Software Engineering, Data stores, and Databases track summary

May 26, 2012
CHEP 2012

David Lange, LLNL
Simone Campana, CERN
Benedikt Hegner, CERN

http://news.yahoo.com/photos/fleet-week-kicks-off-in-nyc-slideshow/fleet-week-photo-1337796103.html

Abstract statistics

- 102 abstracts submitted
- 26 Presentations
- 71 Posters

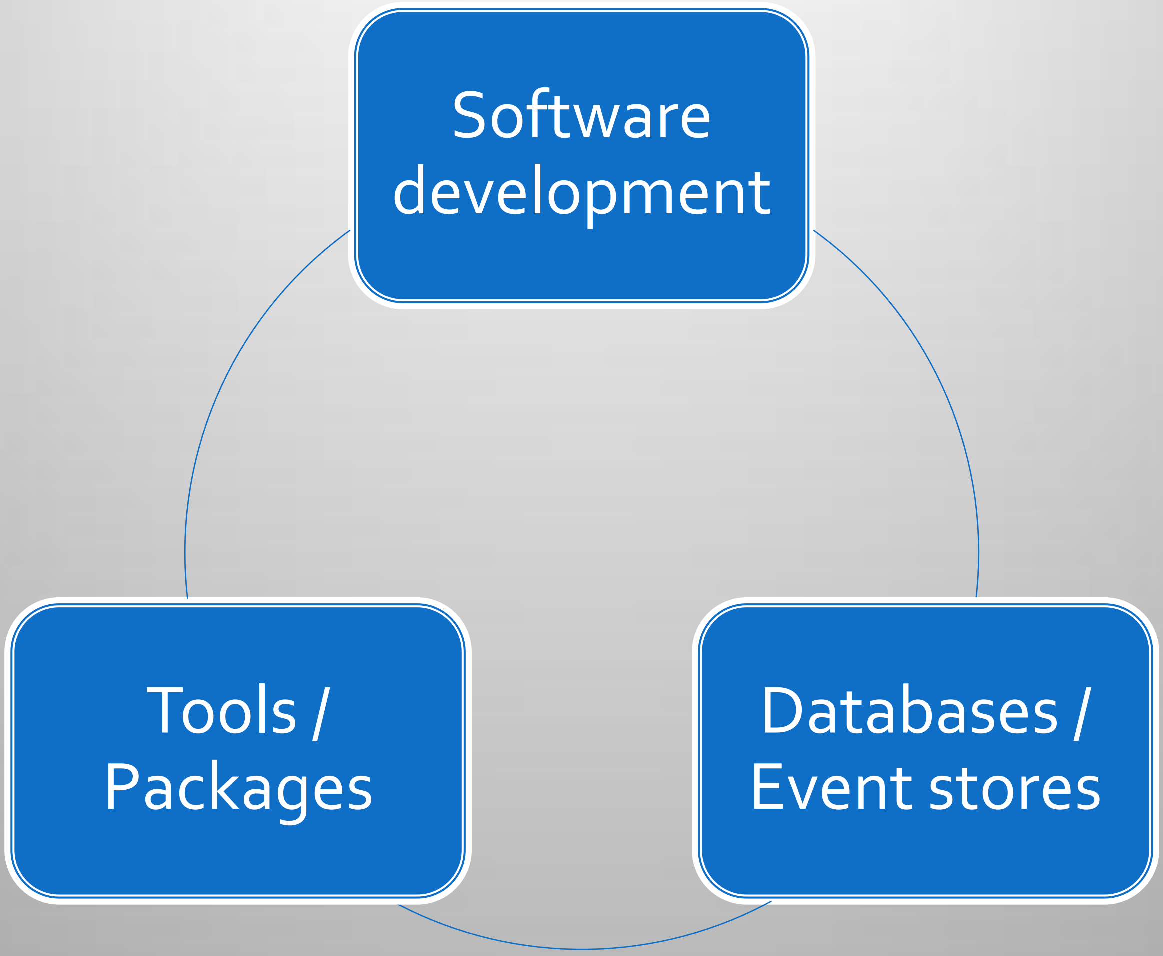
We had many excellent abstracts submitted.

Apologies to those submitters whose contribution
I did not manage to summarize here
(in particular the posters presented on Thursday)

Charge to the Software Engineering, Data stores, and Databases track

Programming techniques and tools	Software testing and quality assurance	Configuration management	Software build, release and distribution tools
Documentation	Foundation and utility libraries	Mathematical libraries	Component models
Object dictionaries	Scripting	Event stores	Metadata and supporting infastructure
	Databases		

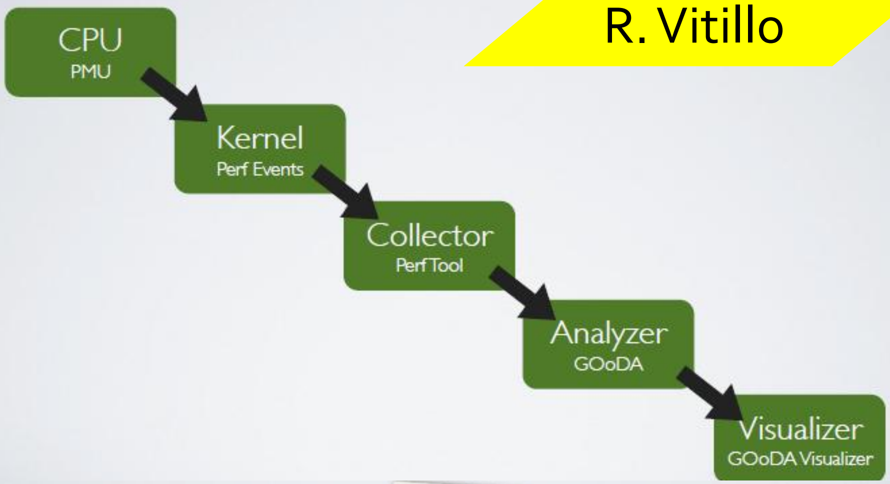
Our mapping onto sessions



Software development

Performance profilers: GOoDA and Intel VTune

R. Vitillo



function name	offset	length	module	process	unhalted_core_cycles		uop
					473185	(100%)	266508
⊞ operator new(unsigned lon…	0x134b0	0x3da	libtcmalloc_minimal.so	athena.py	12927	(100%)	5442
⊞ master.0.gbmagz_	0xfb80	0x4a0b	libBFieldStand.so	athena.py	13882	(100%)	5995
⊞ operator delete(void*)	0x12c10	0x2da	libtcmalloc_minimal.so	athena.py	7619	(100%)	3741
⊞ std::_Rb_tree_increment(s…	0x69c00	0x5a	libstdc++.so.6.0.10	athena.py	8633	(100%)	5697
⊞ get_bsfield_	0xed60	0xe16	libBFieldStand.so	athena.py	11407	(100%)	7809
⊞ Trk::STEP_Propagator::pro…	0x2b230	0x18e2	libTrkExSTEP_Propagator.so	athena.py	6337	(100%)	2792
⊞ Trk::RungeKuttaPropagator…	0x250e0	0x1051	libTrkExRungeKuttaPropagato…	athena.py	7589	(100%)	4478
⊞ ma27od_	0x22000	0x26ee	libTrkAlgebraUtils.so	athena.py	6397	(100%)	2083
⊞ Trk::FitMatrices::solveEq…	0x108a0	0x49a	libTrkiPatFitterUtils.so	athena.py	4935	(100%)	1701
⊞ deflate_slow	0x6850	0x976	libz.so.1.2.3	athena.py	5189	(100%)	2395

A. Mazurov

Gaudi configuration

```
from Configurables import IntelProfilerAuditor
profiler = IntelProfilerAuditor()
profiler.StartFromEventN = 5000
profiler.StopAtEventN = 15000
AuditorSvc().Auditors +=   [profiler]
```

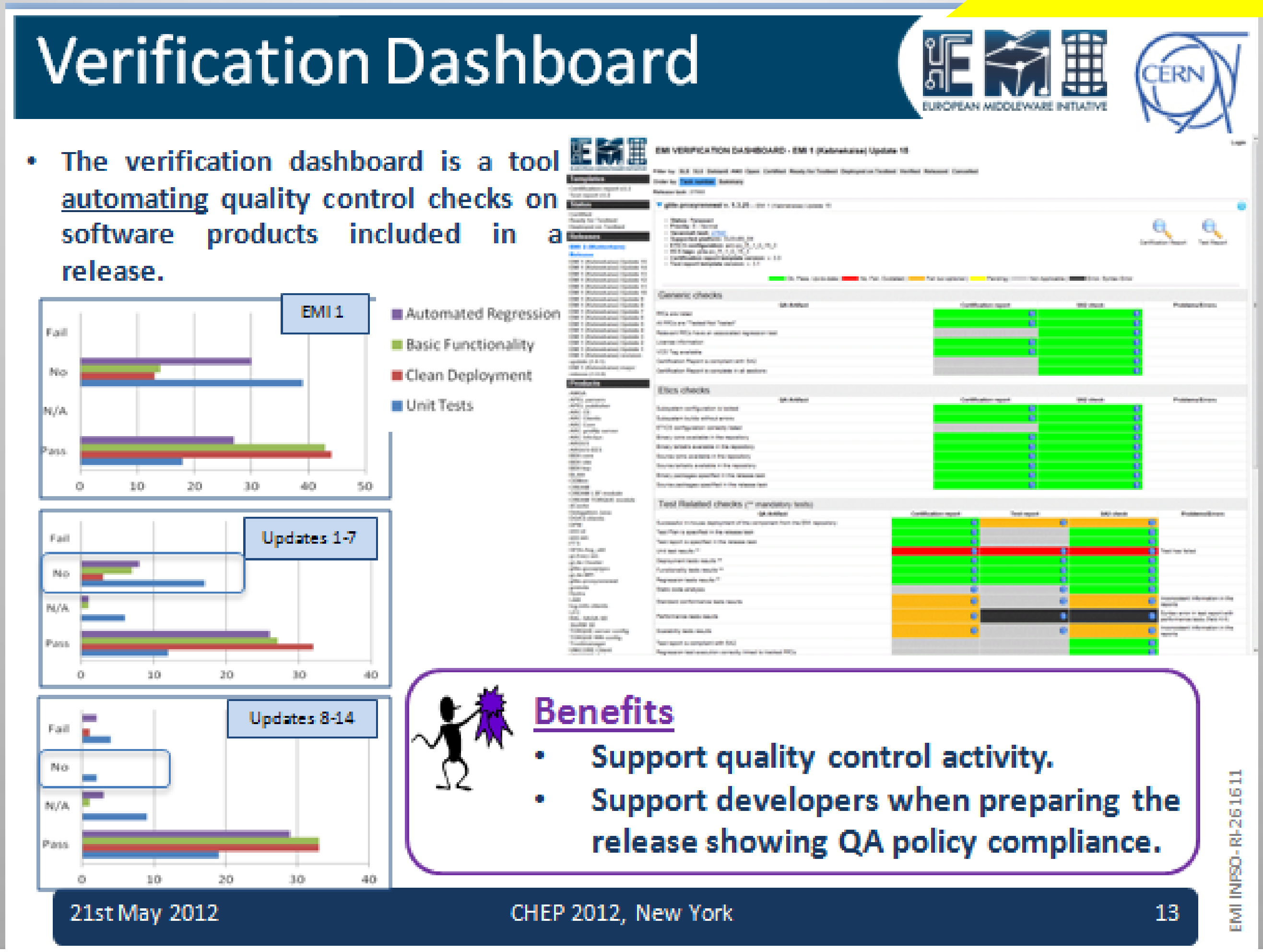
Run:
$> intelprofiler -o /collected/data job.py

Analyze (GUI):
$> amplxe-gui /collecter/data/r001hs

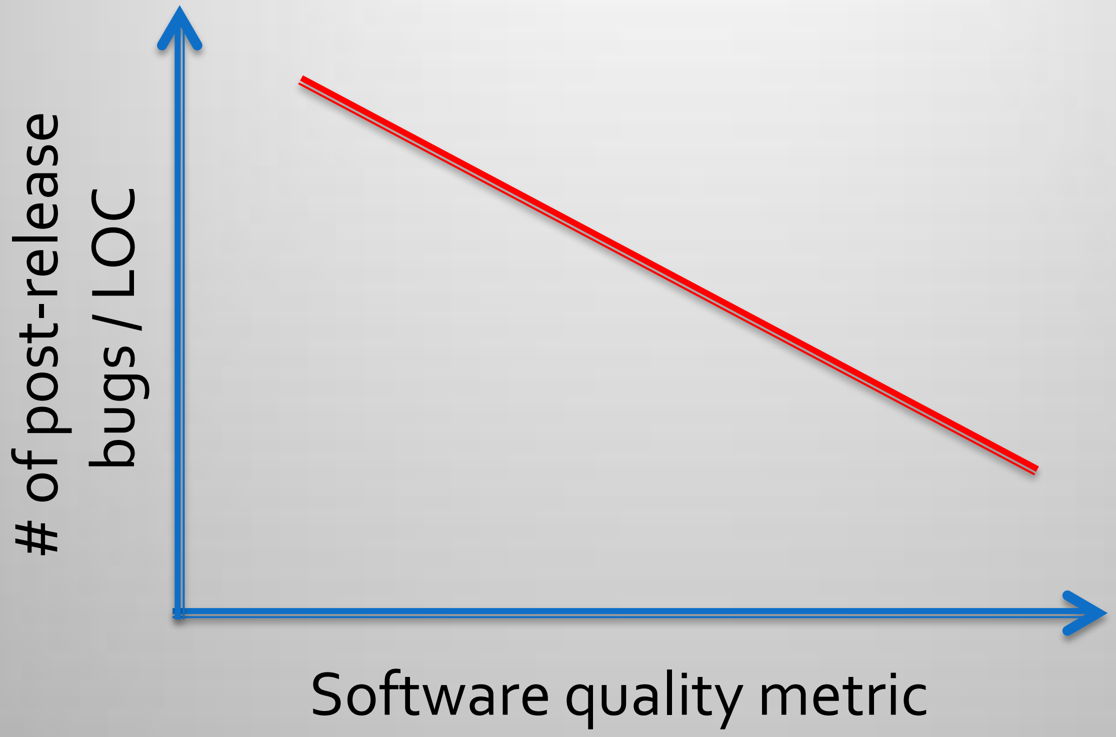
Analyze (CLI):
$> amplxe-cl -reports hotspots -r /collecter/data/r001hs

Software quality metric development

M. Alandes

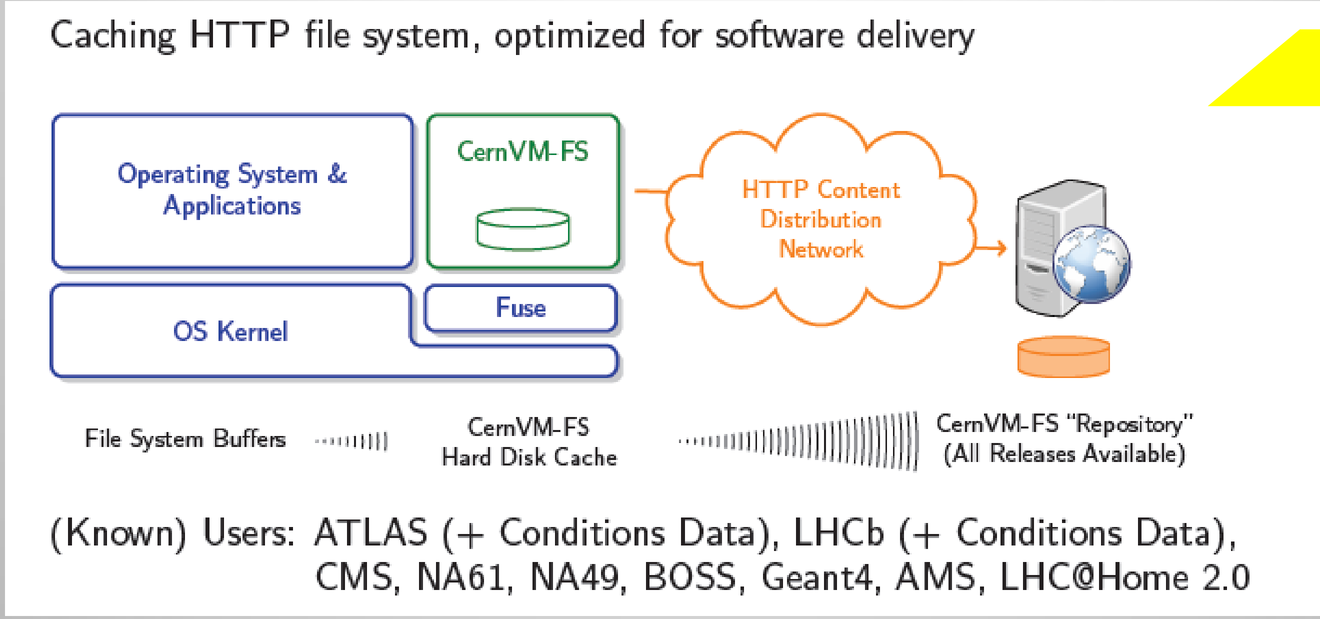


For CHEP 2013???

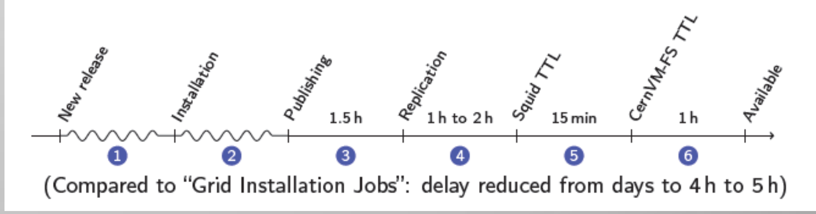


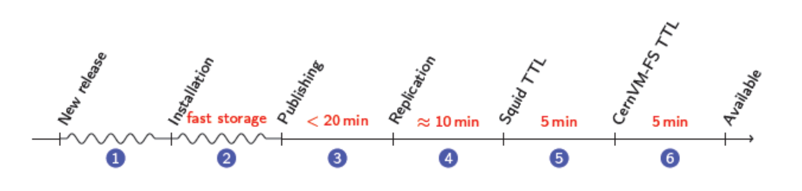
Software distribution: CERN VM-FS

J. Blomer



- Concentrating on improving latency for making software available. ATLAS release used as a metric:





LHCb migration to CMake build system

- Pros
  - projects and subdirectories
  - very powerful (complete) language
  - library of modules for configuration
  - extensible with functions and macros
  - *properties*
- Cons
  - no support for runtime environment
  - cannot override targets
  - transitivity of libraries, but not of includes

Something just fit, something not, but the language and the features are powerful enough to outweigh the limitations.

Current status

- CMT is a valid product, but with limits
- CMake is not meant to address our use case…
- … but it is powerful enough to be adapted
- Developed a CMake-based build framework
  - can replace CMT in LHCb use
  - better performance
  - will be adopted by after some more validation

M. Clemencic

From CMT to CMake (2)

```
# ====================================
package GaudiUtils
version v4r0
# ======= structure ====================
branches GaudiUtils src cmt doc
# ======= dependencies =================
use GaudiKernel *
use ROOT        * LCG_Interfaces
use AIDA        * LCG_Interfaces -no_auto_imports
use Boost       * LCG_Interfaces -no_auto_imports
use Reflex      * LCG_Interfaces -no_auto_imports
use uuid        * LCG_Interfaces -no_auto_imports
use XercesC     * LCG_Interfaces -no_auto_imports
# ======= own includes =================
apply_pattern install_more_includes more=GaudiUtils
# ======= constituents =================
library GaudiUtilsLib Lib/*.cpp \
        -import=AIDA -import=Boost -no_static
apply_pattern linker_library library=GaudiUtilsLib
# ======= constituents =================
library GaudiUtils component/*.cpp \
        -import=Boost -import=Reflex \
        -import=uuid -import=XercesC -no_static
apply_pattern component_library library=GaudiUtils
# ======= local settings ===============
private
 macro_append ROOT_linkopts " -lHist -lXMLIO "
 macro_append Boost_linkopts  " $(Boost_linkopts_date_time) "
end_private
```

```
# ==============================
gaudi_subdir(GaudiUtils v4r0)

# ======= dependencies =================
depends_on_subdirs(GaudiKernel)

find_package(ROOT COMPONENTS RIO Hist XMLIO)
find_package(AIDA)
find_package(Boost COMPONENTS date_time)
find_package(uuid)
find_package(XercesC)

# ======= libraries =======
gaudi_add_library(GaudiUtilsLib Lib/*.cpp
                  LINK_LIBRARIES GaudiKernel Boost R
                  INCLUDE_DIRS AIDA Boost ROOT
                  PUBLIC_HEADERS GaudiUtils)
gaudi_add_module(GaudiUtils component/*.cpp
                   LINK_LIBRARIES GaudiUtilsLib uuid X
                   INCLUDE_DIRS uuid XercesC)
```

MetaData:
Make sure it is there when users need it

D. Malon

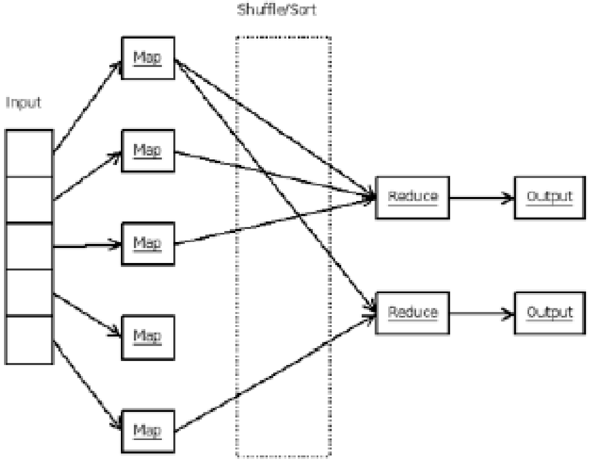
- Metadata are integral to every aspect of ATLAS computing
- The intent of this presentation has been to provide an illustrative view of ATLAS metadata, principally from the point of view of the infrastructure and services needed for metadata flow in the context of a single task
- While metadata components and infrastructure have grown organically as the experiment has matured, a number of principles described herein have informed their design and connectivity
- The infrastructure continues to evolve in a variety of ways, with improvements planned
  - to how dataset-level metadata may be used to reduce the need for peeking into input files,
  - to how metadata are emitted and transported from executing jobs to the collaboration's metadata repositories,
  - to machinery for robust accounting of low-rate error conditions in physics data bookkeeping

Software framework redesign for MICE software (MAUS) to map-reduce

- Plugin design
  - MapReduce framework (Hadoop/Google)
  - input-map-reduce-output
- **Input:** Read in data
  - Access the socket
  - Read in binary DAQ data file or Read in beam for Monte Carlo
- **Map:** Process spill/events & return modified data
  - Monte Carlo simulation
  - Detector reconstruction
- **Reduce:** Summarize data from mapped spills
  - Detector performance plots, accelerator etc
- **Output:** Write out data
  - Write out in ROOT/JSON format

D. Rajaram

- map: User specifies operation on single event
- reduce: User specifies operation on all events



Adapting software for tomorrow's hardware trends

A. Nowak

- Pricing follows market pressure, not technology
- Vectors – growing substantially
  - AVX: 256 bits, designed for more
  - AVX: new execution units
  - LRBni (Intel MIC): 512 bits, new vector instructions, FMA, 3-4op
- x86 microarchitecture
  - steady, but limited improvements (<10% per "tock")
  - increasingly advanced features – can HEP benefit?
- Frequency – very modest changes, if any
- IO, disk and memory do not progress at the same rate as compute power
  - bytes/FLOP decreasing
  - pJ/FLOP decreasing

- # of cores "at home" grows arithmetically
  - various reasons, most linked to the way people use their computers
- # of cores in the enterprise space still grows geometrically
- The number of cores in the datacenter grows between the two, will slow down in the long run
  - The trend is important, not the end amount
- Sockets – slight growth with a limit, ultimately impacts core count per platform
- Two factors to consider:
  - Enterprise and HPC-targeted developments "trickle down" to support datacenter developments (where cost effective)
  - Heterogeneous architectures – cross platform, cross socket, hybrid CPUs, accelerators, split into throughput and classic computing

Recommendations

- introduce a systematized R&D program focused on parallelism with deliverables
- restate the real needs of the HEP community starting with a tabula rasa
- setting clear, realistic, contextualized goals for development in line with the real needs of the HEP community
- devising better metrics for performance and taxing for violations
- implementing a scalability process focused on rapid response
- promoting joint work across stakeholders to share the load
- a careful embrace of emerging technology
- a conscious consideration of where any performance gains should be reinvested (e.g. reduced cost, improved raw performance, accuracy etc)

CMS implementation of vectorized math libraries and prototype parallel track seeding

T. Hauth

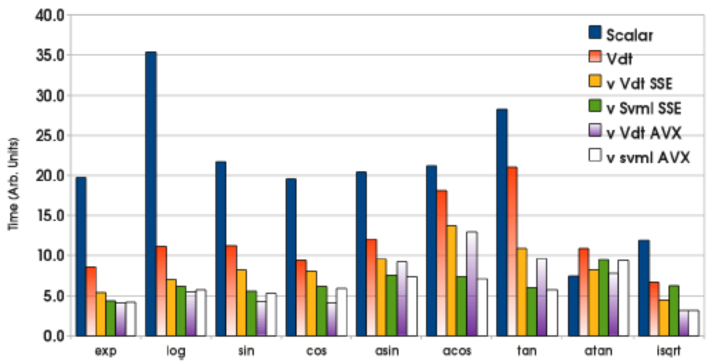
Double Precision Fast Transcendental Functions

Many open source mathematical libraries are available but...

- Only a few treat double precision numbers
- None is easily vectorizable with various SIMD instruction sets (SSE, AVX, ..)

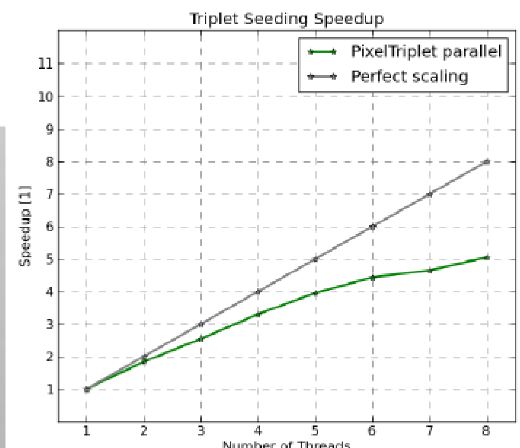
We created a set of auto-vectorizable math functions for double precision, called **vdt math**

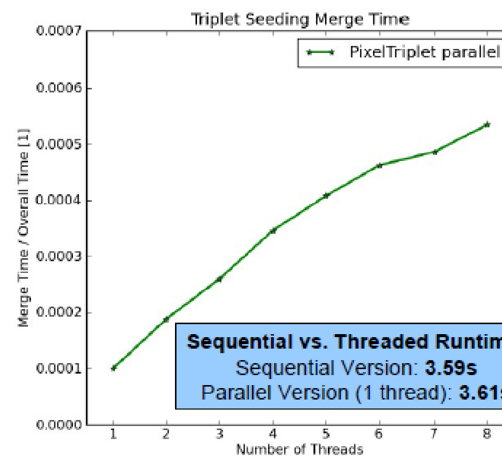
- Start from good-old Cephes library (Padé approximates)
- A multitude of useful math functions are included: `inverse square root, exp, log, sin, cos, tan, asin, acos, atan`
- Very good approximation of stdlib math functions (see backup for details)



Triplet Seeding Runtime and Scaling

- Good scaling up to five cores
- Compared to the overall runtime of the algorithm, the final merge step only takes about .1 to .3 percent of the triplet seeding time
- This depends on the number of threads: for more threads more blocks are partitioned



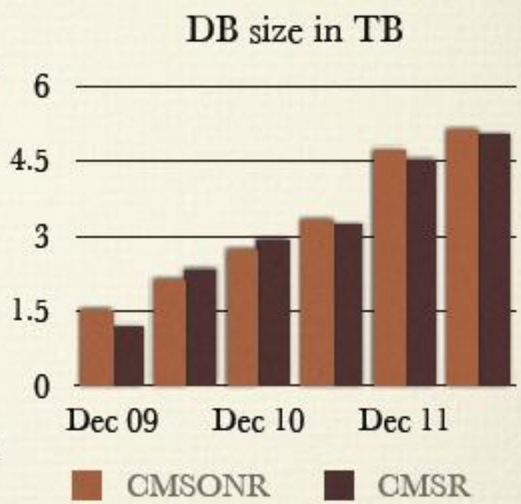


Sequential vs. Threaded Runtime:
Sequential Version: **3.59s**
Parallel Version (1 thread): **3.61s**

Databases / event stores

CMS online and offline databases

- DB growth about 1.5 TB/yr
  - both online and offline
- Condition data is only a small fraction
  - ~ 300 GB at present
  - growth: + 20 GB/yr
  - about 50 Global Tags created each month



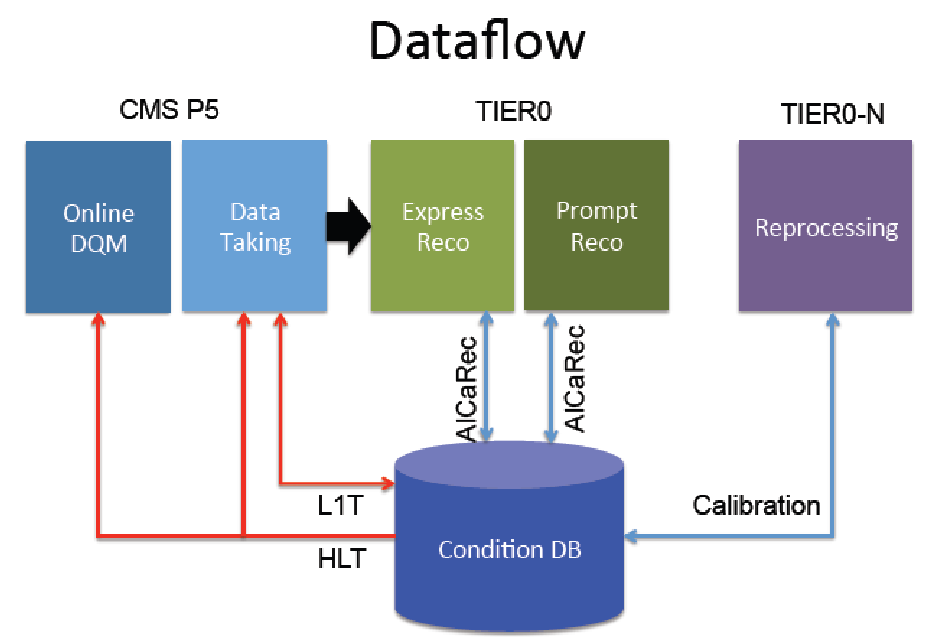
A. Pfeiffer

Smooth operations was a theme of DB session

- Very smooth running
  - CMSONR availability: **99.88 %**
    - 10.5 hours downtime overall in 2011
  - CMSR availability: **99.64 %**
    - 30.7 hours downtime overall in 2011
  - SQL query time stable (few msec)

downtime includes all power-cuts, node reboots, hangs, (some) maintenance, …

CMS conditions operations stable



Most of the work is currently spent in operation

- Follow-up of data taking and processing needs
- Migration of existing data sets to a new CMS proprietary format

Only little development are still ongoing

Focus of the current phase is consolidation of the (still) critical areas

- Bookkeeping system for the DropBox
- Security for DB access (authentication and authorization)
- Improvements for Monitoring System
- Handling of schema evolution for Blob-based storage

No major changes are foreseen in the system for 2 years

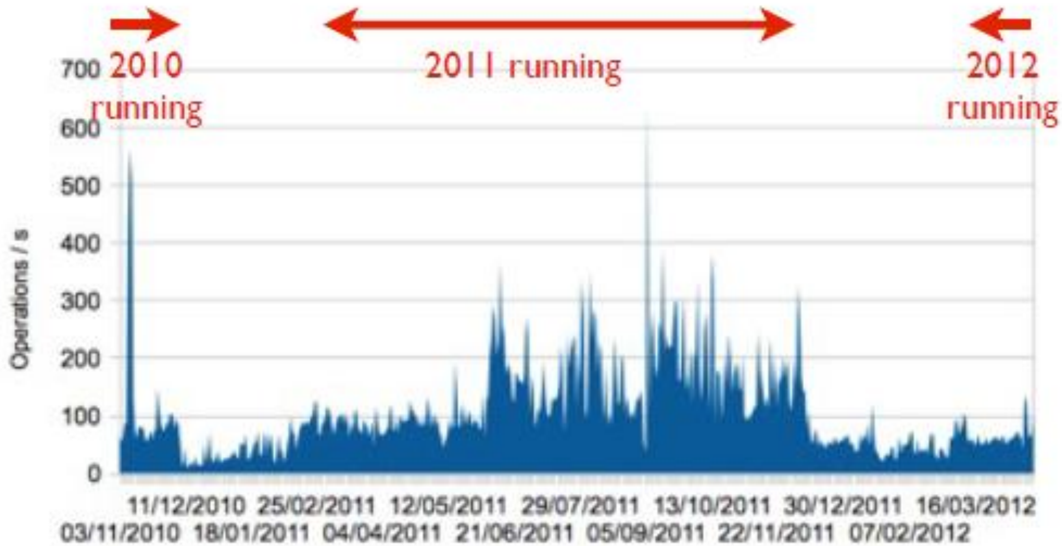
G. Govi

Atlas conditions switch to Frontier successful

A. Dewhurst

- Tier 0 access switched to Frontier from direct access for the start of 2012 data taking

Average number of operations /s per day for the RAL database



Structured storage (aka NoSQL) systems

- So what is NoSQL, pardon, structured storage about?
  - 1. ***Non-relational modelling and storage of data***
    - Use the native data layout of an application
  - 2. ***Linear scalability of data processing***
    - Scalability ≠ Performance

Use cases considered:

1. Log file aggregation
2. Trace mining
3. HTTP cache for dataset downloads

- Structured storage systems are too useful to be ignored
- Hadoop proved to be the correct choice and an excellent platform fc analytical workloads
  - Stable – reliable – fast – easy to work with
  - Survived disastrous hardware failures
- DDM use cases well covered
  - Storage facility (*log aggregation, traces, web sharing*)
  - Data processing (*trace mining, accounting, searching*)
- Miscellaneous
  - All three evaluated products provide full durability, and transactions were
  - We see Hadoop complementary to RDBMS, not as a replacement

- Within one year we had
  - 5 disk failures
    - 20% failure rate!
    - Out of which 3 happened at the same time
  - 1 Mainboard failure
    - Together with the disk failure, but another node
- Worst case scenario experienced up to now
  - 4 nodes out of 12 dead within a few minutes
  - Hadoop
    - Reported erroneous nodes
    - Blacklisted them
    - And resynced the remaining ones
  - No manual intervention necessary
  - Nothing was lost

M. Lassnig

Comparison of Frontier to NoSQL systems

D. Dykstra

	Frontier	NoSQL in general
DB structure	Row/column	Nested key/value
Consistency	ACID DB, eventual reads	Eventual
Write model	Central writing	Distributed writing
Read model	Many readers same data	Read many different data
Data model	Central data, cache on demand	Distributed data, copies
Distributed elements	General purpose	Special purpose

	MongoDB	CouchDB	HBase	Cassandra	Frontier
Stored data format	JSON	JSON	Arbitrary	Arbitrary	SQL types
Flexible queries	Yes	No	No	No	Yes
Distributed write	No	Yes	No	Yes	No
Handles Slashdot Effect	No	Yes, best w/squid	If scaled sufficiently	If scaled sufficiently	Yes
Does well with many reads of different data	Yes	Yes	Yes	Yes	No
RESTful interface	No	Yes	Add-on	No	Yes
Consistency	Eventual	ACID DB, eventual read	Mixed	Tunable	ACID DB, eventual read
Distributed MapReduce	No	No	Yes	Add-on	No
Replication	Few copies	Everything	Tunable	Tunable	Caching

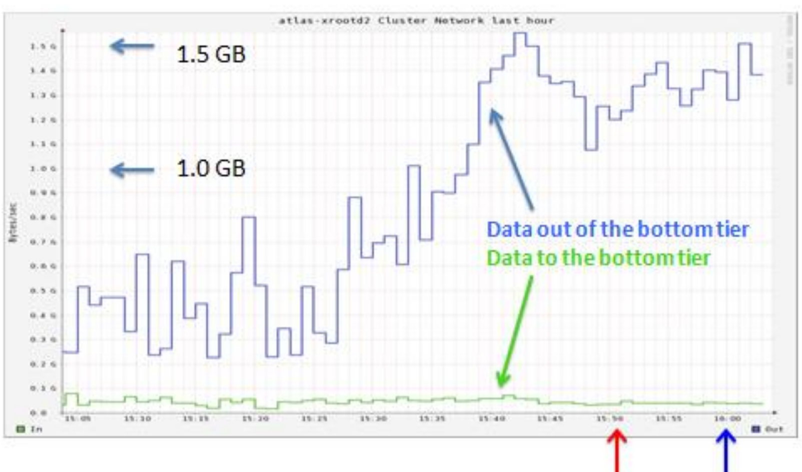
- NoSQL databases have a wide variety of characteristics, including scalability
- Frontier+Squid easily & efficiently adds some of the same scalability to relational databases when there are many readers of the same data
  - Also enables clients to be geographically distant
- CouchDB with REST can have same scalability
- Hadoop HBase has most potential for big apps
- There are good applications in HEP for many different Database Management Systems

XrootD: Tiered storage system

Why Divide Storage into Tiers?

- ALTAS production jobs stage input files to batch nodes, **BUT** analysis jobs read directly from Xrootd storage
- Need high performance storage to serve the random/sparse IO from analysis jobs
- Data becomes cold quickly

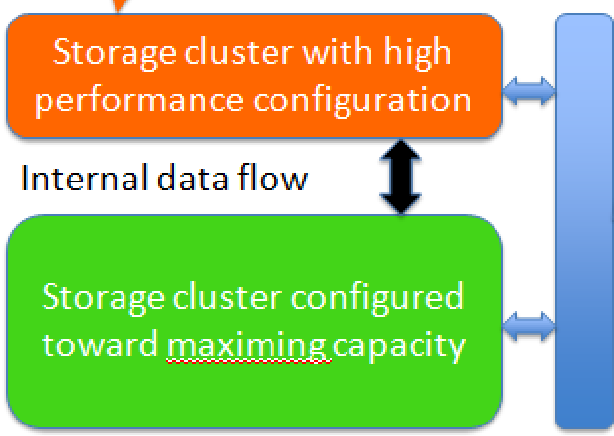
W. Yang



Activity at two time spots 10 minutes apart: 15:50 16:00

Top tier entrance:

- GridFTP data import (over WAN)
- Direct reading by analysis jobs
- All job outputs



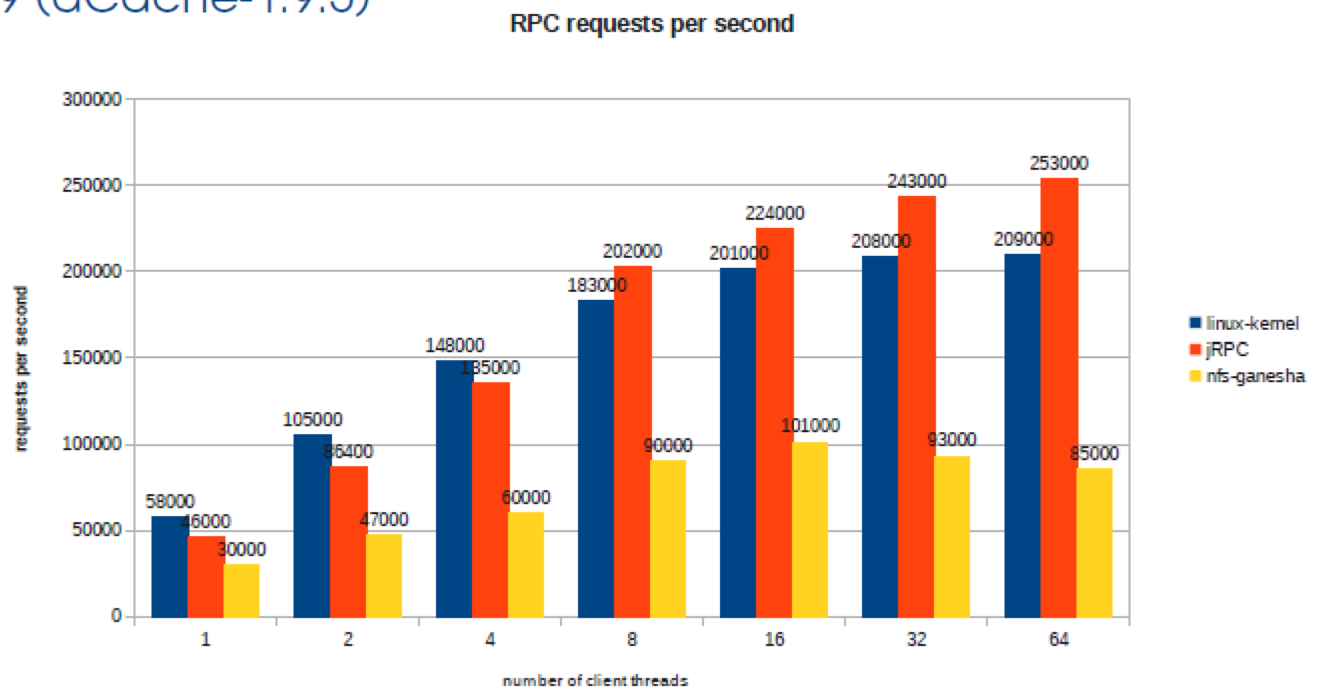
All tier entrance sees all storage:

- SRM & Data management
- Data stage-in by production jobs
- GridFTP data export

dCache: Novel RPC implementation

- High performance RPC library
- Compatible with existing standards
- Meets today's requirements
  - `IPv6, AES256`
- In production since 2009 (dCache-1.9.5)

T. Mkrtchyan



Expert systems: Adding automatic capacity to computing systems

C. Haen

problems

- Huge workload per person.
- Night on-call duty.
- Potential loss of knowledge when a student leaves.

Solution

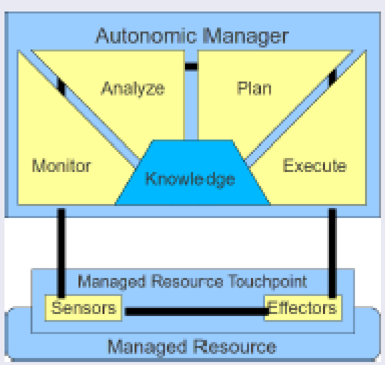
A software that would

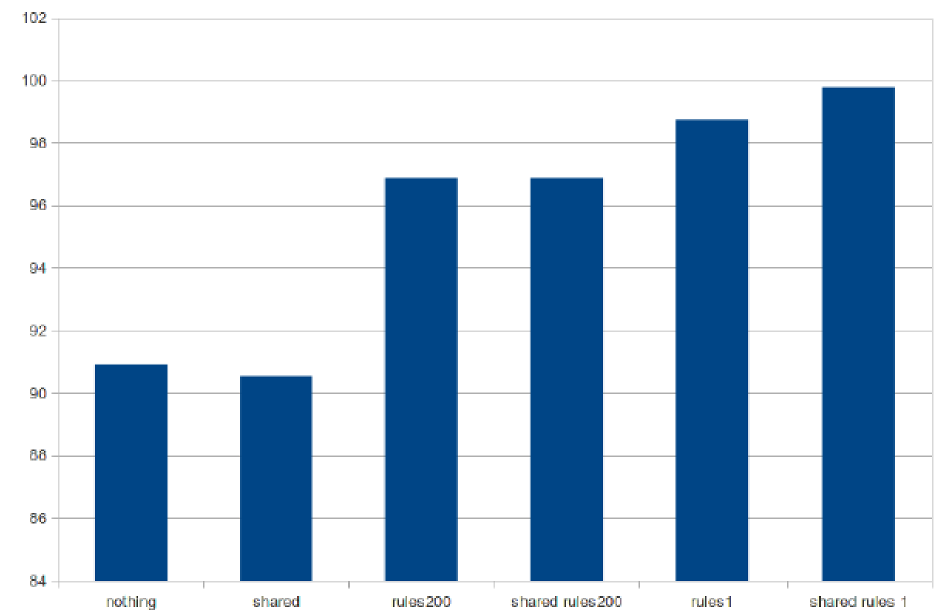
- act as a knowledge base
- act as a history database
- improve with experience

Final goal: ease the work of our system administrators

Challenging testbed problem: Multiple simultaneous failures to diagnose in correct order

MAPE-K loop





Percentage of problems diagnosed

Tools / Packages

LCG Persistency framework: Projects consolidation

R. Trentadue

PF usage in the experiments

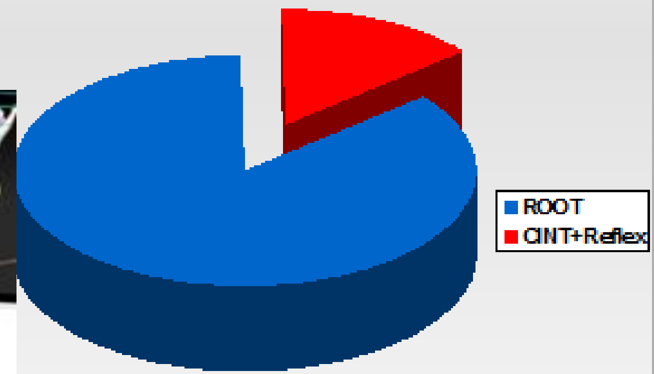
CERN IT Department

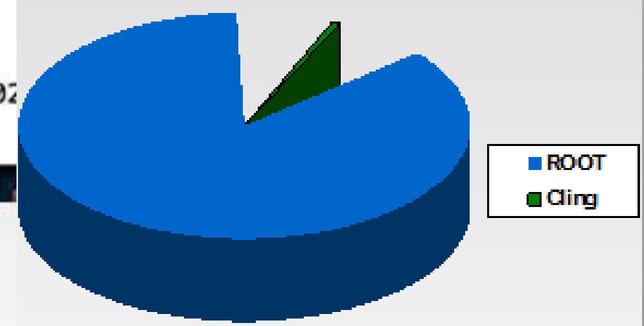
Persistency Framework in the LHC experiments	ATLAS	CMS	LHCb
CORAL (Oracle, SQLite, XML authentication and lookup)	Conditions data (COOL) Geometry data (detector descr.) Trigger configuration data Event collections/tags (POOL)	Conditions data Geometry data (detector descr.) Trigger configuration data	Conditions data (COOL)
CORAL + Frontier (Frontier/Squid)	Conditions, Geometry, Trigger *(R/O access in Grid, Tier0)*	Conditions, Geometry, Trigger *(R/O access in Grid, HLT, Tier0)*	--- *(will be tested in 2012)*
CORAL Server (CoralServer/CoralServerProxy)	Conditions, Geometry, Trigger *(R/O access in HLT)*	---	---
CORAL + LFC (LFC authentication and lookup)	---	---	Conditions data *(authentication/lookup in Grid)* *(will be dropped in 2012)*
COOL	Conditions data	---	Conditions data
POOL (ROOT storage service)	Event data Event collections/tags Conditions data (payload)	---	Event data *(dropped in 2011)*
POOL (Collections – ROOT and Relational)	Event collections/tags	---	---

Cling replacing CINT in ROOT 6 (November 2012)

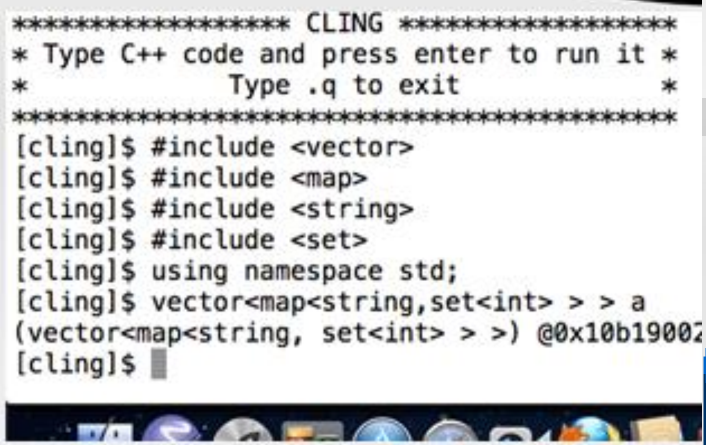
V. Vassilev

- Much less code to maintain





Cling Is Better Than CINT



- Full C++ support
  - STL + templates
  - Path to C++11
- Correctness
- Better type information and representations
- Always compile in memory

Example of turning compiler into interpreter: Function unloading

```
[cling]$ .L Calculator.h
[cling]$ Calculator calc;
[cling]$ calc.Add(3, 1)
[cling]$ (int) 2  //WTF!?*
[cling]$ .U Calculator.h
[cling]$ .L Calculator.h
[cling]$ Calculator calc;
[cling]$ calc.Add(3, 1)
[cling]$ (int) 4  //☺
```

```
// Calculator.h
class Calculator {
  int Add(int a, int b) {
    return a - b;
  }
...
};
```

```
// Calculator.h
class Calculator {
  int Add(int a, int b) {
    return a + b;
  }
...
};
```

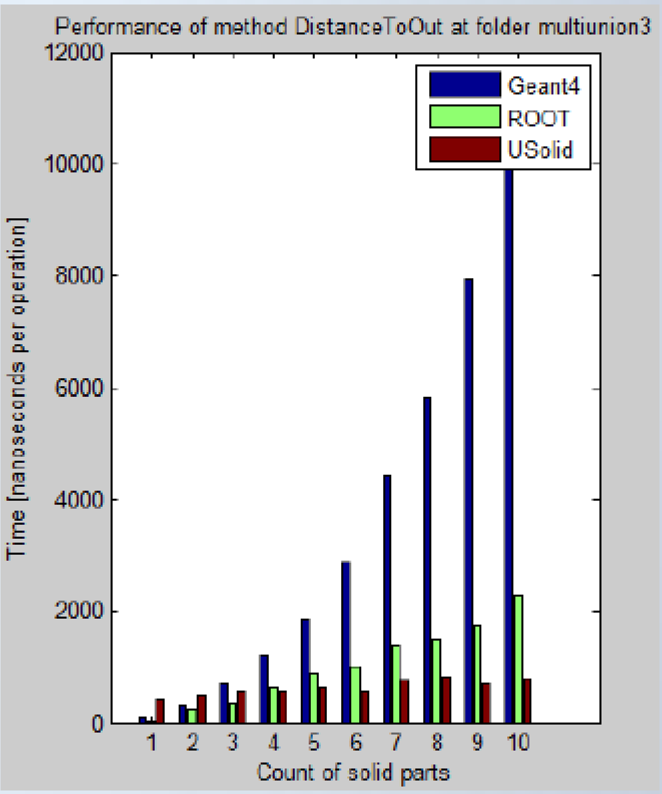
* What's That Function

New, common, solids library for root and Geant4

Motivations

M. Gayer

- Optimize and guarantee better long-term maintenance of Root and Gean4 solids libraries
  - A rough estimation indicates that about 70-80% of code investment for the geometry modeler concerns solids, to guarantee the required precision and efficiency in a huge variety of combinations
- Create a single library of high quality implementations
  - Starting from what exists today in Geant4 and Root
  - Adopt a single type for each shape
  - Create a new Multi-Union solid
  - Aims to replace solid libraries in Geant4 and Root
  - Allowing to reach complete conformance to GDML solids schema



- Significant performance speed up in some common tasks

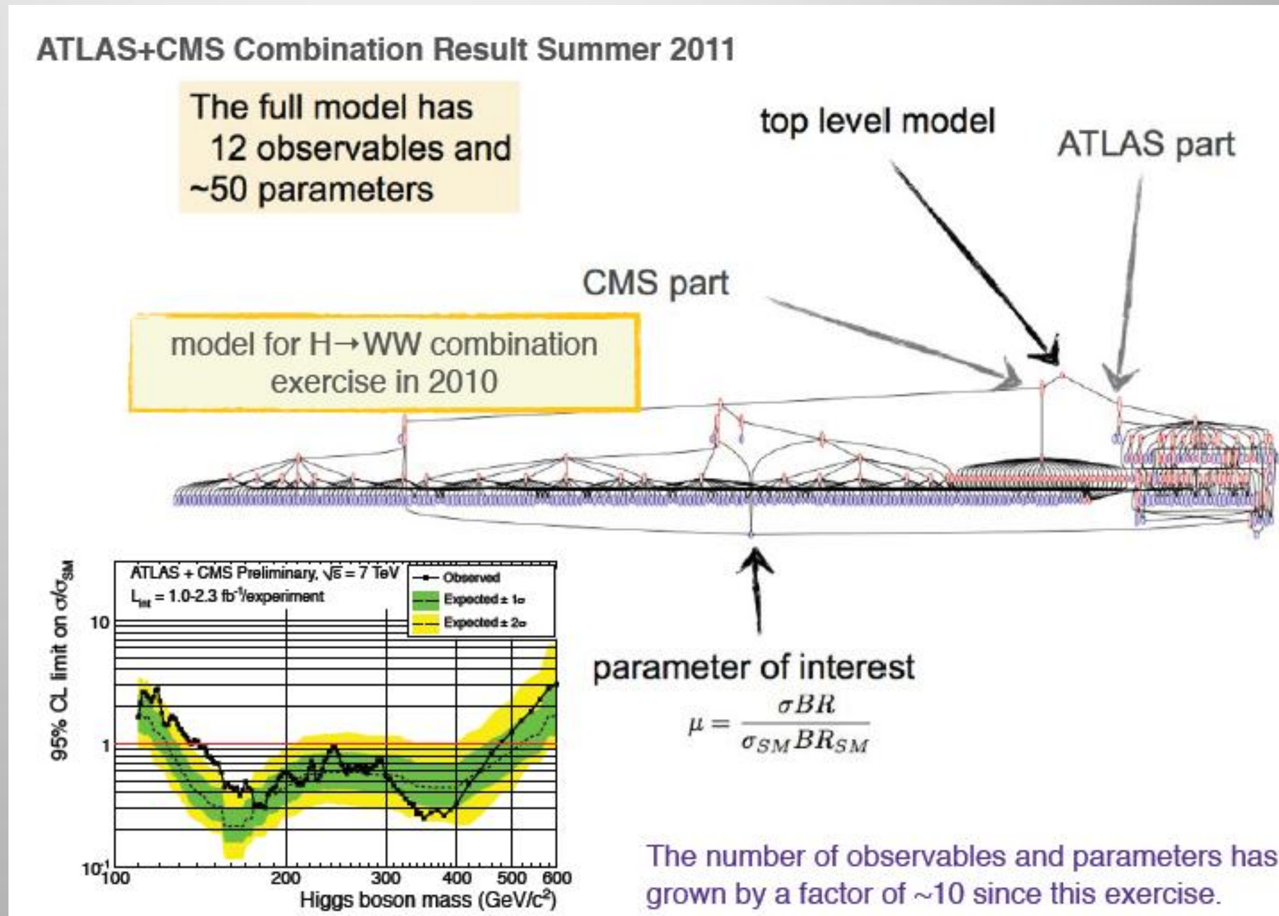
RooStats: common statistical tools library based on root (and RooFit)

S. Kreiss

Available tools

HypoTestCalculators
- AsymptoticCalculator
  - calculates a p-value according to an analytic expression for the asymptotic form of the test statistic distribution
- FrequentistCalculator
  - frequentist calculation (profile nuisance parameters)
- HybridCalculator
  - hybrid Bayes-Frequentist calculation (marginalize nuisance parameters)
- ProfileLikelihoodCalculator
  - the method of MINUIT/MINOS, based on Wilks' theorem

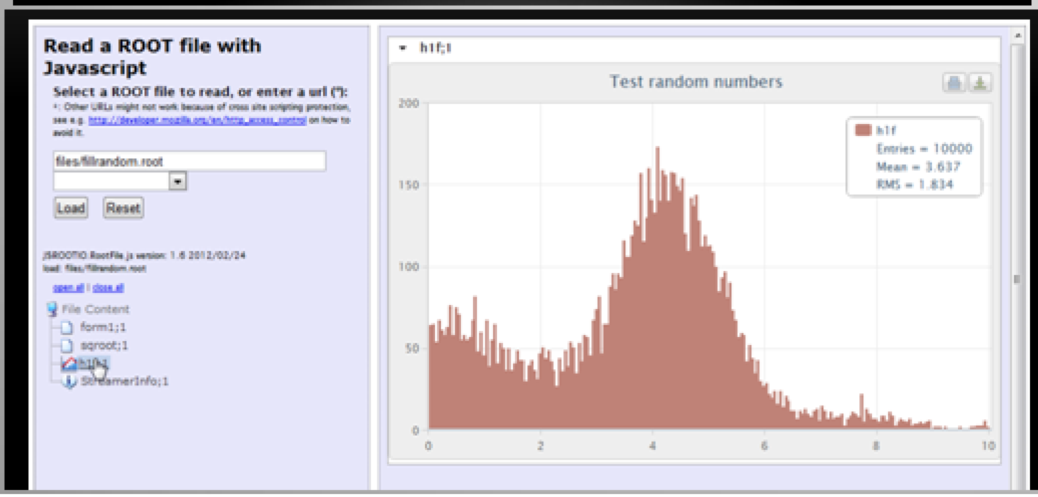
IntervalCalculators
- HypoTestInverter
  - takes a HypoTestCalculator and forms an IntervalCalculator
- ProfileLikelihoodCalculator
  - method of MINUIT/MINOS, based on Wilks' theorem
- NeymanConstruction
  - general purpose Neyman Construction class, highly configurable: choice of TestStatistic, TestStatSampler (defines ensemble/conditioning), integration boundary (upper, lower, central limits), and parameter points to scan
- FeldmanCousins
  - specific configuration of NeymanConstruction for Feldman-Cousins (generalized for nuisance parameters)
- MCMCCalculator
  - Bayesian Markov Chain Monte Carlo (Metropolis Hastings), proposal function is highly customizable
- BayesianCalculator
  - Bayesian posterior calculated via numeric integration routines, currently only supports one parameter



Browsing root files via JavaScript

B. Bellenot

- How to share thousands of histograms on the web, without having to generate picture files (gif, jpg, …)?
- How to easily share a ROOT file?
- How to browse & display the content of a ROOT file from any platform (even from a smartphone or tablet)?
- Online monitoring?
- And obviously, all that without having to install ROOT anywhere?



- Uses HighCharts JavaScript charting library

HOW TO USE IT?

- Simply copy the ROOT file(s) anywhere on the web
- Create a simple html page next to the files
  - Only two lines have to be added in the <head>

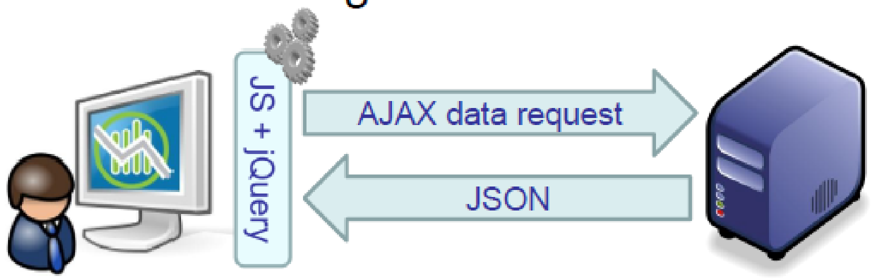
```
<head>
   <title>Read a ROOT file in JavaScript (Demonstration)</title>
   <link rel="stylesheet" type="text/css" href="style/JSRootInterface.css" />
   <script type="text/javascript" src="scripts/JSRootInterface.js"></script>
</head>
```

- And a few lines in the <body>. Here is a complete example:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
   <head>
      <title>Read a ROOT file in Javascript (Demonstration)</title>
      <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
      <link rel="stylesheet" type="text/css" href="style/JSRootInterface.css" />
      <script type="text/javascript" src="scripts/JSRootInterface.js"></script>
   </head>
   <body onload="BuildSimpleGUI()">
      <div id="simpleGUI" files="file_1.root;file_2.root;file_n.root;"></div>
   </body>
</html>
```

JavaScript analysis in dashboard

- Client-side view generation



- Single page interface
  - GUI-style
  - Data loaded on-demand

D. Tuckett

Technology Cocktail

CERN IT Department

- jQuery Core & UI
  - DOM manipulation
  - UI widgets
  - Popular! →
- Plugins
  - URL hash: BBQ
  - MVC events: Backbone
  - Templating: Handlebars
  - Plotting: Highcharts
  - Tables: DataTables
  - Utilities: Underscore
  - …

Hbrowse for hierarchical data

- ATLAS Task Analysis
  https://dashb-atlas-prodsys-prototype.cern.ch/templates/task-analysis/#timerange=lastMonth&demo=on
- CMS Interactive View
  http://dashb-cms-job.cern.ch/dashboard/templates/web-job2/
- ATLAS Dataset distribution
  http://dashb-atlas-task.cern.ch/templates/pandadatasetdist/
- …

Xbrowse for matrix data

- ATLAS DDM Dashboard
  http://dashb-atlas-data.cern.ch/ddm2/
- WLCG Transfers Dashboard
  http://dashb-wlcg-transfers.cern.ch/ui/
  (See poster: *[289] Providing WLCG Global Transfer Monitoring*)

PyPy provides python syntax with C++ speed

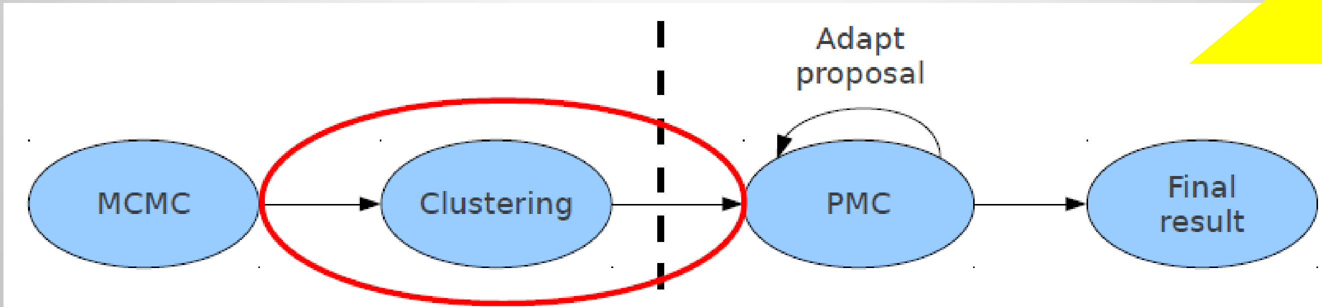
R. Vitillo

- Original results:
  - C++ .... 10,000,000 “events”: 1.26 secs (1x)
  - Python ..... 10,000,000 “events”: 68.7 secs (55x)
- Exact same Python code, but now JIT-ed TTree:
  - PyPy .... 10,000,000 “events”: 3.45 secs (2.7x)

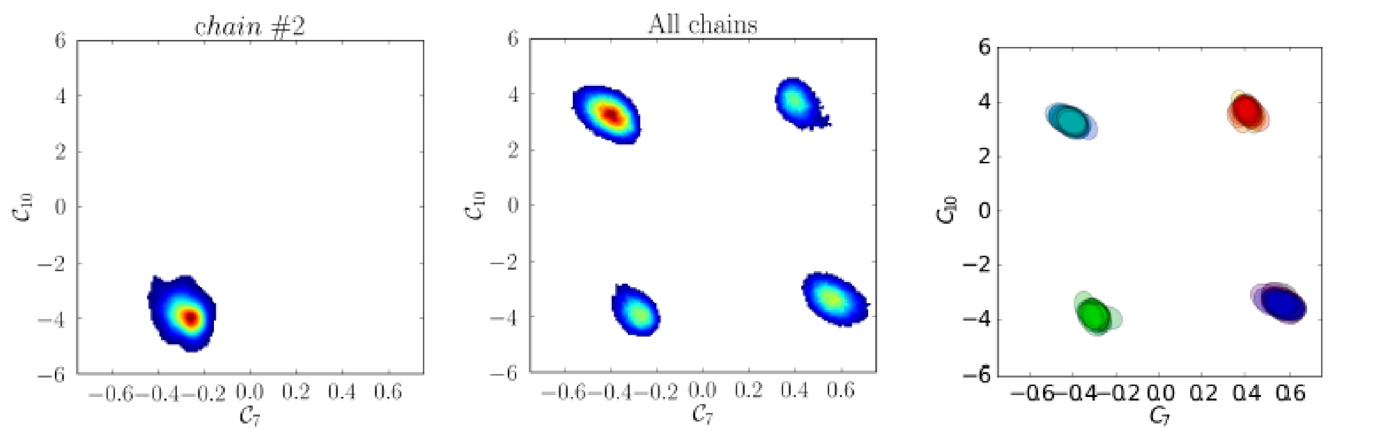
Huge improvement in Python-based ROOT I/O has been achieved using PyPy's tracing JIT!

MCMC with population Monte Carlo in BAT

C. Haen



- Example problem to illustrate ability of PMC to identify multimodal solutions in complex problem: 22 observables, 28 nuisance parameters



Single chain – single mode | 50 chains – four groups | 35 components per group

Summary

- Our track had a very nice set of presentations and posters with considerable discussion during our parallel sessions

Main themes:

- Stability in operations
- Adapting community solutions, industry collaboration
- Time to think ahead to ensure our software is ready new computing technologies