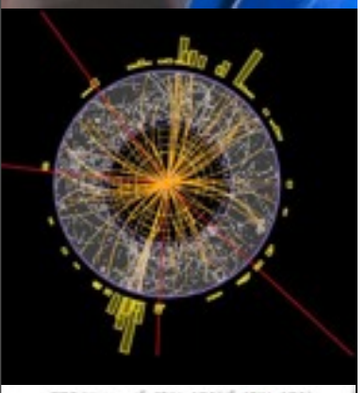
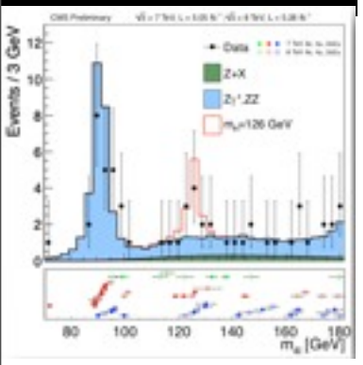
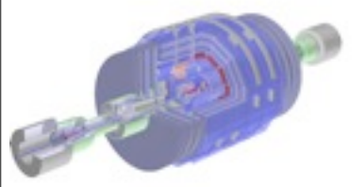






ROOT Plan of Work 2013 Review

Delivering ROOT 6

Fons Rademakers

CERN, 14 May 2013

Outline

- The challenge
- The development process
- The team
- The plan of work
- The roadmap
- Conclusions

The ROOT 6 Challenge

- Changing interpreters
  - Ripping out CINT and replacing it by the all new Cling
  - Basically like open heart surgery

Typical ROOT Development Process

- Release early, release often
- Get early adopters to try out new features
- User feedback drives largely development schedule

Typical Release Schedules

- Normally release schedules are time driven
  - Agreed with experiments, usually May and November
- Typically (small) incremental improvements
- Allows for a decision on what goes in and what not a few months ahead of the release date
- Typically release is closed 2 weeks before Release Candidate (RC) release date
- RC1 - 4 weeks before release date
- RC2 - 2 weeks before release date
- Then release

ROOT 6 Release Schedule 2012

- **Here different kind of schedule**
  - Swapping out the heart of the application, the interpreter, cannot be done stepwise, it has to be done in one go
- Feature driven schedule
- We first created and released standalone Cling, our new Clang/ LLVM based interpreter, summer 2011
- Based on that success, we made a first estimate of being able to swap Cling in for CINT and have a release by the end of the 2012

Updated ROOT 6 Release Schedule

- However, some essential part of the system, the module merging feature, needed for the dictionaries, proved to be more complicated and untested than foreseen
- **Making estimating ETA very difficult**
  - Lacking in-house compiler technology expertise
  - Learning on the job

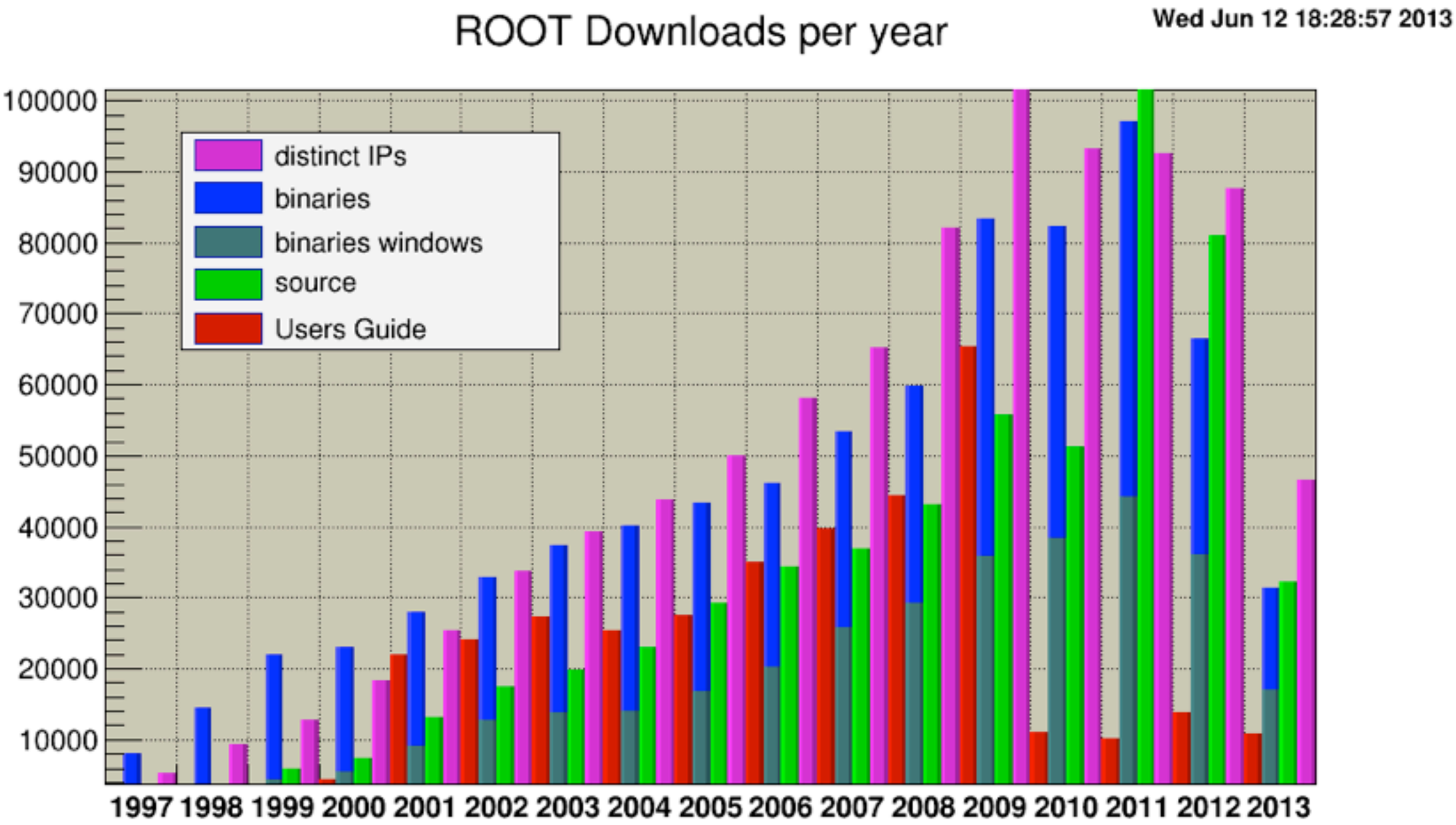
The 2013 Core ROOT Team

ROOT FTE's

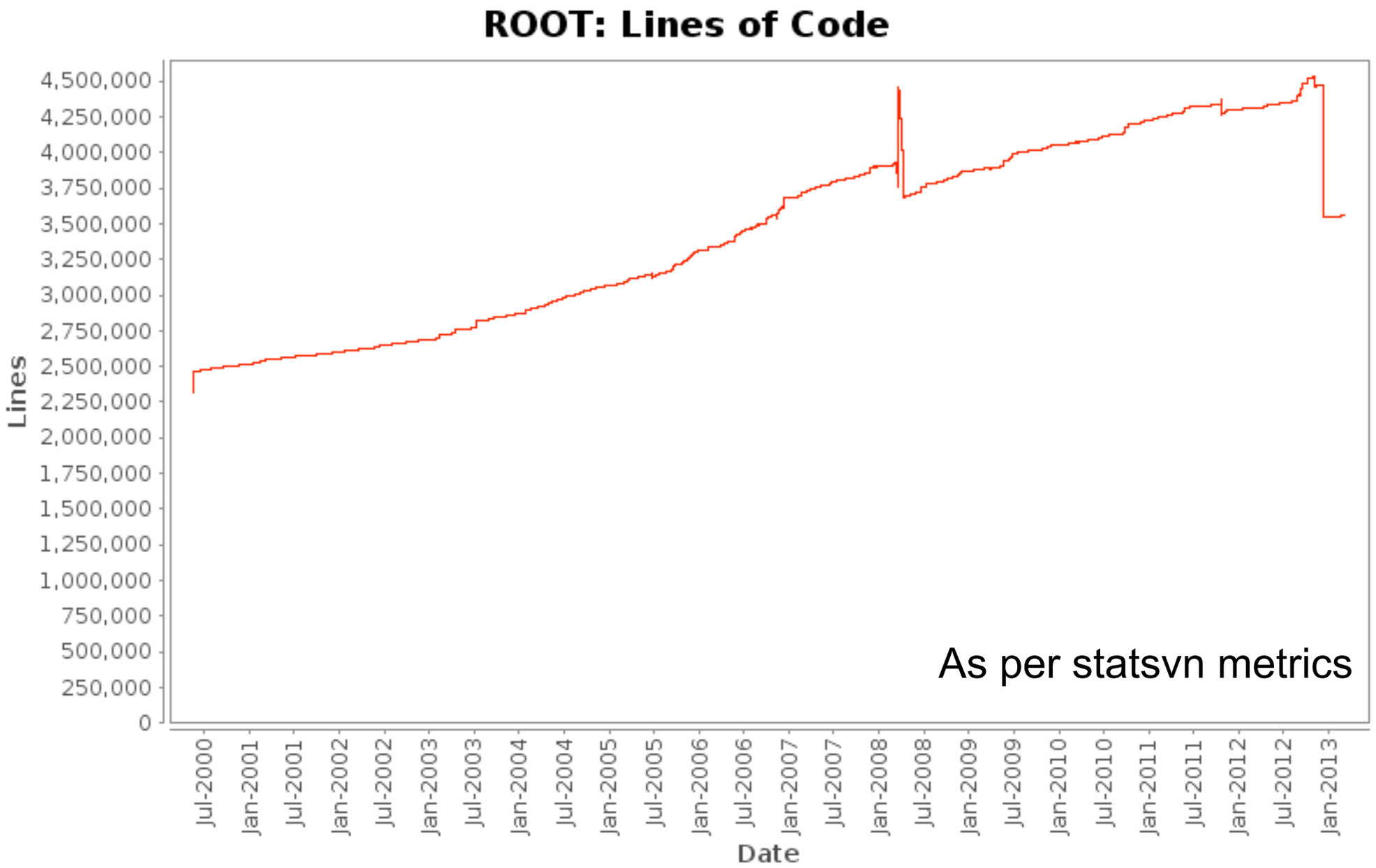
- Axel Naumann (100%, CERN)
  - Interpreters, dictionaries, infrastructure
- Olivier Couet (100%, CERN)
  - 2D/3D graphics, documentation
- Timur Pocheptsov (50%, CERN, PJAS 5/14)
  - 3D graphics, iOS
- Gerri Ganis (60%, CERN)
  - PROOF
- Bertrand Bellenot (75%, CERN)
  - GUI, Windows, ROOT in Javascript, Cling
- Lorenzo Moneta (80%, CERN)
  - Math, statistics
- Fons Rademakers (100%, CERN)
  - Project management, core classes, infrastructure
- Vassil Vasilev (100%, CERN, Fellow 2/14)
  - Cling
- Philippe Canal (50%, FNAL)
  - I/O, Trees, Interpreters
- Dario Berzano (100%, CERN, DS 4/14)
  - PROOF
- Wim Lavrijsen (50%, LBL)
  - PyROOT
- Andrei Gheata (10%, CERN)
  - Geometry package
- Paul Russo (50%, FNAL)
  - Cling
- Cristina Cristescu (100%, CERN TS, per 7/13)
  - Cling
- Matevz Tadel (10%, UCSD)
  - 3D graphics, event visualization
- Wouter Verkerke (30%, NIKHEF)
  - RooFit
- Helge Voss (50%, Heidelberg, 12/13)
  - TMVA
- Anar Manafov (50%, GSI)
  - PROOF, PoD

ROOT - In Numbers of Users

- Increasing number of users
  - 6800 forum members, 68750 posts, 1300 on mailing list
  - Used by basically all HEP experiments and beyond

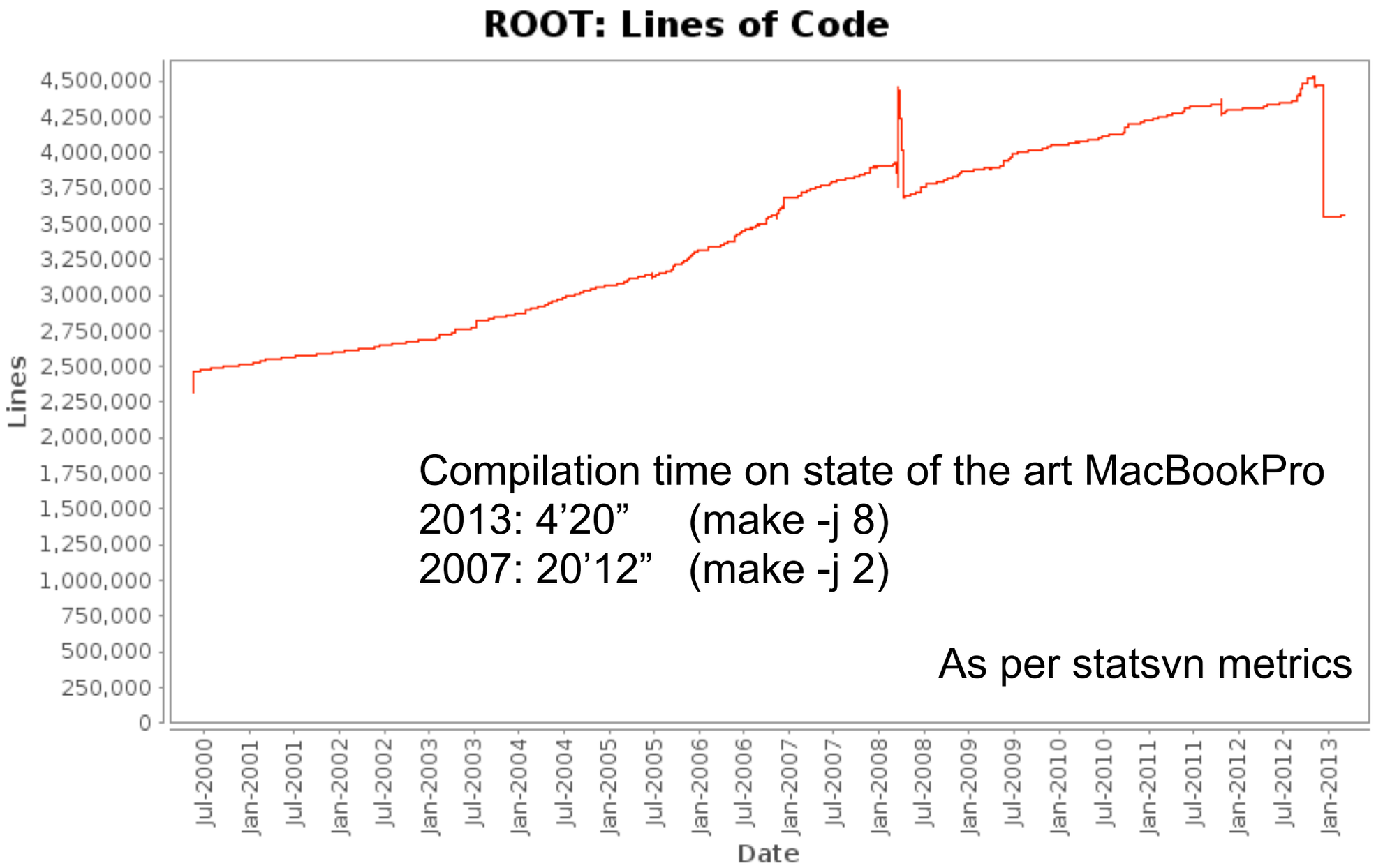


ROOT - In Lines of Code



Real MLOCs: 2.75, was 3 with CINT

ROOT - In Lines of Code



Real MLOCs: 2.75, was 3 with CINT

ROOT - In Numbers of Bytes Stored

As of today

177 PB

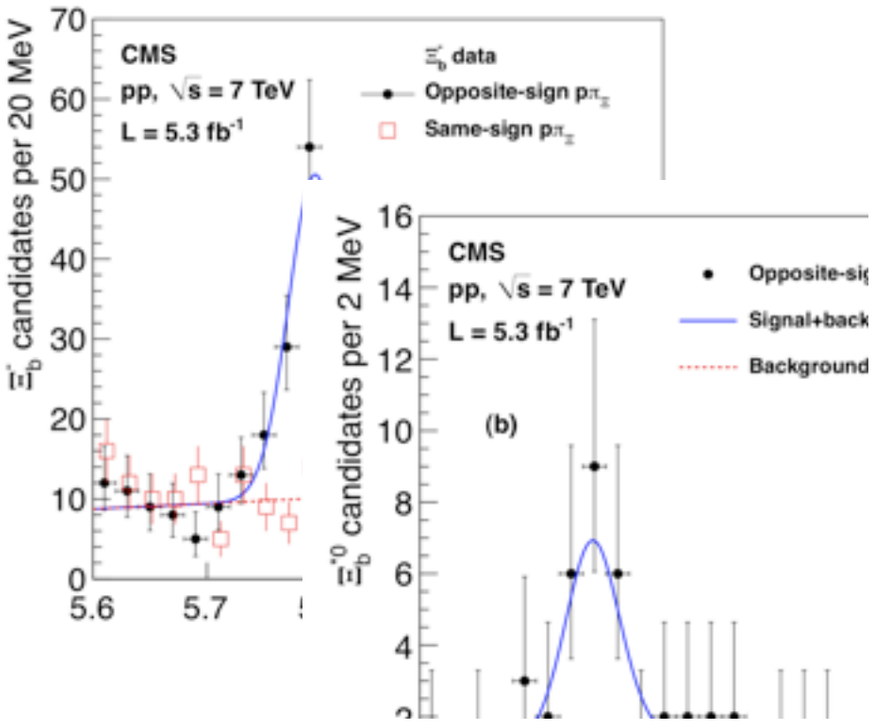
of LHC data

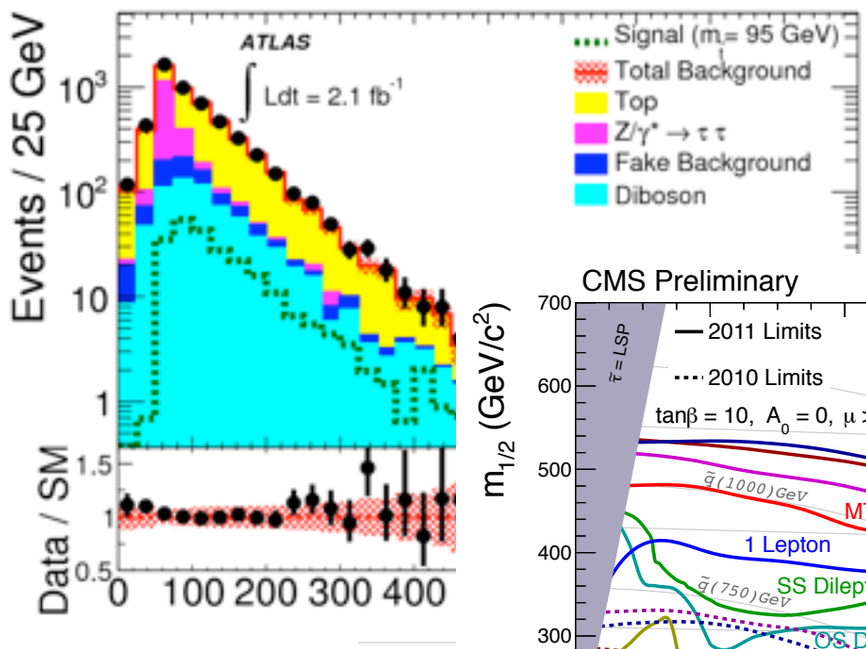
stored in ROOT format

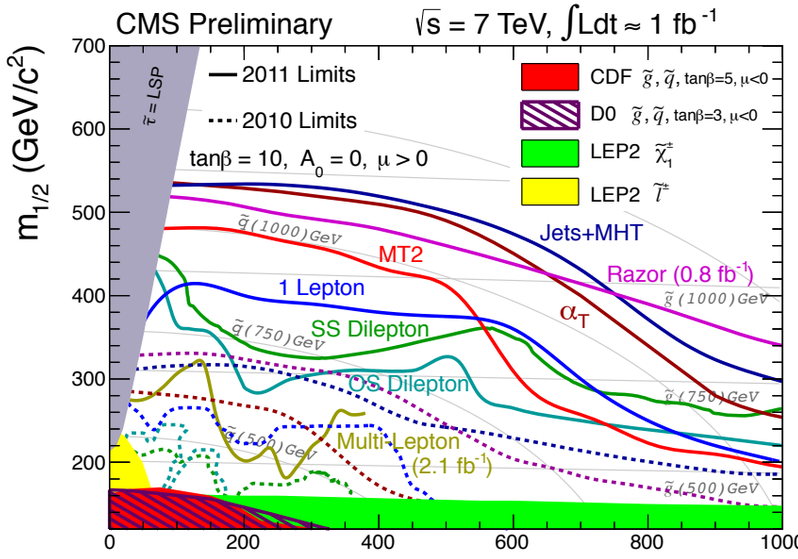
ALICE: 30PB, ATLAS: 55PB, CMS: 85PB, LHCb: 7PB

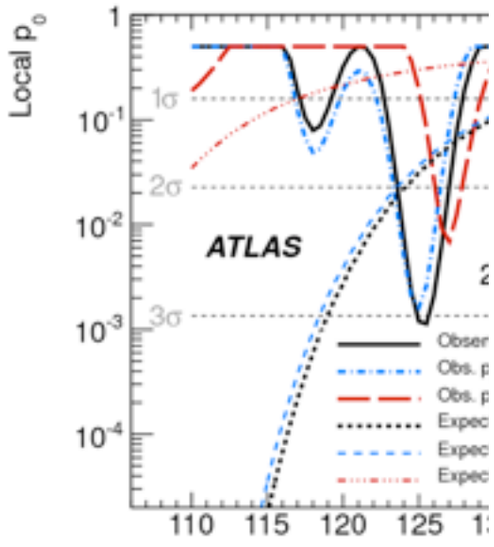
ROOT - In Plots

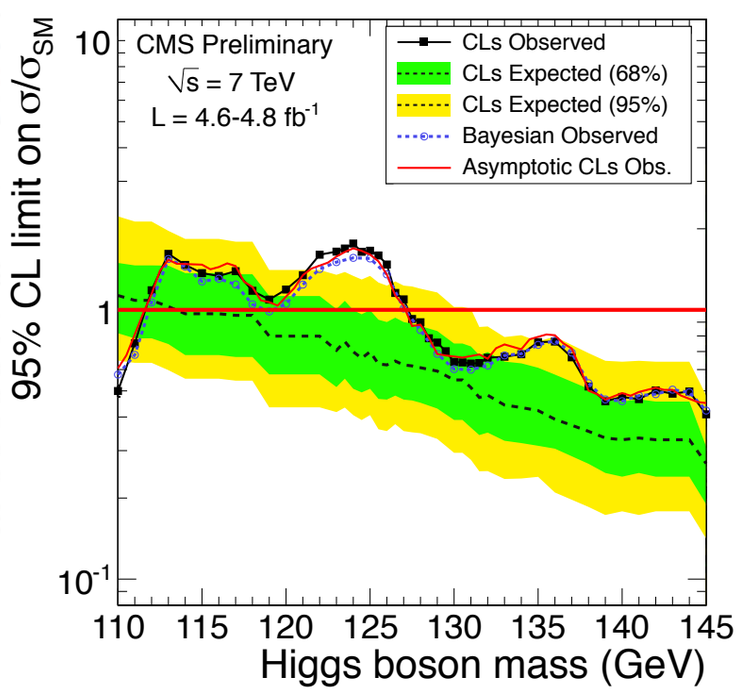
ROOT - In Plots

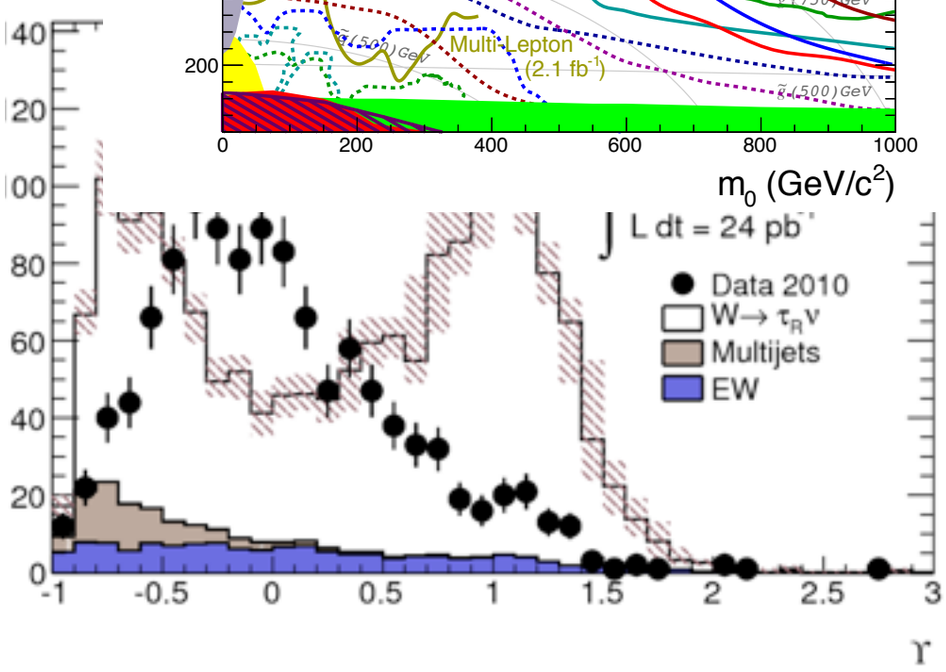












ROOT 2012 Highlights

- Standalone Cling released in summer
- Great progress with ROOT I/O and graphics in Javascript
- Native OSX version released
- ROOT graphics ported to iOS
- Introduction of advanced LaTeX engine via TMathText
- Good progress in I/O due to the ROOT I/O workshops
- Async tree prefetching improved and multiple tree caches
- PROOF widely deployed on Tier-3's
- Alternative cmake based build system
- Ports to more platforms: x32, ARM (iOS, linux)
- Growing user base (active forum community)

The ROOT 2013 Users Workshop

The Workshop Outcome

- ROOT's features overwhelm its documentation
  - AKA, we want better and up-to-date documentation
  - Starter guide, primer, book
  - Topical guides, more HowTo's
  - More training (see ATLAS survey)
- Improve end user graphics experience
  - Improvement in default text placement
  - Intelligent legend box placement
  - Text alignment
  - Better anti-aliased graphics
- Better multi-threaded support
  - Use threads to speed up common operations - compression of buffers, read-ahead, ...
- Improved parallel file merging
- TTreeReader prototype was very well received

User Support and QA

- We can and have to do better
- With Rene's departure we lost 2FTE in user support
- The forum has too many 0 response messages
  - Clearly there is no critical mass yet in self help
  - Will re-start dedicating more time on forum
- Bug fixing remains top priority, blockers and critical bugs are investigated and handled immediately
  - Move to JIRA has been very positive; improved dashboard and issue presentation helps
- Number of QA tests in roottest increasing
- Some stress tests may need re-calibration

Cling Integration

- Cling released in July 2011
- Cling based on LLVM and clang compiler libraries
- Fully functional C and C++11 interpreter
- Just-in-Time compilation, i.e. executes always machine code
- Support multiple interpreter objects (c.f. multi-threaded context)
- Interfaces via TInterpreter (TCling) and TClass
- Precompiled headers as dictionary, no huge source files

- All details in Axel's presentation

I/O Developments

- Continuous review of I/O performance in close collaboration with experiment experts
- Reimplementation of OptimizeBaskets
- Make use of new major ROOT revision to introduce beneficial file format changes
- Implementation of the parallel merger
- Improvements in thread safety, TFile per thread

- All details in Philippe's presentation

Math and Stats Developments

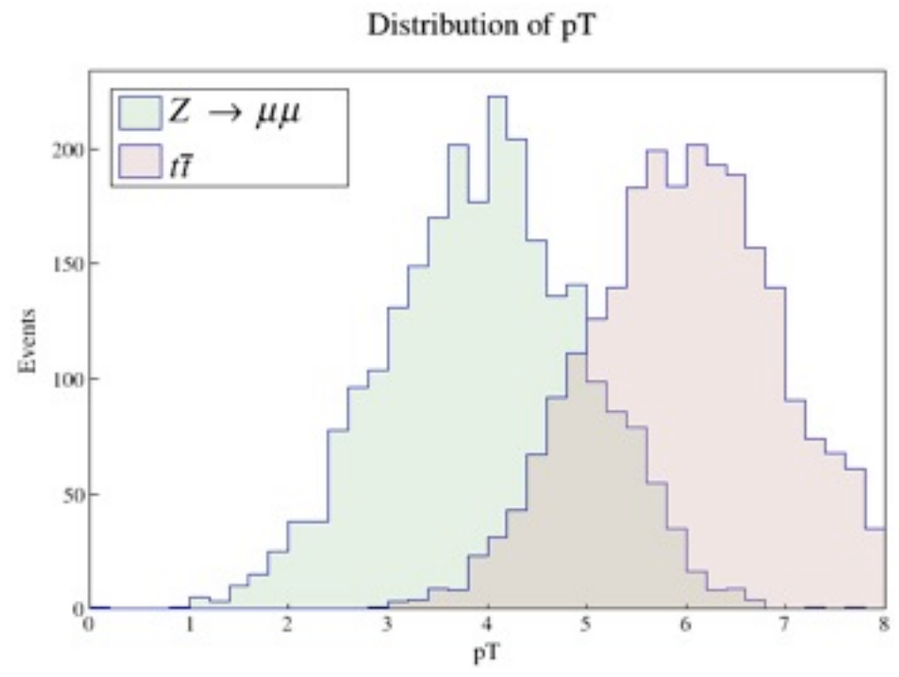
- Plan to merge Vc using physics vectors and matrices (SMatrix)
- Vectorization of mathematical functions and fitting
- Make use of cling/LLVM in Math libraries (e.g. new TFormula, automatic differentiation, etc...)
- Consolidation of RooFit, RooStats and TMVA

- All details in Lorenzo's presentation

Core Libraries

- Work on thread safety issues
  - Most globals use thread local storage (gFile, gPad, etc)
  - Protect or remove statics
  - Remove big container lock
  - [Cling will allow a finer grain interpreter lock]
- Investigate lock free containers
- TThread replaced by C++11 threads or thin layer for backward compatibility
- Focus on C++11 thread, atomicity, tld, concurrency support

Graphics



- **Use transparency more widely in ROOT**. In particular it should be added in the color editors.

- **Transparency with X11**: Transparency is not possible on X11. In order to get it we are now investigating the possibility to render graphics in Pad using TASImage.

- **"Perceptual" colormaps** explicitly identifying the "0" (or any value). This will require a new way of managing color maps because a value will trigger the change of color map.

- **Filled polygons** with color gradients.

- **Improve the GraphViz** interface (interactivity) and use these classes in packages like THTML.

Graphics

- **TMathText PS output to be completed**. Now PS files containing TMathText are too big because the complete font files are loaded in the PS file. The next step will be to load only the needed characters.

- **TMathText** should be implemented for **PDF** output.

- **TMathText** can also be used to extend the list of **marker types**.

$$\prod_{j\geq 0}\left(\sum_{k\geq 0} a_{jk}z^k\right) = \sum_{n\geq 0} z^n \left(\sum_{\substack{k_0,k_1,\ldots\geq 0 \\ k_0+k_1+\cdots=n}} a_{0k_0}a_{1k_1}\cdots\right)$$

$$W^{3\beta}_{\delta_1\rho_1\sigma_2} = U^{3\beta}_{\delta_1\rho_1\sigma_2} + \frac{1}{8\pi^2}\int_{a_1}^{a_2} d\alpha_2' \left[\frac{U^{2\beta}_{\delta_1\rho_1} - \alpha_2' U^{1\beta}_{\rho_1\sigma_2}}{U^{0\beta}_{\rho_1\sigma_2}}\right]$$

$$d\Gamma = \frac{1}{2m_A}\left(\prod_f \frac{d^3p_f}{(2\pi)^3}\frac{1}{2E_f}\right)|\mathscr{M}(m_A - \{p_f\})|^2(2\pi)^4\delta^{(4)}(p_A - \sum p_f)$$

$$4\mathrm{Re}\left\{\frac{2}{1-\Delta\alpha}\chi(s)\left[\hat{g}_v^e\hat{g}_v^f(1+\cos^2\theta) + \hat{g}_a^e\hat{g}_a^f\cos\theta\right]\right\}$$

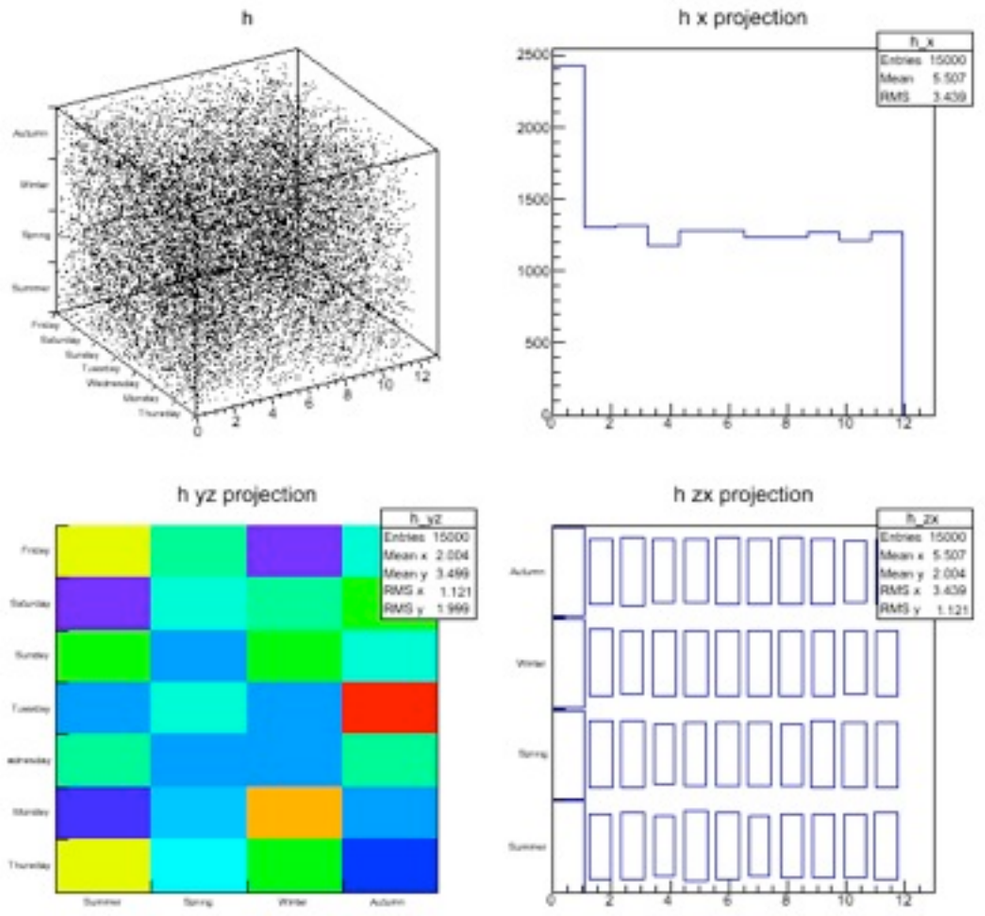
$$p(n) = \frac{1}{\pi\sqrt{2}}\sum_{k=1}^{\infty}\sqrt{k}A_k(n)\frac{d}{dn}\frac{\sinh\left\{\frac{\pi}{k}\sqrt{\frac{2}{3}}\sqrt{n-\frac{1}{24}}\right\}}{\sqrt{n-\frac{1}{24}}}$$

$$\frac{(\ell+1)C_\ell^{TE}}{2\pi} \quad \mathbb{N}\subset\mathbb{R}$$

RHIC スピン物理 Нью-Йорк

Graphics

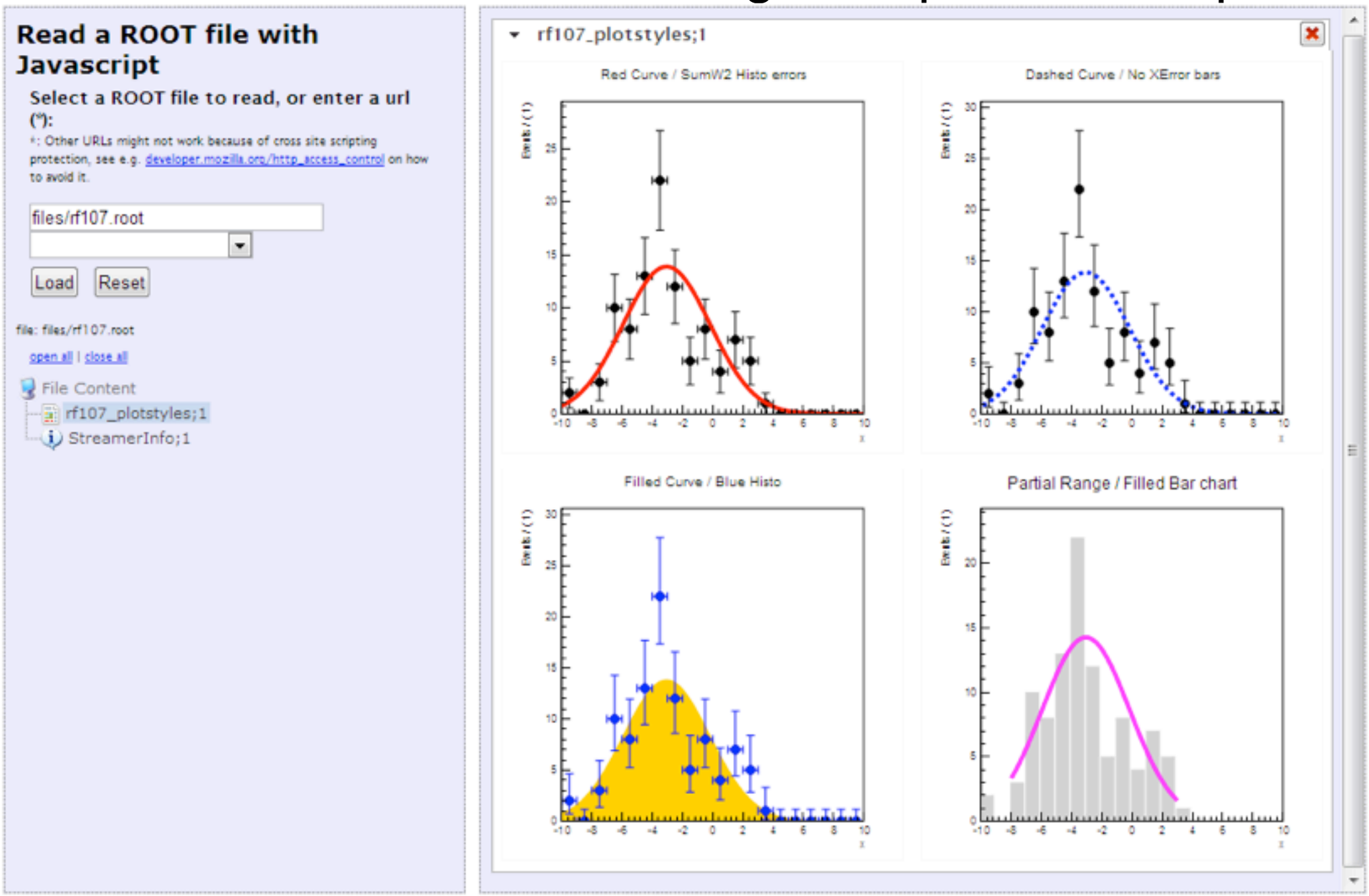
- One of the outcomes of the ROOT User's Workshop was that the **quality of the default graphics styles has to be improved**. The basic idea is that, without changing any settings a ROOT user should get by default publication quality plots. We plan will make a survey of several graphics packages to understand their default styles using different benchmark plots. Then a user survey will be held to determine the most attractive styles before implementing the chosen style in ROOT.



- **Improve the interactivity guidance of the placement of ROOT graphics objects** on the canvas. For instance the interactive move of a piece of text on the graphics window should be guided by some rulers allowing a good alignment against other objects presents on the canvas.

ROOT I/O and Graphics in JavaScript

- Provide ROOT file access entirely locally in a browser
  - ROOT files are self describing, the “proof of the pudding...”



ROOT I/O and Graphics in JavaScript

- Stable and in maintenance mode
- Only a few missing graphics and options (e.g. TProfile2D, TF2, ...)

- Next steps (low priority)
  - Implement missing features (e.g. Latex)
  - Help integration with experiments
  - Following-up Elsevier integration

- GUI is in maintenance mode

PROOF Developments

- Establish PoD for resource management
  - Addresses stability, setup and admin issues.
  - Adapt to ‘static’ and new ‘virtual’ clusters
  - See: http://proof.web.cern.ch/proof
- Integration with CernVM-FS
  - Successful for ALICE, ATLAS, CMS
- Address specific use-cases (merging, running of existing macros, roostat/roofit, ...)

PROOF Developments

- Reduce output memory footprint (PROOF-30)
- Improved data packetizer (PROOF-24)
- Dynamic workers setup (PROOF-26)
- Integration of TParallelMergingFile (PROOF-32)
- Package manager (ROOT-97)

Geometry Developments

- Adding vector-based API for geometry navigation

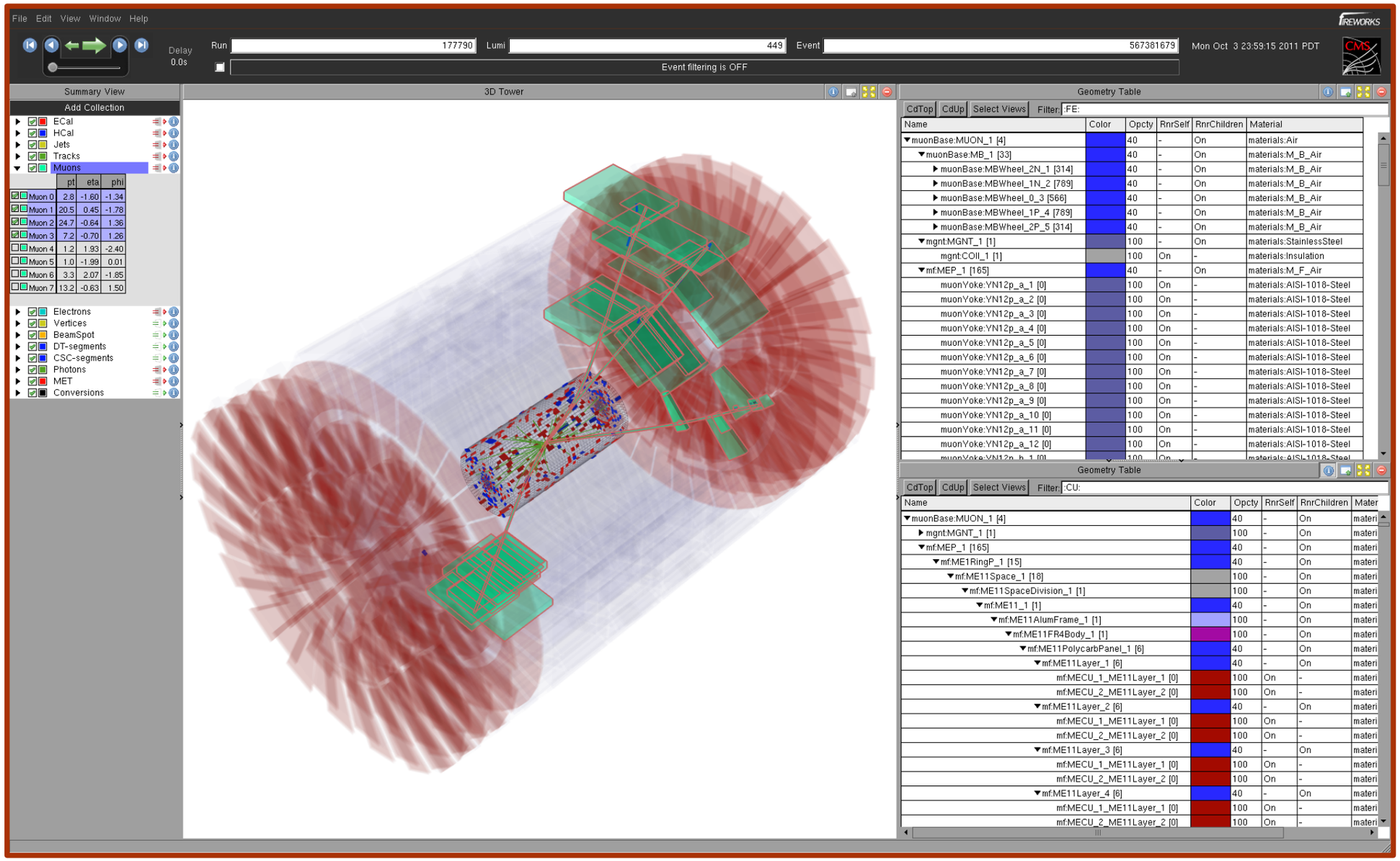
```
Bool_t      TGeoShape::Contains(Double_t *point)
Bool_vect_t TGeoShape::Contains(Point3D_vect_t points)
```

  - First implementation as loop factorizing out common computation
  - Longer term effort to achieve auto-vectorization widely
- Vector and GPU-friendly navigator
  - Allowing to work with vectorized transport on CPU AND co-processors
  - Only bootstrap version by November
- Bridge class to interface to USolids library

```
class TGeoUShape : public TGeoBBox
```

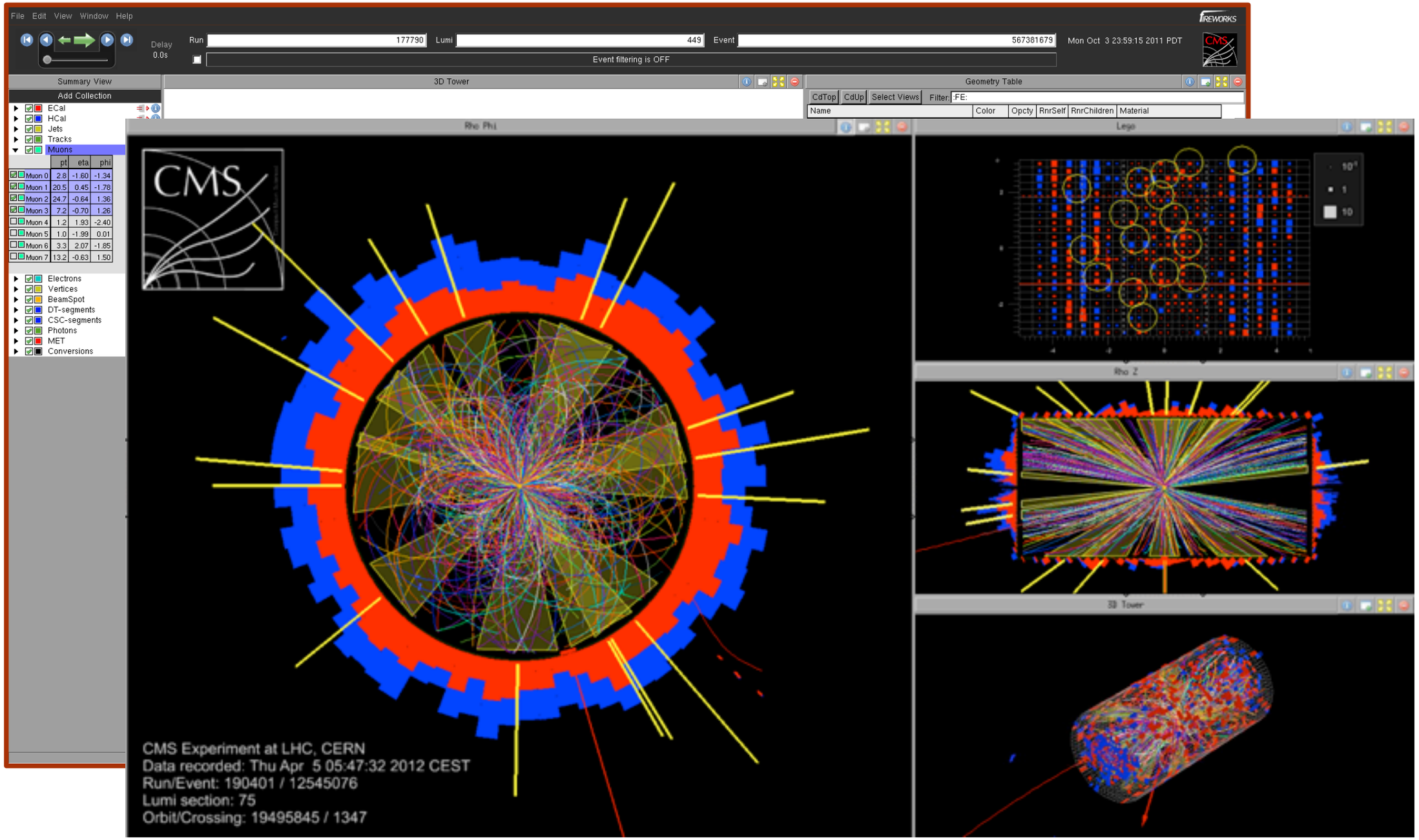
  - Allowing to use all new features and enhancements from Usolids
- Stable usage of geometry navigation in parallel operation
  - Minimal overhead compared to the single-threaded version

Eve Event Display



- Eve is in maintenance mode

Eve Event Display



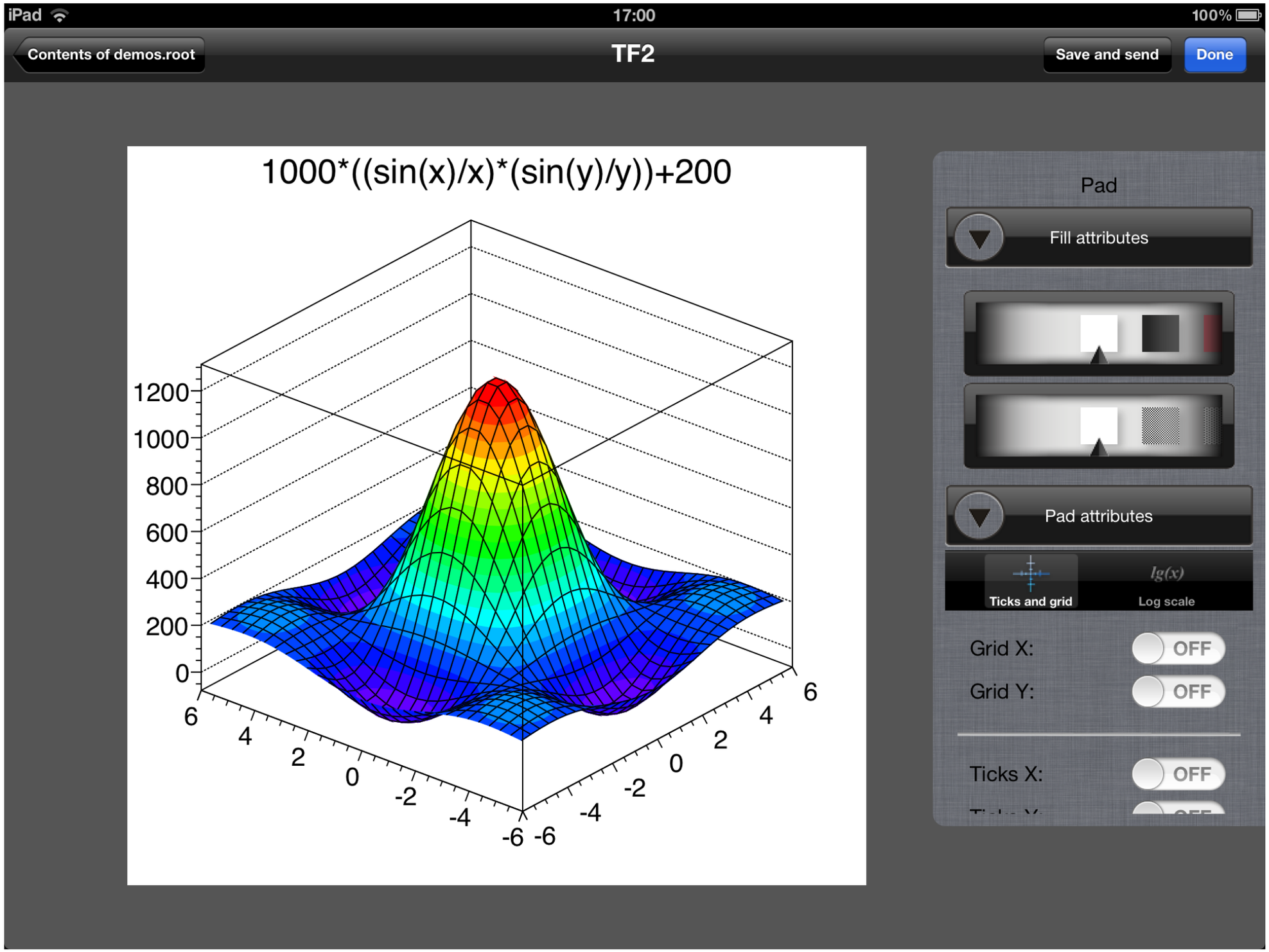
- Eve is in maintenance mode

iOS Development

- ROOT Browser App:
  - Extend picking algorithms (TObject::DistanceToPrimitive)
    - To make all primitives in canvas/pad selectable by touch gestures
  - Deploy in CERN App store

- Eve for iOS:
  - OpenGL ES version of our gl/eve modules (core render)
  - UI for iPad/iPhone
  - Integration with CERN app (“LIVE” part of application)

ROOT Browser App on the iPad



Infrastructure

- Build using “./configure;make” and now also “cmake;make”
- Code now managed in Git
- Moved to JIRA for issue management
- Static code analyzer
  - Coverity (commercial)
- New continuous build system
  - Electric Commander (commercial)
  - Incremental and full builds and nightly build snapshots for 14 platforms/configurations
- New source code convention checker
  - Eclair (commercial, but uses LLVM inside)
- New documentation system
  - From MS Word to Docbook to Markdown

From Docbook to Markdown

```
<sect1>
<title>Main title</title>
<para>
To make a paragraph just skip one line.
</para>
<sect2>
<title>A sub-section</title>
<para>
Now an itemized list:
</para>
<itemizedlist>
 <listitem><para>item 1</para></listitem>
 <listitem><para>item 2</para></listitem>
</itemizedlist>
<para>
The basic idea behind this mark-up language is
to be as close possible to what you would write
in a email.
</para>
</sect2>
</sect1>
```

```
# Main title

To make a paragraph just skip one line.

## A sub-section

Now an itemized list:

 - item 1
 - item 2

The basic idea behind this mark-up
language is to be as close possible to
what you would write in a email.
```

ROOT and C++11

- ROOT does not contain C++11 code, but is C++11 compatible, i.e. it can be compiled by C++11 compilers
  - Does not prevent experiments from moving to C++11
- ROOT 6 can parse C++11 headers to generate dictionaries (#include at prompt)
- Only when all experiments have moved to C++11 can ROOT use C++11 code
  - Only then can overhaul interfaces and have C++11 in API's
  - Only then can use C++11 stdlib (e.g. thread and atomic)

Status of Tutorials

tested	failed	failed (%)
351	167	48
		Failed (%)
load libGpad.so	"top-level" tutorials, total : 10 failed : 7	70
hsimple.C	Pass	
benchmarks.C	crash, benchmarks_trace.txt	
demos.C	compilation error, demos_log.txt	
demoshelp.C	Pass	
regexp.C	Pass	
regexp_pme.C	compilation error, regexp_pme_log.txt	
htmlex.C	crash, hmlex_trace.txt	
rootenv.C	compilation error, rootenv_log.txt	
tasks.C	compilation error, tasks_log.txt	
rootmarks.C	error - requires benchmarks.C to work	
		Failed (%)
load libGpad.so	cocoa : total 5 failed : 2	40
grad.C	Pass	
graf2.C	Pass	
transp.C	Pass	
parallelcoordtrans.C	compilation errors, crash, parallelcoordtrance_trace.txt	
transp_text.C	compilation errors, crash, transp_text_trace.txt	
		Failed (%)
	cont, total : 1 failed : 0	0
TListAndSTL.C	Pass	
	fft, total : NA failed : NA	
		Failed (%)
load libGpad.so	fit, total: 35, failed : 7	20
ConfidenceIntervals.C	Pass	
ErrorIntegral.C	Pass	
FittingDemo.C	Pass	
Ifit.C	Pass	
NumericalMinimization.C	Pass	
TestBinomial.C	Pass	
TwoHistoFit2D.C	Pass	
combinedFit.C	Pass	
exmapleFit3D.C	Pass	
fit1.C	crash, fit1_trace.txt	
fit1_C.C	compilation errors, fit1_C_log.txt	
fit2.C	Pass	
fit2a.C	Pass	
fit2d.C	Pass	

Most failures are due to 5 or 6 Cling issues

Status of Stress

```
[master] (macrdm) [430] ./stress -b 30
******************************************************************
*  Starting  R O O T - S T R E S S test suite with 30 events
******************************************************************
Test  1 : Functions, Random Numbers, Histogram Fits.............. OK
Test  2 : Check size & compression factor of a Root file........ OK
Test  3 : Purge, Reuse of gaps in TFile.......................... OK
Test  4 : Test of 2-d histograms, functions, 2-d fits.......... OK
Test  5 : Test graphics & Postscript............................. OK
Test  6 : Test subdirectories in a Root file..................... OK
Test  7 : TNtuple, selections, TCut, TCutG, TEventList.......... OK
Test  8 : Trees split and compression modes...................... OK
Test  9 : Analyze Event.root file of stress 8.................... OK
Test 10 : Create 10 files starting from Event.root.............. OK
Test 11 : Test chains of Trees using the 10 files............... OK
Test 12 : Compare histograms of test 9 and 11.................... OK
Test 13 : Test merging files of a chain.......................... OK
Test 14 : Check correct rebuilt of Event.root in test 13........ OK
Test 15 : Divert Tree branches to separate files................ OK
Test 16 : CINT test (3 nested loops) with LHCb trigger.......... OK
******************************************************************
*  SYS: Darwin macrdm.rademakers.org 12.4.0 Darwin Kernel Version 12
*  SYS: 10.8.4 Mac OS X
******************************************************************
stress    : Total I/O =   51.1 Mbytes, I =   33.2, O =  17.9
stress    : Compr I/O =   46.7 Mbytes, I =   29.9, O =  16.8
stress    : Real Time =   7.19 seconds Cpu Time =   6.96 seconds
******************************************************************
*  ROOTMARKS =1097.4   *  Root5.99/02   20130612/2149
******************************************************************
[master] (macrdm) [431]
```

ROOT 6 JIRA Gantt Chart

Gantt-Chart

within period -4w to 12w Update | daily | ROOT Open Tasks | MS-Powerpoint | MS-Project | Config

Issue		24. w (Jun. 2013)							25. w (Jun. 2013)							26. w (Jun. 2013)							27. w (Jul. 2013)							28. w (Jul. 2013)				
		10.	11.	12.	13.	14.	15.	16.	17.	18.	19.	20.	21.	22.	23.	24.	25.	26.	27.	28.	29.	30.	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.
6.00/00																																	
ROOT-142	Fully updated User's Guide																											?					
ROOT-141	Port of Eve/OpenGL to OpenGL/ES for iOS																																
ROOT-139	Cling fully integrated in ROOT, replacing CINT																																
ROOT-149	validate modules (PCM) as dictionary replaceme																																
ROOT-150	rootcling																																
ROOT-154	PyROOT moved to cling																																
ROOT-157	TCling's interpreter interfaces																																
ROOT-159	Validate TClass, I/O																																
ROOT-160	Validate name translation																																
ROOT-163	ShowMembers																																
ROOT-169	ROOT-view on cling Value-with-storage																																
ROOT-5160	Prevent reloading of definitions from PCM																																
ROOT-5161	Cross-PCM reuse																																
ROOT-5162	Validate use of multiple interpreters																																
ROOT-5163	Evaluate threading issues with cling																																
ROOT-5167	Update manual																																
ROOT-44	Clang/LLVM based interpreter backend																																
no fixed version																																	
ROOT-5266	Access to properties from selection.xml																																
ROOT-5121	Documentation																																
ROOT-5120	JavaScript Math/Latex																																
ROOT-5119	JavaScript ROOT Graphics																																
ROOT-5118	JavaScript ROOT I/O																																
ROOT-5117	ROOT & Cling on Windows																																
ROOT-5105	More / alternative cut expressions in TTreeViewe																																

Use agile project tracking tools

Risk Analysis

- We are still convinced that the choice to go for LLVM is the only viable way to move ROOT into the C++11 era
- LLVM is a project beyond critical mass, still growing quickly, still working on many advanced new features
- Cling basically works and shows that an LLVM based C++11 interpreter is possible
- The only plan B for the moment is more time and hard work
  - Hybrids of CINT/Cling, Reflex/Cling are not solutions (lessons learned from CINT7)
  - We will maintain v5-34-xx at least for the next two years

ROOT Roadmap

- Current stable version is v5-34-0x
  - It is an LTS (Long Term Support) version
  - New features will be back ported from the trunk
- Version v6-00-beta is scheduled for 27 Nov 2013
  - Has all features needed for migration of experiment frameworks
- Version v6-00 is scheduled for end of May 2014
  - Cling feature complete and a great successor to v5
  - It might not have Windows support (if not, likely in 6-02)
- Along the way several “Technology Previews” will be made available
  - Two days ago, June 12, we released 5.99.02 (Preview 2)

Conclusions

- To keep ROOT on the bleeding edge and future proof we need a modern C++ interpreter architecture that follows the C++ language evolution
- Choosing LLVM to build the interpreter is the right choice
- Delays in some essential developments in LLVM are the main cause for ROOT 6 rescheduling
- The needed features are now in the pipeline
- With ROOT 6 we return to our time driven release schedule
- And be in a good condition to benefit from the many exciting features LLVM will offer us