CINT & Reflex: New Reflection Data Structures

Masa(haru) Goto • Agilent

Philippe Canal • Fermilab/CD

Stefan Roiser, Axel Naumann • PH/SFT

ROOT CERN

CINT Is The Prompt, Right?

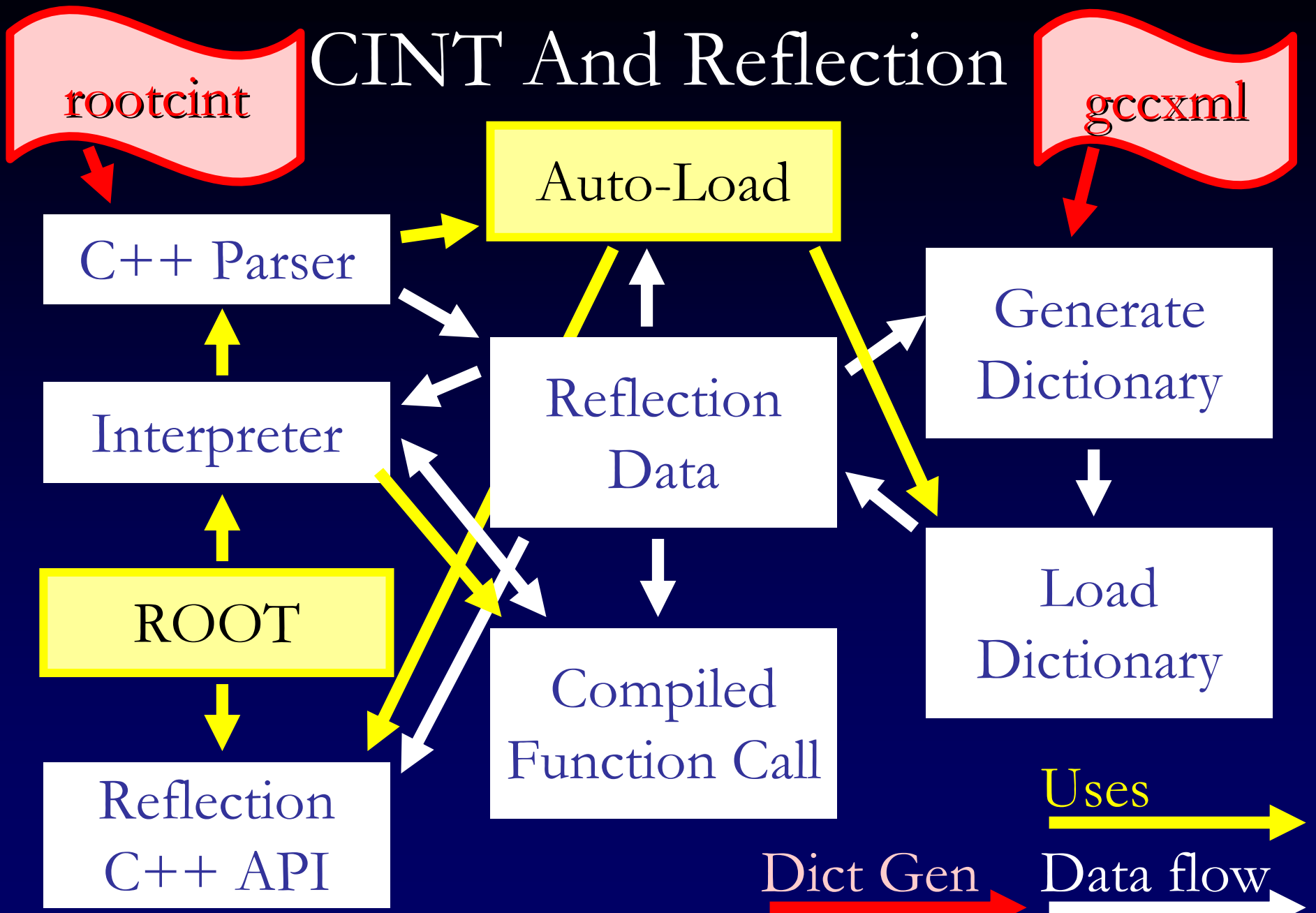
CINT is

- Backend for all dictionary based interpreters
- Execution engine (compiled and interpreted code)
- Prerequisite for signal/slot
- Prerequisite for I/O

Reflection is

- Part of CINT's knowledge, used for all of above
- Info on types and their members: name, type, access to values or call

CINT And Reflection



Comments From Last Review

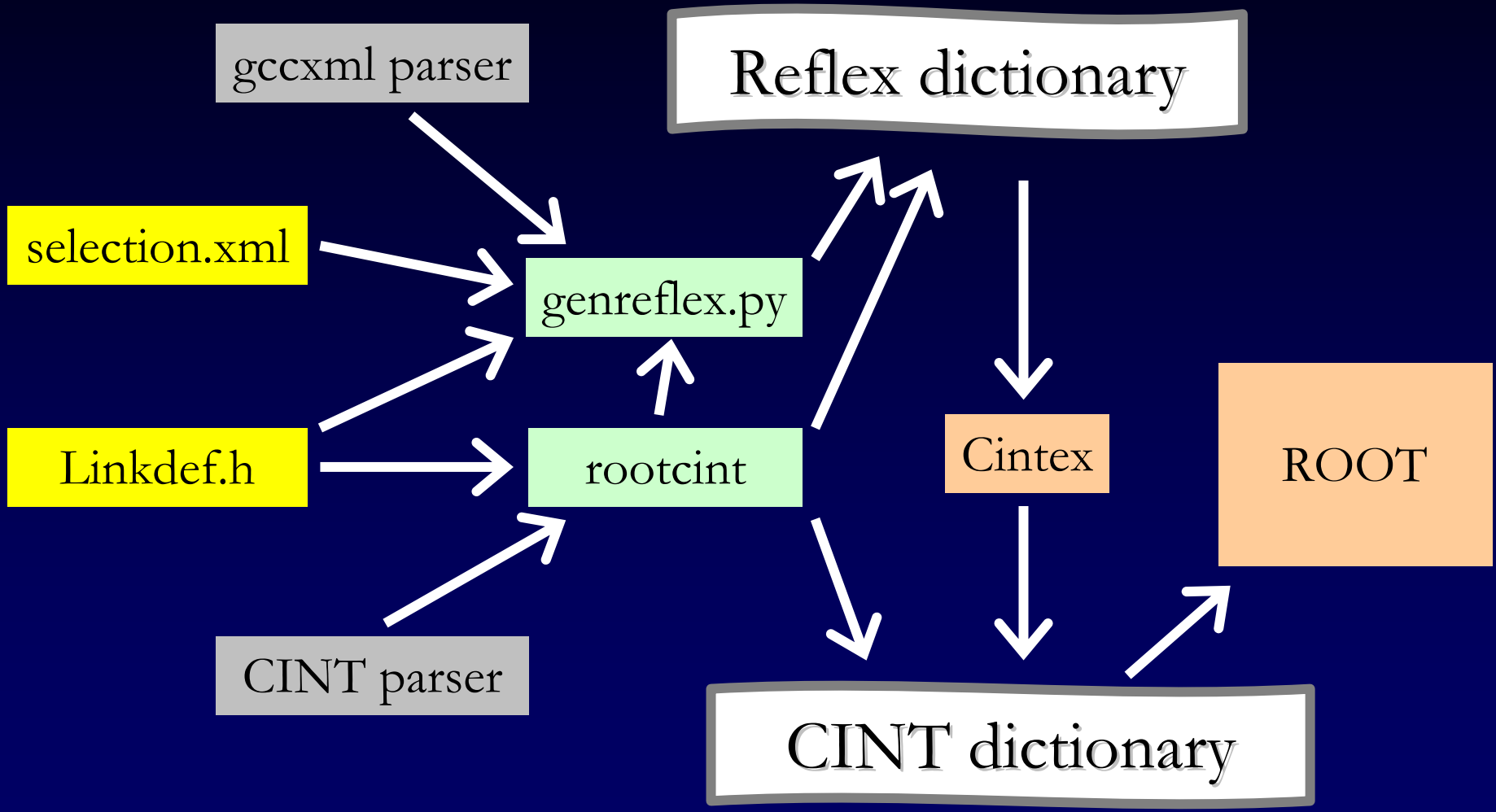
> *"Common Dictionary - unify dictionary clients for experiments using lcgdict/reflex, unify automatic generation of glue code (e.g. python bindings) [JJ Blaising, feedback talk 1.4.2005]*

Common Linkdef.h interface

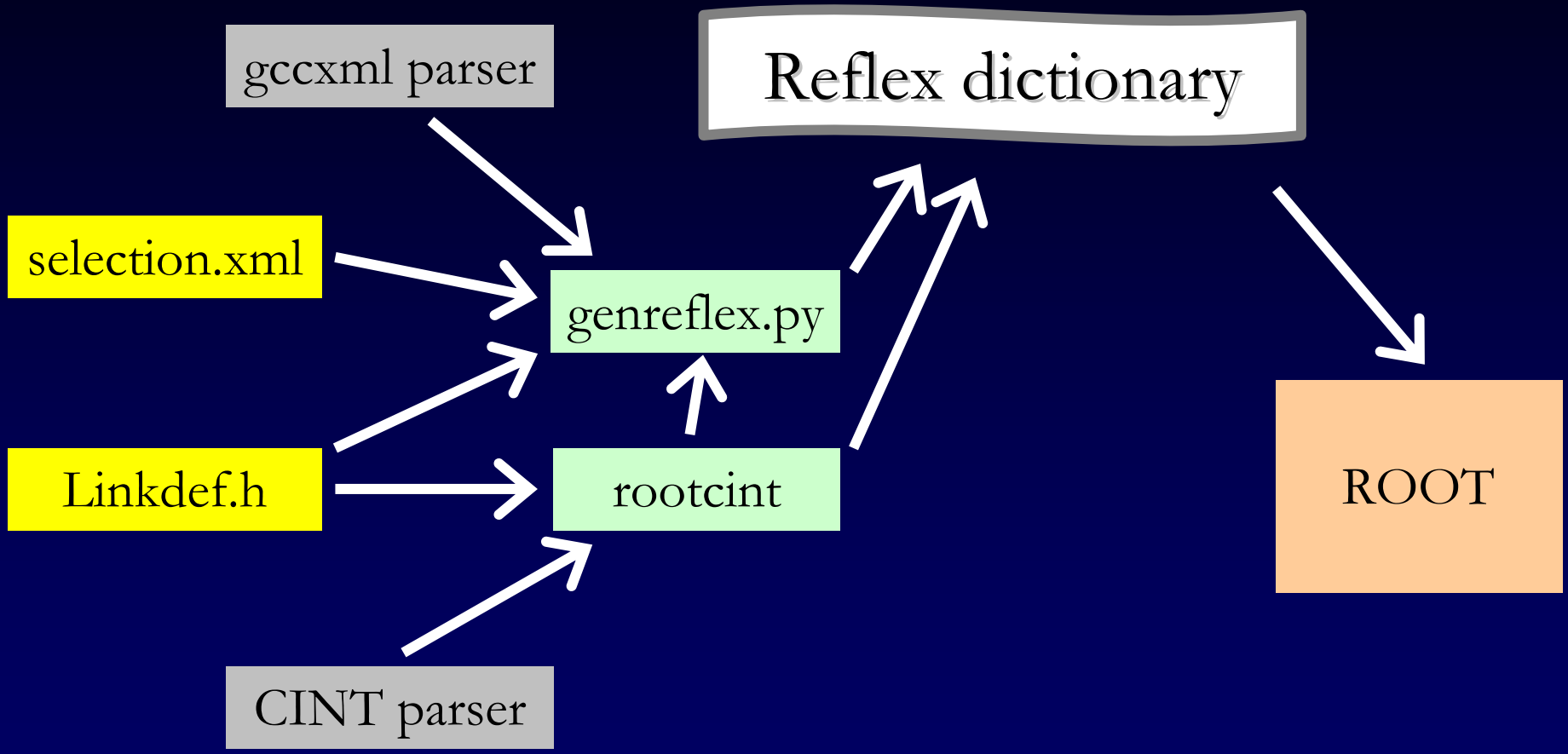
rootcint -cint, -reflex, -gccxml

Unified Python interface: PyROOT

Dictionary Options – You Choose!



Dictionary Options – The Goal



Comments From Last Review (cont'ed)

"*Ordering, scheduling, priorities [Final Report 2005]*

First version of Cint with Reflex reflection data base expected for end of year – not yet production grade!

Production-grade version available mid 2007

Reasons for delay:

- Moving functionality from CINT to Reflex
- Keeping CINT API backward compatible
- Merge process more involved than expected

That's It? Already?

No. This talk's menu offers following courses:

- Motivation: why we (still) need the merge
- Merge implementation details
- Steps of the merge
- Merge related development in Cint and Reflex
- Status
- Outlook

Motivation For Cint-Reflex Merge

C++ code replacing C structs

- Smaller heap footprint: factor 3 in current CVS for CLHEP based example – expected to deteriorate
- Dynamic data structures, no hard-wired limits
- Easier to maintain
- Refactoring, separation of functionality
- Preparation for thread-safety

C++ Code Vs. C structs Example

Current Cint has global C structs

```
struct G__tagtable {
   char* name[G__MAXSTRUCT];
   … } G__struct;
```

Reflex allows OO access to reflection data

```
Reflex::Scope::GlobalScope();
```

Covers types, data members, function members, templates,…

Merge Implementation - Context

Why you care: reason for merge schedule

- CINT mixes reflection and execution data, need to split carefully
- CINT extremely complex: extensive use of globals, spread of functionality, side effects
- CINT grown and debugged, don't want to risk breaking parts that work
- Changes need to be "invisible" to users

Merge Implementation - Consequence

- Small step changes
- Always compare to working version with extensive test suite
- Replace data structures first

Major benefit will come only later:

- replace functionality
- remove code duplication
- take advantage of OO data structures

Steps Of Merge

- Need duplication of scope info for both CINT and Reflex while merging (for contained types)
- Scaffolding, wrappers, convenience functions for Reflex and CINT
- Typedef database: **`G__newtype`** → **`Reflex::Type`**
- Globals: **`G__*tagnum`** → **`Reflex::Scope`**
- Structs: **`G__struct`** → **`Reflex::Scope`**
- Functions: **`G__ifunc`** → **`Reflex::FunctionMember`**
- Data members: **`G__vararry`** → **`Reflex::DataMember`**

Merge Tactics

- New **`configure; make`** CINT build system to allow sensible multi-platform tests
- CVS branch for merge, apply bug fixes from HEAD
- All other major development for CINT stalled until merge is done
- Update definition of CINT's API. Will keep API backward compatible, use export statements to ensure no internal function is used by e.g. ROOT
- Once done, make libCint.so and libCintNew.so binary compatible, to allow switch without rebuild: **`mv lib/libCintNew.so lib/libCint.so; root`**

Development In CINT For Merge

- New build system for standalone CINT/Reflex
- API redefinition
- Scaffolding to allow transition of data structures without large re-write of code
- **`extern "C"`** wrappers for some functions and classes

+ the actual merge…

Development In Reflex For Merge

- Unloading of dictionary
- Changes to Reflex::Builder interface
- Late (on-demand) initialization of reflection data
- Extended C++ support: using directives, hiding `typedef struct A A`, using declarations, friend
- Name Lookup: major boost in Reflex functionality, and damn complex

Development In Reflex (cont'ed)

- Fast access of reflex objects' properties, needed for CINT execution data
- Reflex objects can create dictionary representation of themselves
- Several convenience functions, e.g. discover the raw type (**`const char*`** → **`char`**), the kind of fundamental type etc.

Status - Where Are We Now?

! struct/class/…, function, data member reflection database
On demand initialization of Reflex data
Reflex support for using declarations, friend

To Do

Duplication of scopes for CINT and Reflex
Scaffolding, wrappers, convenience functionality
! `configure && make` build system
Update of definition of CINT API
! Typedef, globals reflection database
! Unloading of dictionary
Reflex::Builder interface changes
Reflex support for using directives, hiding `typedef struct A A`
Fast access to Reflex objects' properties

Done!

! Name Lookup
Reflex dictionary generator API

On the way

!: a lot of work

Outlook - Merge

Merge related:

- First version of CINT using Reflex by end of year
- ROOT's TClass, TMethod,… classes to be updated after Cint/Reflex merge, i.e. 2007
- Adaptation of PyROOT to use Reflex API
- Other ROOT occurrences of CINT API calls to be replaced by Reflex API calls

Outlook - CINT

2007:

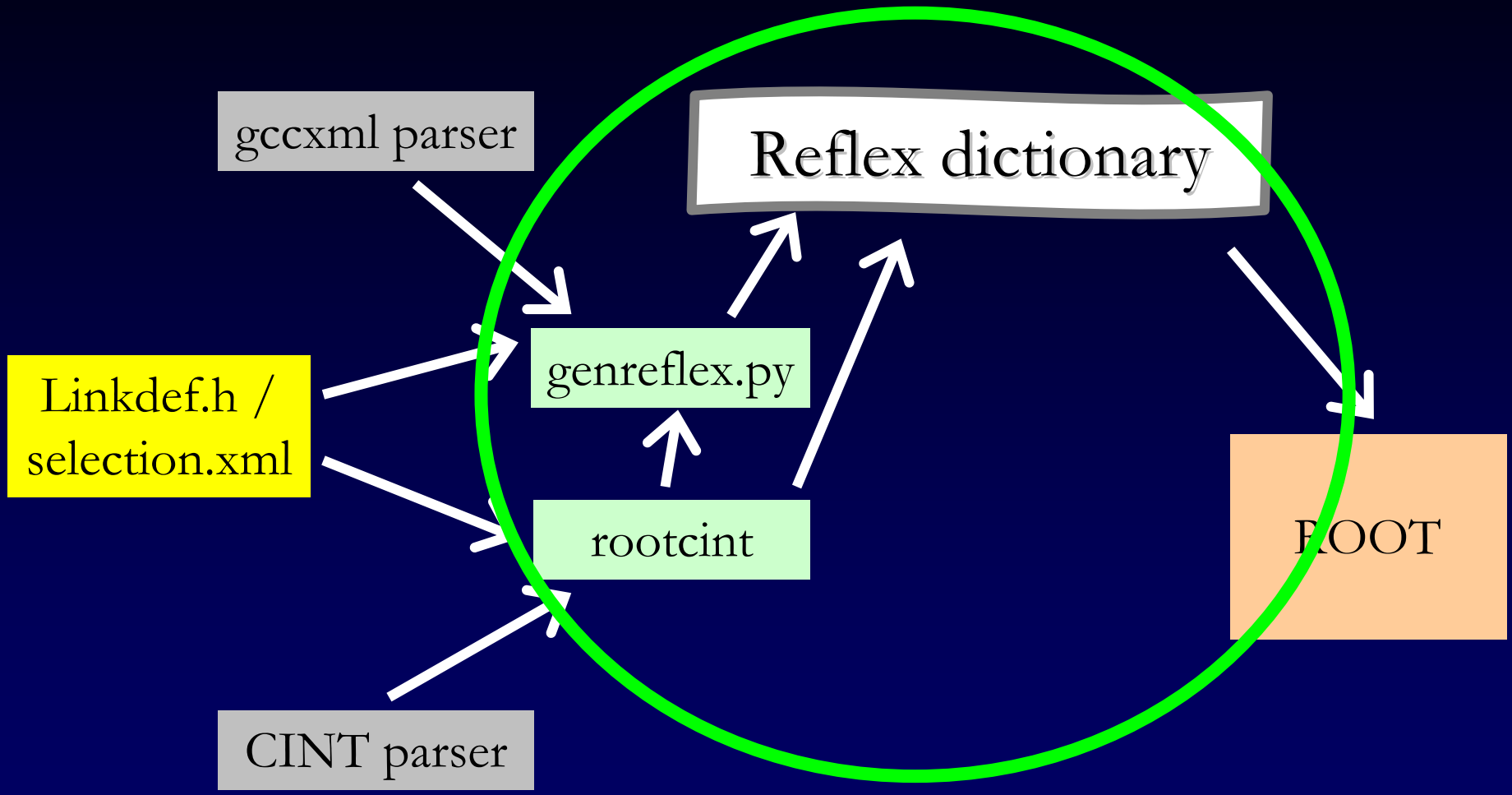
- Function stubs / call wrappers will be removed on selected platforms (GCC3,4; MSVC)
- Just-in-time dictionary generator builds dictionary for all template instances needed, when needed

2008:

- Reflex persistency - .root instead of .so

???

Outlook – Dictionary



Outlook – Dictionary

- Current Reflex dictionaries: **100%**
- 2007: Remove function stubs on selected platforms **33%**
- 2007: Generate template dictionaries on demand
- 2008: Don't "store" Reflex database as .cxx/.o (part of lib) but as .root file, generate member offset info on-the-fly: **– ~1%**

ROOT itself would shrink by ~25%!

Improvement independent of selection.xml / Linkdef.h and gendict.py / rootcint

Outlook

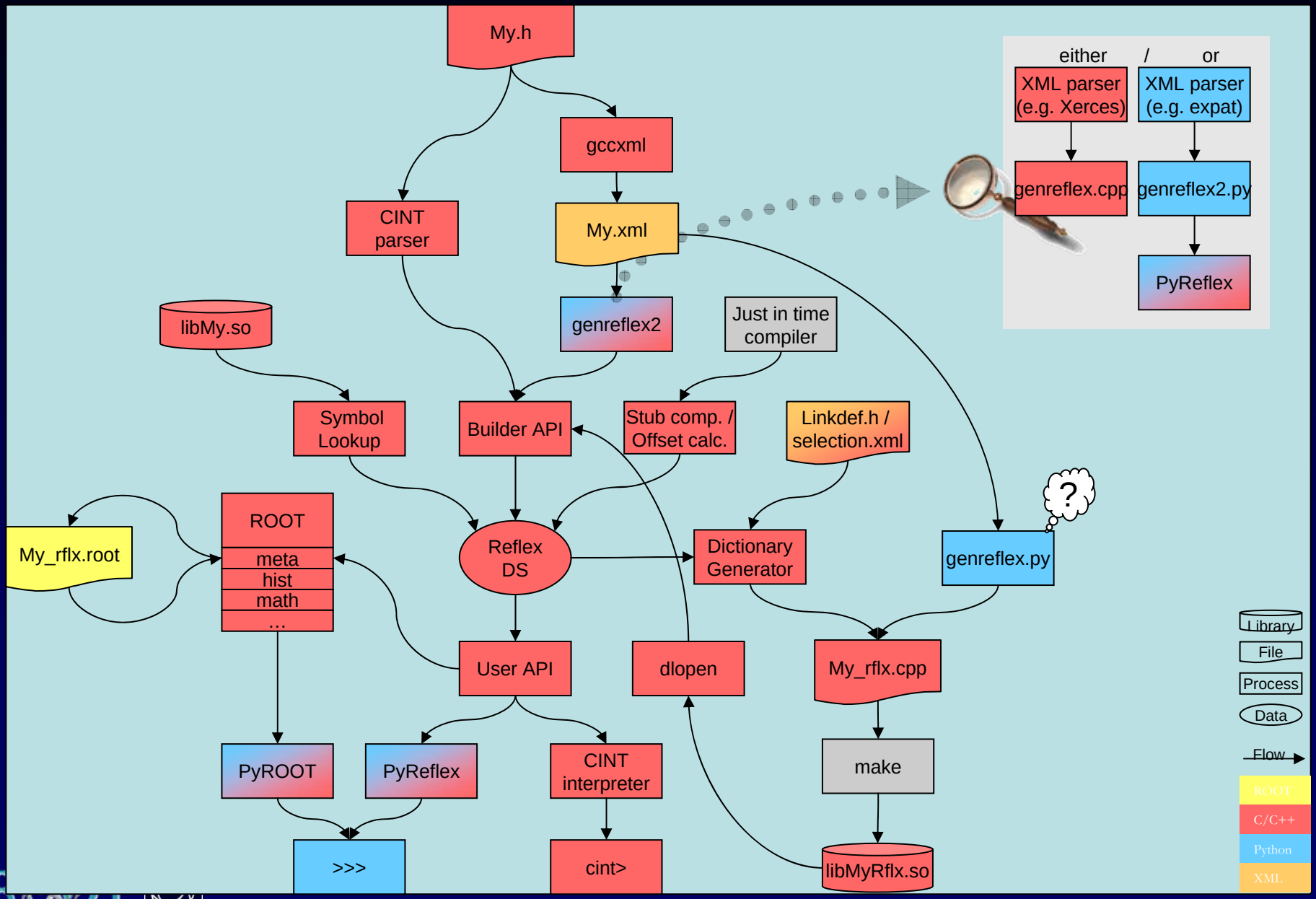
2008+:

- Future: reconsider gccxml to Reflex path: gendict.py? C++? PyROOT based?
- CINT thread safety

More details in the coming months – early comments welcome!

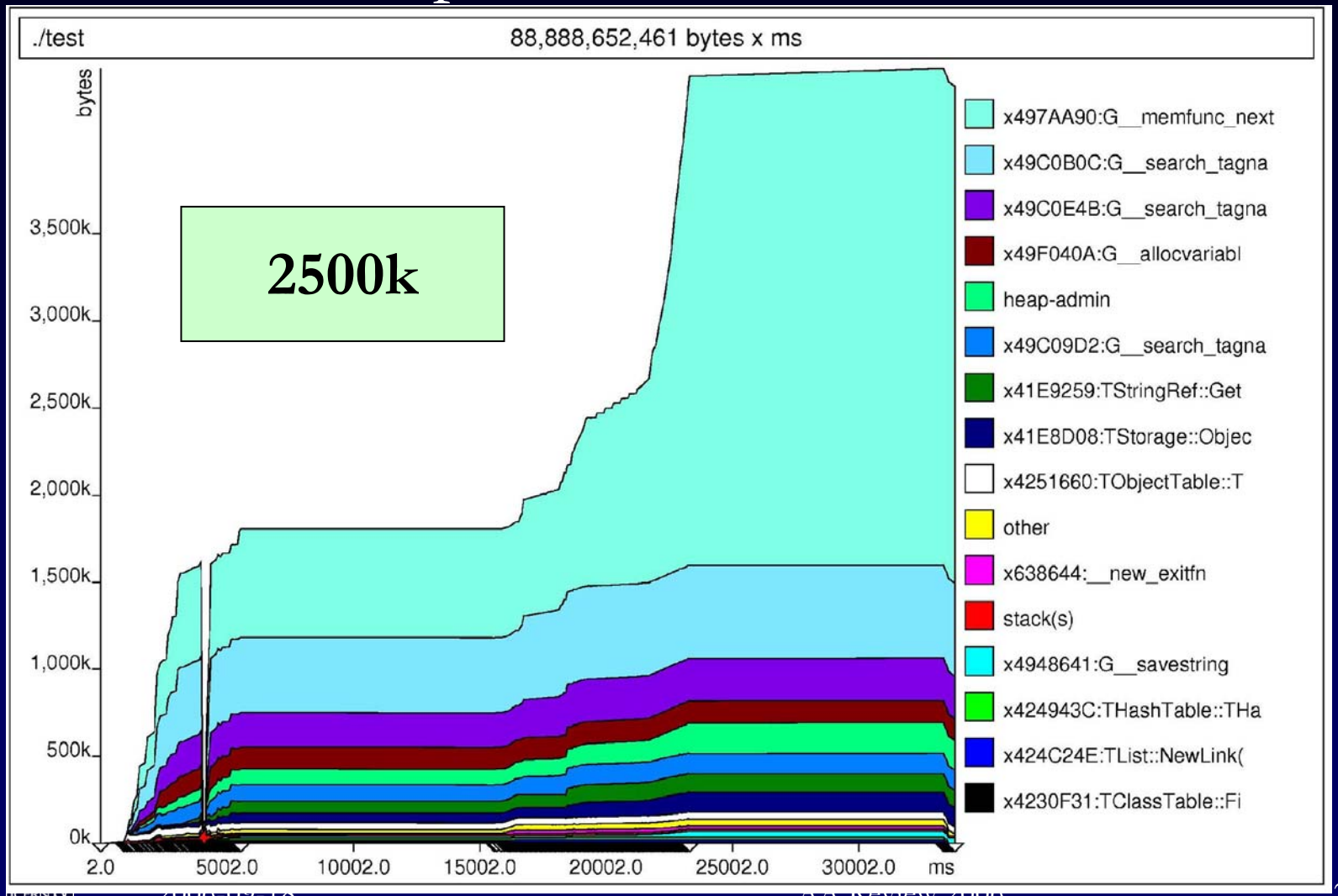
FIN

Dictionaries – The Full Picture 200x



Dictionary Heap Usage

CINT, CLHEP example



Dictionary Heap Usage

Reflex, CLHEP example