Adding bridges to ROOT

Bianca-Cristina Cristescu

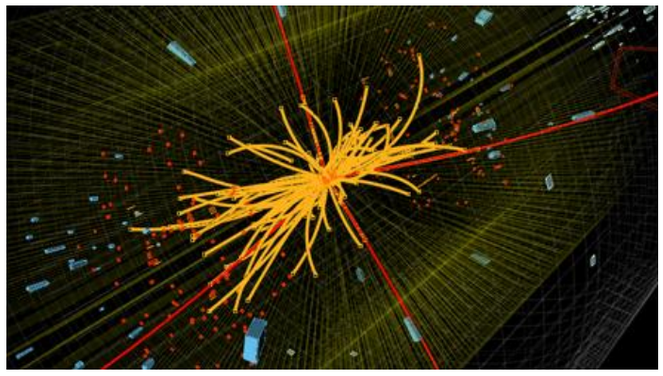
Contents

- Working plan
- Bridges “built”
- Software development model
- Launching with ROOT6

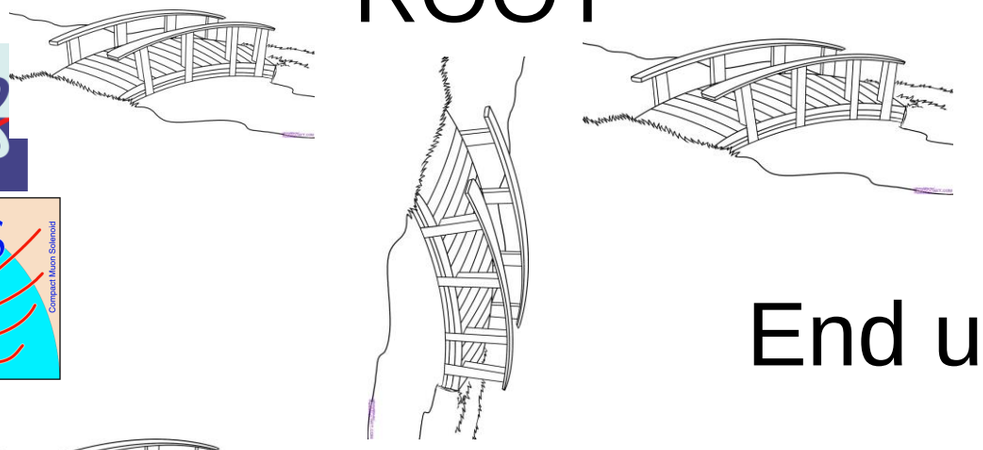
Working plan

- Build up the connection between ROOT and Cling: Reflection
- End-user features

=> Build up some bridges

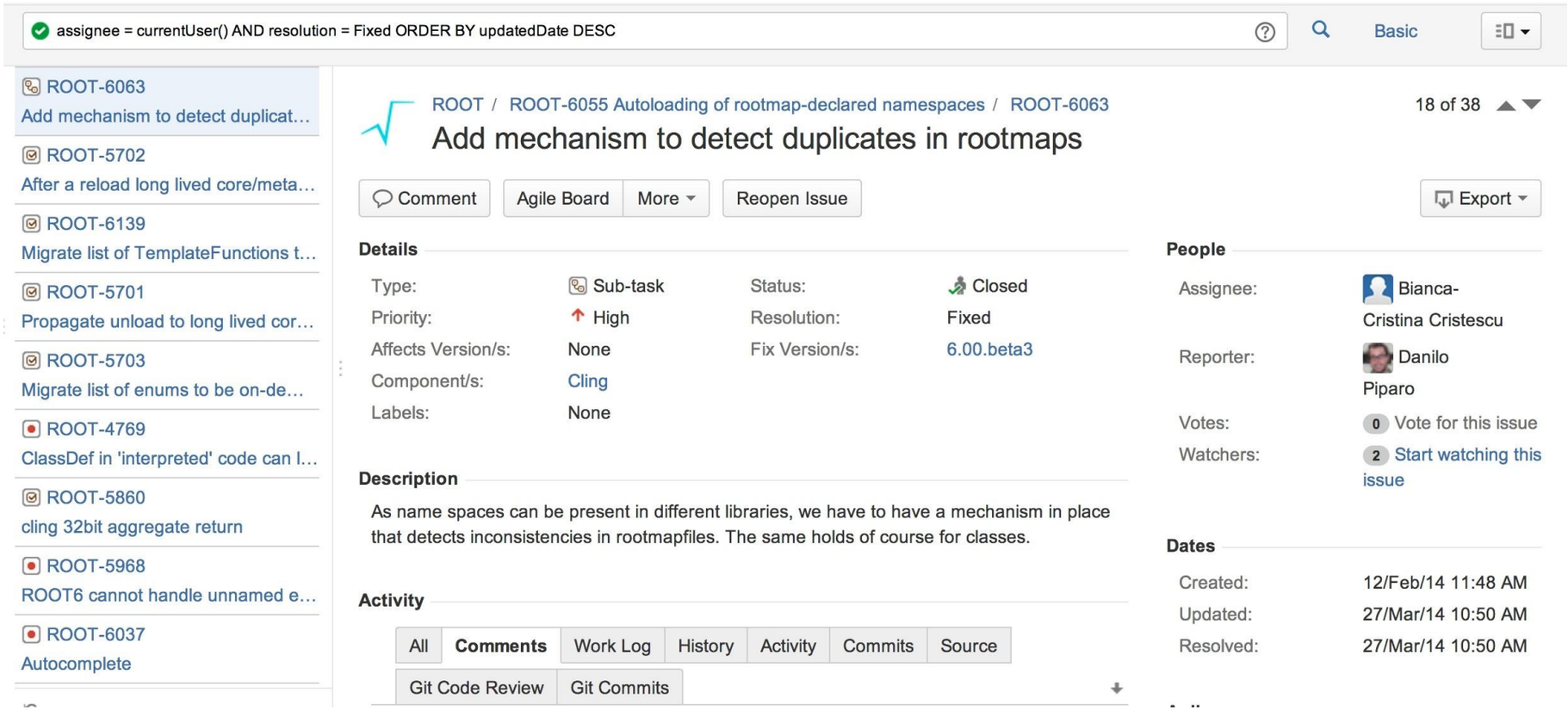


.....



- No stand-alone, big, fat task, but “on-demand” tasks

and the always pilling up Jira issues:



Bridges built

- **ROOT - Experiments bridge**
- **ROOT - Cling bridge**
- **ROOT - End User bridge**
- **PCM reproducers to Clang**

ROOT - Experiments bridge

- Implemented features available in Reflex, but not in ROOT 6 (enum reflection, attributes = reflex selection properties)
- Implemented features in the past provided by each experiment: GetMissingDictionaries
- Implemented features not there (enums-on-demand, TFunctionTemplate)
- Extended existing features: offset calculation

List of missing dictionaries

- Feature to allow knowledge about the missing dictionaries of a class to facilitate the generation of the missing ones easily

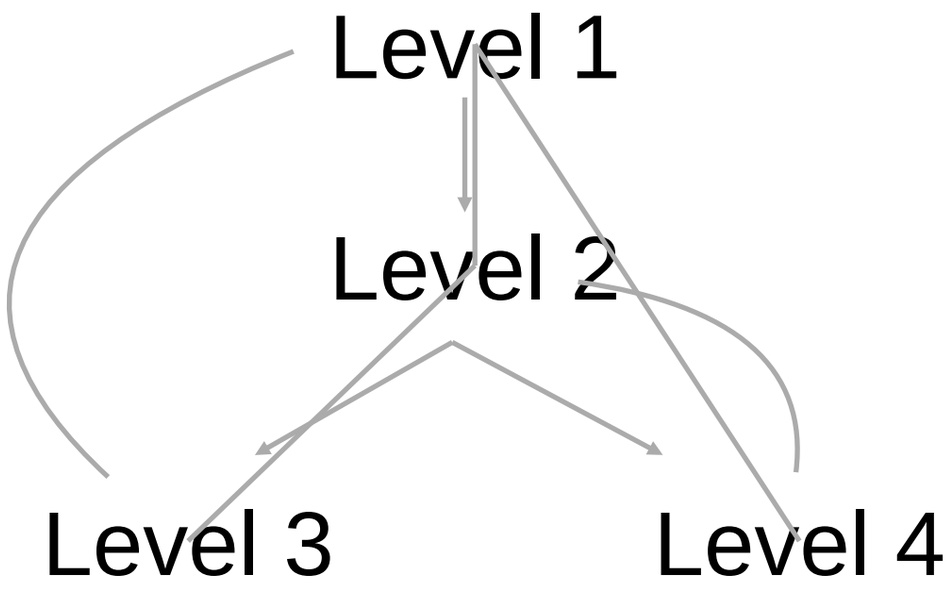
- Important feature for CMS

- Again at the middleware between TClass* objects and Clang we have to determine using clang::Decls which are the classes within our TClass that have missing dictionaries

- Fun complications with (Subst)TemplateParmTypeType

GetMissingDictionaries

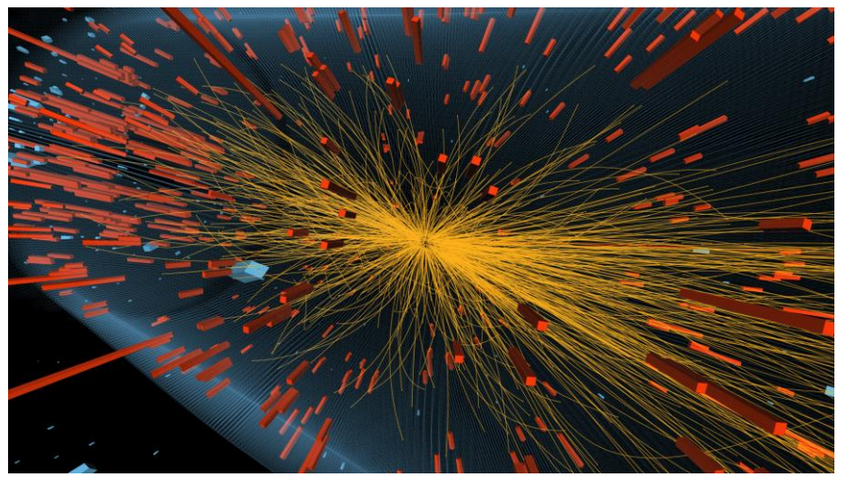
- Atypical recursion model with many different behaviour cases:



Virtual base offset

Multiple inheritance

Virtual base class



- Non-virtual multiple inheritance cases use Clang AST information and the multiple paths error cases are handled
- Virtual base cases: conversion compiled and executed at runtime
- Cast-to-derived versus Cast-to-base: long time bug in reflex

Virtual base offset

Getting the offset:

-> non-virtual base case

-> look up the offset in the cache

-> else if offset can be computed

-> store compiler calculated offset

-> virtual base case

-> look up the offset calculation *function* and execute it
-> else -> generate the *function* to calculate the offset

-> execute the wrapper and store the *function* for future calls.

Since the function has to be re-run for every object, caching the *function* is reducing significantly the cost

ROOT - Cling bridge

- Preprocessor Macro error recovery and printing
- Unloading on ROOT’s side
- Value Extraction Synthesiser

Preprocessor Macro error recovery and printing

- Important feature/requirement of the interpreter graceful recovery from errors

- MacroDeclQueue has been added in order to keep track of the preprocessor macros that have to be unrolled in case of error

- Contents need to be removed from caches and all the dependencies connected to it have to be updated as well

- To check the correctness of Cling’s state after recovery the structure of the preprocessor macro was made printable

Unloading on ROOT’s side

- ROOT's reflection data needs to be informed about unloaded objects
- ROOT's reflection data is persistent: unload means invalidate

```
root [0] TGlobal* g;
root [1] int i
(int) 0
root [2] g = (TGlobal*)gROOT->GetListOfGlobals()->FindObject("i")
(class TGlobal *) 0x7f8c7400da60
root [3] .undo 2
root [4] g->IsValid()
(Bool_t) false
```

ROOT - End User bridge

- Output redirection
- Tab completion
- Printing of const array chars

Output redirection

- Redirection was implemented by adding another command symbol to Cling's grammar .>

```
.>  /tmp/redirectoutput.txt
.2> /tmp/redirecterror.txt
.&> /tmp/bothfile.txt
```

- Having a RAI structure in the interpreter enabled the redirection command to support multiple levels of nesting

```
.> /tmp/redirect1.txt
.> /tmp/redirect2.txt
.>
.>
```

Tab Completion

- Pointless to explain what is tab completion and especially why we need it! (for the people in the room and remote)

```
root [0] gROOT->

AddClass
AddClassGenerator
Browse
Class
ClassSaved
Class_Name
Class_Version
CloseFiles
ConvertVersionCode2Int
ConvertVersionInt2Code
```

- Changed tab completion to the structure and features of ROOT 6

Value extraction synthesiser

e.g

root [0] **std**::**string** sarr[3] = {"A", "B", "C"}
(std::string [3]) { @0x7f92b8713b20 c_str: "A", @0x7f92b8713b38 c_str: "B",
@0x7f92b8713b50 c_str: "C" }

PCM bugs reproducers

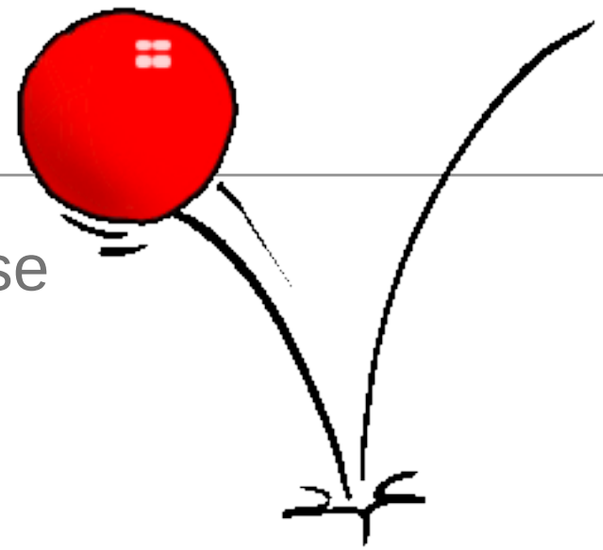
- In order to be able to build root with PCM modules, Clang standalone has to work
- Submitted 5 bugs to Clang, 1 got fixed already

Software development model

- Agile Development: requirements and solutions evolve through collaboration between self-organising, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement and encourages rapid and flexible response to change.

- Scrum: "a flexible, holistic product development strategy where a development team works as a unit to reach a common goal", challenges assumptions of the "traditional, sequential approach" to product development, and enables teams to self-organise by encouraging physical co-location or close online collaboration of all team members, as well as daily face-to-face communication among all team members and disciplines in the project.

Software development model



Although Scrum as part of Agile methods gets quite close to what we did, we had a characteristic way of working:

Our software development method:

- an extension to Scrum where we would have very often (3-4 times a week) Skype meetings and discussed our tasks, found ideas, buried ideas, built test cases, got unstuck. More importantly in this way I could get a informal review on my work
- Send a pull request to Axel's repository; that was the second level of reviewing my patches which enabled me to pick up the coding style and the conventions of the project more easily; this also enabled Philippe to review my patches too
- Like a bouncing ball the work was always turn on different sides and it got polished

Launching with ROOT6

- Usually students finish their projects and they leave before seeing the part that they worked on being used

- Lucky person to see my work being used e.g tab com

- ROOT6 launched and helping students to upgrade to ROOT6 and seeing posters being made using it

- As this was my first full time job my launching in the world as well..

Before

“Vague, but exciting!”

Mike Sendall

After

Clearer, still exciting!

Thank you!