Scripting

Application Area Internal Review

Shared Environment for Applications at LHC

Aims and Motivation

- Architecture Blueprint RTAG proposes the inclusion of Python in the architecture
  - scripting environment
  - configuration tool
  - component bus
- Python's presence should be non-intrusive

Binding Technologies

- Motivation and Aim
  - RTAG Mandate
  - Investigate ways in which Python bindings could be created
  - Make recommendations of best practice.
- What
  - Boost.Python and SWIG are the clear favourites
  - No convincing technical argument for choosing one over the other
  - Investigate interoperability issues
    - Looks like significant work necessary
  - PyLCGDict provides an alternative approach
    - Use for quick bindings
    - Resort to Boost or SWIG for "product" wrappings

Binding Technologies

- What next
  - Pick either Boost or SWIG on non-technical grounds ?
  - Contract Boost Consulting to make Boost and SWIG interoperate ?
    - Estimate: 65-105 hours work for the world's expert in the field
  - Attempt interoperability work ourselves ?
  - Delay decision until confronted with real problems ?
    - Attempt ad-hoc solutions ?
    - PyLCGDict as bridge technology ?
    - Attempt interoperability after all ?

PyLCGDict

- Aim
  - Provide access to C++ libraries from Python, “for free”
- What
  - Automatically generates Python proxies for C++ objects
  - Vector-like C++ objects support the Python sequence protocol
  - Namespaces and Templates look natural in Python

  Pro: Anyone can create bindings effortlessly

  Con: User has no control over the C++ -> Python interface mapping
- How
  - Proxies based on information in LCG Dictionary
    - C++ meta-(meta-)objects wrapped with Boost
    - Python objects manipulated from C++ via Boost

PyLCGDict

- When
  - Appeared in SEAL 1.0.0; July 18
- What next
  - Migrate much of functional core from C++ to Python
  - Exploit Python's metaclasses, to make a more natural, simpler representation
  - Support more natural Python features (eg iterator protocol)

PyROOT

- Aim
  - Provide access to ROOT functionality from Python
- What
  - All ROOT basics
    - histograms
    - ntuples
    - files
    - graphics
- When
  - Appeared in SEAL 0.2.0; April 7
  - Reincarnation and recasting of ideas previously available in Gaudi repository

PyROOT

- How
  - Boost.Python
  - Avoids binding individual ROOT classes
  - Uses CINT dictionary for reflection
- What next
  - Undergoing performance improvements
  - Unify core with PyLCGDict ?
  - Gateway Root ➟ Python
    - Python from ROOT
      - issue Python command in CINT for execution by Python interpreter

Python Courses

- Aim
  - Provide assistance in the use of Python
- What, How
  - 3 day course: Hands-on Introduction to Python Programming
  - Available through CERN Technical Training programme
  - Receives excellent reviews via course feedback questionnaires
- When
  - Originally run for ATLAS' software week in May
  - Now, every month or two, depending on demand

Python Courses

- What next
  - Half-day or day-long sessions on specific areas
    - PyROOT
    - Creating bindings
    - More advanced Python programming
    - GUI building in Python ?
  - Feedback questionnaires often request further general or specific courses

What next

- More bindings
  - GSL ?
    - A non-CERN PyGSL project exists
      - Partial binding only
      - SWIG based
      - Hand-written; slow progress
    - Are we interested in contributing ?
  - Working with POOL on FileCatalog
  - Need input from potential users about what binding products they need.
- PyBus
  - Exploratory effort to understand how to decorate Python's module management system