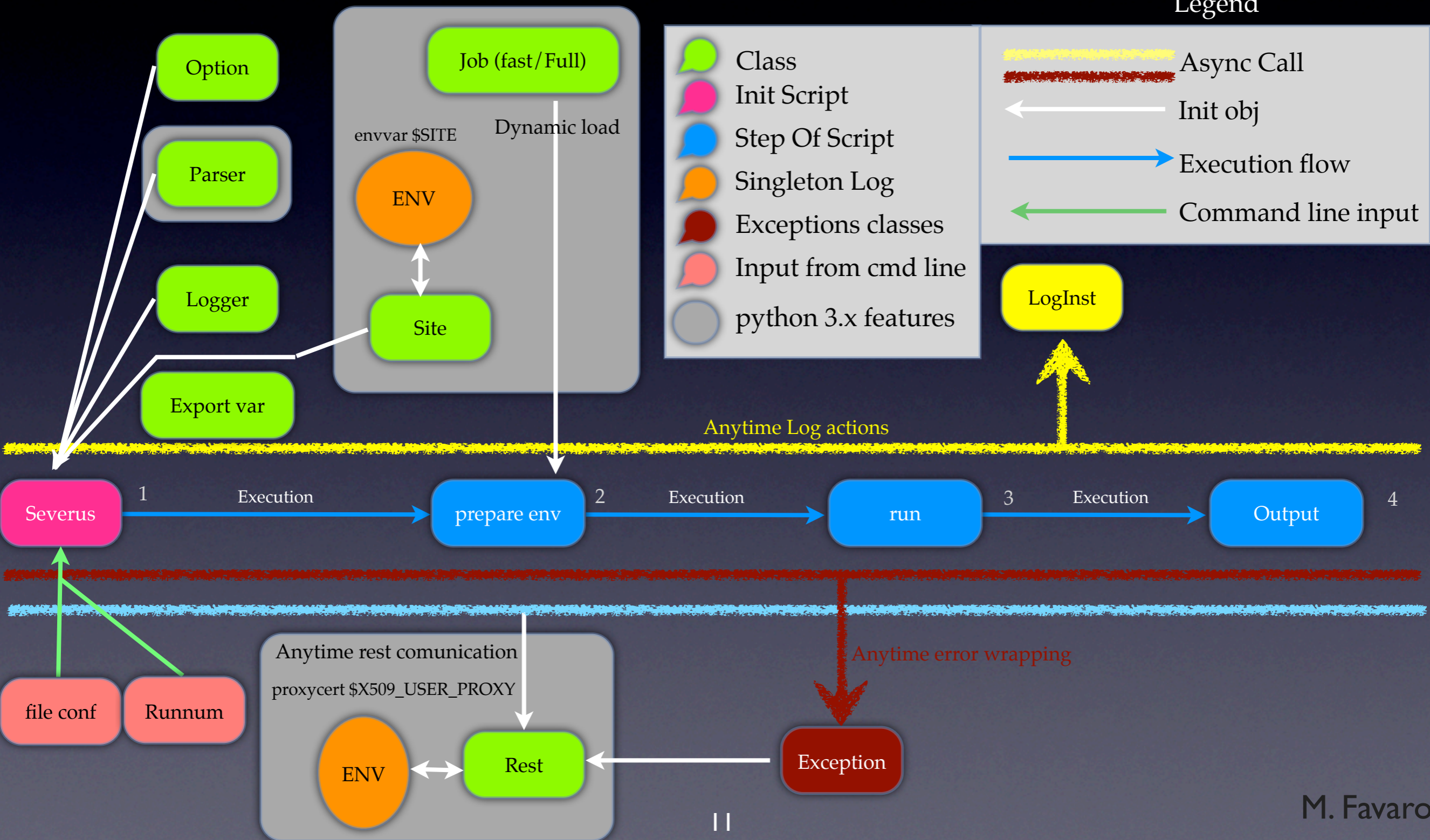
# Python

## Legend

-  Class
-  Init Script
-  Step Of Script
-  Singleton Log
-  Exceptions classes
-  Input from cmd line
-  python 3.x features

-  Async Call
-  Init obj
-  Execution flow
-  Command line input



# Python

- Available versions

- SL5: 2.4.3

- SL6: 2.6.5

- Current stable

- 2.7.1

- 3.2

- [Python 3.2](#), documentation released on 20 February 2011.
- [Python 3.1.3](#), documentation released on 27 November 2010.
- [Python 3.1.2](#), documentation released on 21 March 2010.
- [Python 3.1.1](#), documentation released on 17 August 2009.
- [Python 3.1](#), documentation released on 27 June 2009.
- [Python 3.0.1](#), documentation released on 13 February 2009.
- [Python 3.0](#), documentation released on 3 December 2008.
- [Python 2.7.1](#), documentation released on 27 November 2010.
- [Python 2.7](#), documentation released on 4 July 2010.
- [Python 2.6.6](#), documentation released on 24 August 2010.
- [Python 2.6.5](#), documentation released on 19 March 2010.
- [Python 2.6.4](#), documentation released on 25 October 2009.
- [Python 2.6.3](#), documentation released on 2 October 2009.
- [Python 2.6.2](#), documentation released on 14 April 2009.
- [Python 2.6.1](#), documentation released on 4 December 2008.
- [Python 2.6](#), documentation released on 1 October 2008.
- [Python 2.5.4](#), documentation released on 23 December 2008.
- [Python 2.5.3](#), documentation released on 19 December 2008.
- [Python 2.5.2](#), documentation released on 21 February 2008.
- [Python 2.5.1](#), documentation released on 18 April 2007.
- [Python 2.5](#), documentation released on 19 September 2006.
- [Python 2.4.4](#), documentation released on 18 October 2006.
- [Python 2.4.3](#), documentation released on 29 March 2006.
- [Python 2.4.2](#), documentation released on 28 September 2005.
- [Python 2.4.1](#), documentation released on 30 March 2005.
- [Python 2.4](#), documentation released on 30 November 2004.

# Python

- 2.4, 2.6 and 2.7 are not compatible with each other (code rewrite needed even for upgrading from SL5 to SL6)
- 2.x and 3.x are not compatible
- We need to decide the version to start with

# Python

- from python.org

## Should I use Python 2 or Python 3 for my development activity?

### What are the differences?

*Short version: Python 2.x is the status quo, Python 3.x is the shiny new thing.*

At the time of writing (July 4, 2010), the final 2.7 release is out, with a statement of extended support for this end-of-life release. The 2.x branch will see no new major releases after that. 3.x is under active and continued development, with 3.1 already available and 3.2 due for release around the turn of the year.

**3.x is the newest branch of Python and the intended future of the language.** Guido van Rossum (the original creator of the Python language) decided to clean up Python 2.x properly, with less regard for backwards compatibility than is the case for new releases in the 2.x range. This allowed several aspects of the core language (such as `print` and `exec` being statements, integers using floor division) to be adjusted to be easier for newcomers to learn and to be more consistent with the rest of the language. It also allowed later language features (such as iterators) to be applied to older language features (such as the `range` builtin which returns a list in 2.x, but a memory efficient iterable in 3.x).

The [🌐 What's New in Python 3.0](#) document provides a good overview of the major language changes and likely sources of incompatibility with existing Python 2.x code.

However, the broader Python ecosystem has amassed a significant amount of quality software over the years. The downside of breaking backwards compatibility in 3.x is that a lot of that software doesn't work on 3.x yet.

# Python

- from python.org

## So which version should I use?

Which version you ought to use is mostly dependent on what you want to get done.

If you can do exactly what you want with Python 3.x, great! There's a few downsides, such as comparatively limited library support and the fact that current Linux distributions and Macs are still shipping with 2.x by default, but as a language Python 3.x is definitely ready. As long as actually getting Python 3.x on your user's computers (which ought to be easy since a lot of people reading this may only be developing something for themselves or an environment they control) and you're writing things where lack of third party software isn't a major impediment (or where you know the packages you need already support Python 3), Python 3.x is an excellent choice. Also, quite a few distributions have Python 3.x available already for end-users, even if none of them are using it as the default interpreter at the moment.

However, there are some key issues that may require you to use Python 2 rather than Python 3.

Firstly, if you're deploying to an environment you don't control, that may impose a specific version rather than allowing you a free selection from the available versions.

Secondly, if you want to use a specific third party package or utility that doesn't yet have a released version that is compatible with Python 3, and porting that package is a non-trivial task, you may choose to use Python 2 in order to retain access to that package.

Popular modules that don't yet support Python 3 include Twisted (for networking and a bunch of other stuff), gevent (like Twisted but different), Django and Pylons (for building websites), PyGTK<sup>1</sup> and PySide (for making GUIs), py2exe (for packaging your application for Windows users), PIL (for processing images)...

# Python

- SuperB software doesn't use python yet  
(FullSim and FastSim, ?)  
Geant4Py, PyBoost, PyROOT, ... (2.x branch?)
- It might be useful to take advantage of the new features of 3.x versions
- It requires installation on experiment software area (and maintenance)



# Python v3 features

- `try{} catch{} finally{}`  
better flow control and error management, failover procedures, ...
- extended object-oriented support (`metaclassing`, interfaces and abstract classes, ...)  
better code re-use, code maintenance, modularization, ...
- file `parser`  
no need to write configuration file parsers, cleaner code, ...
- `SSL` and `certificates` improvements  
native http/https communications, SSL module standard compliant, proxy certificate compatibility, certificate chain and multiple CA check, ... (already tested for REST communications)

# Python

- LHC experiments have their own Python installation (usually 2.6.x/2.7.x)
- Python 3.2 just released (stable)
  - New releases are at most twice a year
- Test on SL5 & 3.2 shows no dependencies issues

## Python 3.2

Python 3.2 was released on February 20th, 2011.

Python 3.2 is a continuation of the efforts to improve and stabilize the Python 3.x line. Since the final release of Python 2.7, the 2.x line will only receive bugfixes, and new features are developed for 3.x only.

Since [PEP 3003](#), the Moratorium on Language Changes, is in effect, there are no changes in Python's syntax and only few changes to built-in types in Python 3.2. Development efforts concentrated on the standard library and support for porting code to Python 3. Highlights are:

# Conclusions

- rDBMS: upgrade to MySQL 5.1.57 (just one machine @cnaf) and plan to upgrade to v5.5.x and investigate others rDBMS
- Python: try to install version 3.2 in the experiment software area (requires discussion, agreement and efforts of many people)

# Discussion

- We can extend the discussion into the Computing Planning session on Wednesday morning