The CDB evolution - part 6

10/08 –Nikhef - Amsterdam

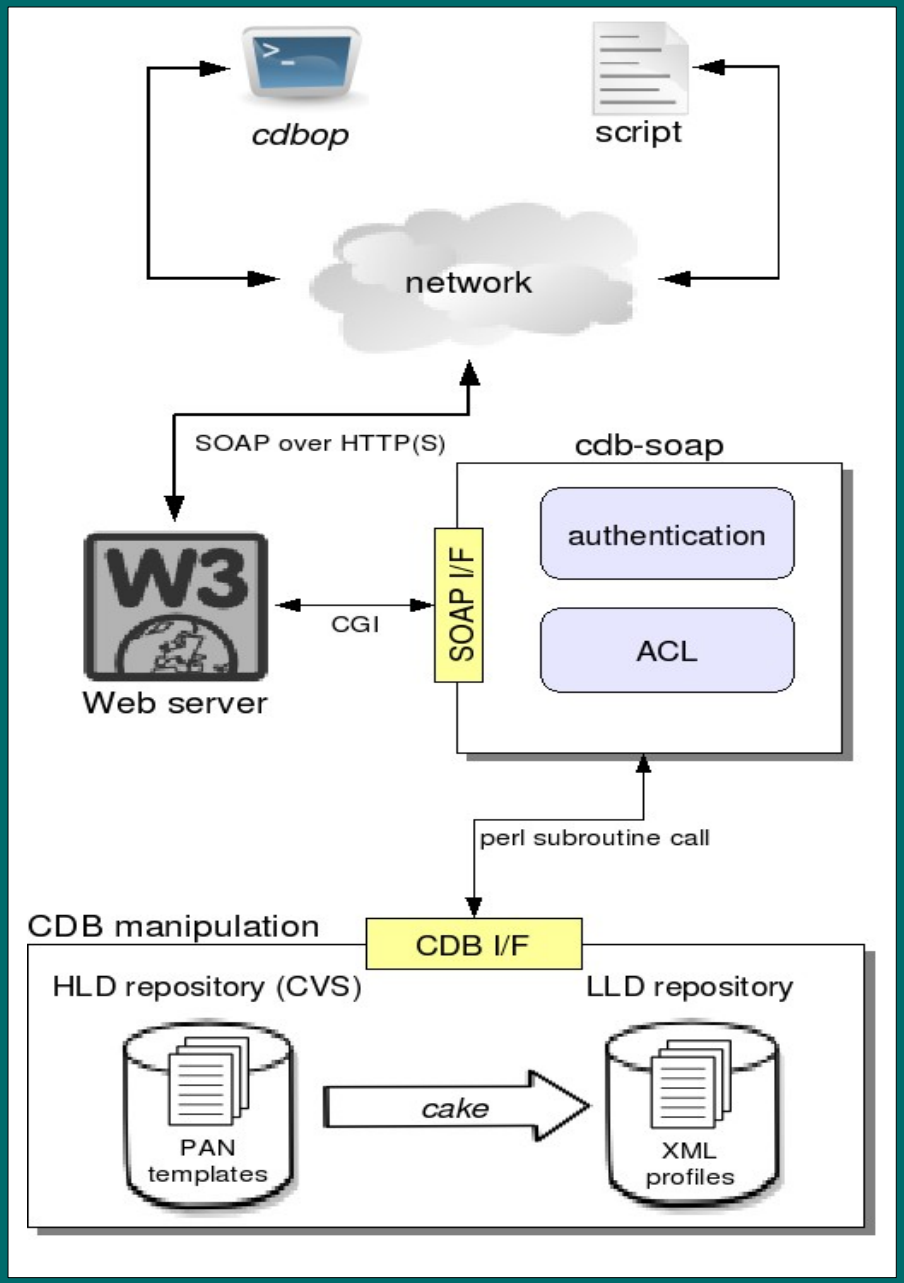
Marco Emilio Poleggi – *INFN-CNAF*

Marco.Poleggi@cnaf.infn.it

Outline

- CDB overview
- Tasks since last workshop @ CNAF
- What's new
- Wish list
- Boiling in the pot

CDB overview



Three-tier architecture

- SOAP client
  - cdbop interactive/batch shell
  - scripts
- SOAP middle-ware
  - Apache + *cdb-soap* CGI
  - stateless: each connection conveys one command
- CDB back-end
  - it's a library, not a server
  - templates compiled via cake
  - templates stored in *CVS*
  - stateful: partially transactional semantic through "sessions"

Tasks since last workshop @ UAM

- Completed
  - Visualization/navigation tool (not CDB-specific)
    - `cdb-tpl-view` and `pangraph` ready for **`panc-8`** (waiting for leaner logging options)
- Left behind
  - Fine-grained CDB locking with fair queuing. Long term
  - A common authentication service. Long term

What's new

- `cdb-cli (a.k.a. cdbop)`
  - Only bug fixes
- `cdb-soap`
  - Authentication module now supports locking
  - Faster state management
  - Unit tests
- `cdb`
  - Faster state management
  - Multi-threaded dependency calculation
  - Other optimizations

Wish list

- From last workshop, still pending:
  - Expose to clients some useful CVS features, like tagging, in a controlled manner [#9734, #17827]
  - Selective synchronization among multiple servers [#24687]
  - Suppress notifications for a given set of targets [#26433]
  - Allow to [#20280]:
    - query the CVS backend
    - log real user name and comment on commit
- Add ACLs/restrictions on the CDB configuration tree for "include" [#24983]: what's the status of the discussion about extending Panc?

Boiling in the pot

Ideas

- Waiting for leaner `panc-8`'s logging options to allow a feasible template visualization service.

Issues

- Manpower
  - I can't dedicate any longer much time to development :-(
- Still problems with `panc-8`. I'm investigating...
- Scalability of dependency calculation:
  - Currently all profiles are scanned: complexity is `O(#profiles)*O(#modfied templates)*O(#avg dependencies)`, e.g. at CERN ~100M operations for 10 templates touched.
  - Reverse dependency calculation + caching might be a solution.


quattor
http://quattor.org/