

**Valeriy Onuchin**

# **ROOT Apache Module**



Valeriy Onuchin, IHEP, Protvino

# *What is it?*

**Allows to execute C++ macros on server side and send result to browser.**

**Allows to embed C++ code directly into HTML body ( as similar as SSI )**

**... and HTML code into C++ ... allows redirect `stdout` to web-browser.**

**Full power of the ROOT framework is available, e.g.**

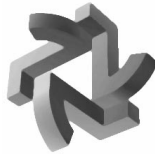
- executing macros, producing pictures on-the-fly and sending them to web-browser.
- extending ROOT running on server with external libraries
- executing compiled macros for native performance and code hiding
- much more ...

**Includes functionality of `mod_root.c` module which was used for reading content of ROOT files over net.**

**Allows to browse content of server-side ROOT files over net.**

**Provides inter-server communication methods.**

**Along with RDBC can be used for building database-driven Web sites.**



# Examples

---



# How does it work?

## ROOT-Apache consists of two shared libs:

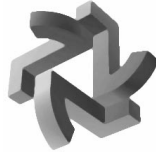
**libroot.so** – apache module itself. This module handles client (web-browser) requests

**libCarrot.so** – currently contains **TApache**, **TApacheBrowser** classes. These classes are available to user. They allow to manipulate with data and communicate with client or other server.

When http daemon starts it loads **libroot.so**. It itself loads **libCarrot.so** and instantiates **ROOT** session and loads **ROOT** libs. So, **httpd** daemon "contains" running **ROOT** session.

When client sends request to server for a file which has extension either **.C** or **.html** or **.root**, such request is handled by **ROOT-Apache** module.

- In case of **.C** extension, **ROOT** macro is loaded/executed and result sent back to client.
- In case of **.html** extension, C++ code is extracted and executed. The body of C++ code is substituted with output result of its execution. The final **HTML** code is sent to client.
- In case of **.root** extension either **TApacheBrowser** started or communication mechanism used for reading of content of **ROOT** file is involved (**TWebFile** reading )



# Use Cases

- **Web interface to experimental data**
- **Dynamic database-driven webpages**
  - ROOT+Apache as an option to ASP,PHP,ColdFusion,Lasso,mod\_perl ...
- **Web-based GUI for online software**
  - ROOT+Apache as an option to Java
- **TApache::POST/GET methods allow to communicate between ROOT–Apache driven web-servers – send "programs", "objects", data, get response etc.**
  - batches/stored procedures in MySQL+ROOT+Apache
  - distributed parallel processing
- ... ???



# Requests

## Grand Challenge Request:

Make it possible to run few ROOT sessions on the same computer for the period of few months and executing thousands of macros per day without crashing.

**Garbage Collection and Leakage Control in ROOT/CINT.**

**Customised error logging for CINT.**

**Possibility to execute C++ macro from memory buffer.**

**Possibility to rewind/undo ROOT actions performed during the session**

- gInterpreter->ResetAll()
- gInterpreter->Rewind(to\_some\_point)
- gSystem->Unload(lib)

**I need modification of some ROOT GUI classes, e.g. TBrowser, TGxxx classes, in order to have possibility to derive web GUI classes from them.**



# TODO

---

- **Testing, docs, examples**
- **Add missing features to the module:**
  - sharing data between HTTP childs, persistent db.connections
  - authentication & authorization
  - customised error logging
  - customised configuration of the module
  - possibility to define Init/Exit scripts
    - global init, global exit, child init, child exit scripts
- **Web-based GUI classes and webpage templates**
  - pulldown, popup, tabbed menus, list trees, etc.
- ++???

