Π Project
Status and plan

AA Internal Review, 16 October 2003

Preliminary
Final Version on Sunday

Vincenzo Innocente

CERN/EP

Project Overview

- Physicist Interface (PI) started in mid Nov`02
  - **Review with experiments to define workplan**
  - Project proposal to SC2 end Jan`03, reviewed on July`03
- Five working areas identified
  - Analysis Services
  - Analysis Environment
  - Pool & Grid PI
  - Event & Detector Visualization
  - Infrastructures & Documentation
- Not all items have same priority:
  - Analysis Services first
- **Resources:**
  - V.I. 30%, Andreas Pfeiffer (40%), Lorenzo Moneta (50%)
  - Visitors on rotational basis (Hurng-Chun Lee for 3 months)

Event Display and Detector Visualization

- WP 4 – Event Display and Detector Visualization
  - HepVis: review, adjust, extend (to cover LCG and Geant4 needs)
  - Integrate into interactive analysis environment
  - Geant4 visualization application

  - In collaboration with the experiments (aim for common product at a later stage)
    - At the very end of the food-chain (no other product depends on it)
    - Very specialised field: few developers often in remote institutes
      - Technology and implementation details seem not to be “details”
      - Not much interest in a wide collaboration or a common product

Frozen indefinetively

POOL & GRID Interfaces

- WP 3 – POOL & GRID Interfaces
  - Collections & Metadata
  - File Catalog and Replicas (both local and remote)
  - Job Wizard (preparation, submission, validation)
  - Job Helpers (monitoring, error recovery, resource discovery)

In the Requirements & Analysis Phase (RTAG 11: ARDA)

Analysis Environment

- WP 2 – Analysis Environment
  - Basic interactive analysis application:
    - based on SEAL python binding, plugin manager, distributed interfaces
  - Visualization services: interactive canvas, Model-Document-View
    - Connected to WP4 (frozen)
  - Bridge to and from ROOT: interoperability at cint prompt, etc.

  - in collaboration with SEAL, POOL
  - Sharing of responsibility among projects not obvious
    - Infrastructure seems to be in hand of SEAL
    - Concrete implementations in the hand of the developers of the corresponding C++ product
    - PI may have a little role to play here besides taking care of its own product: analysis services

Analysis Services

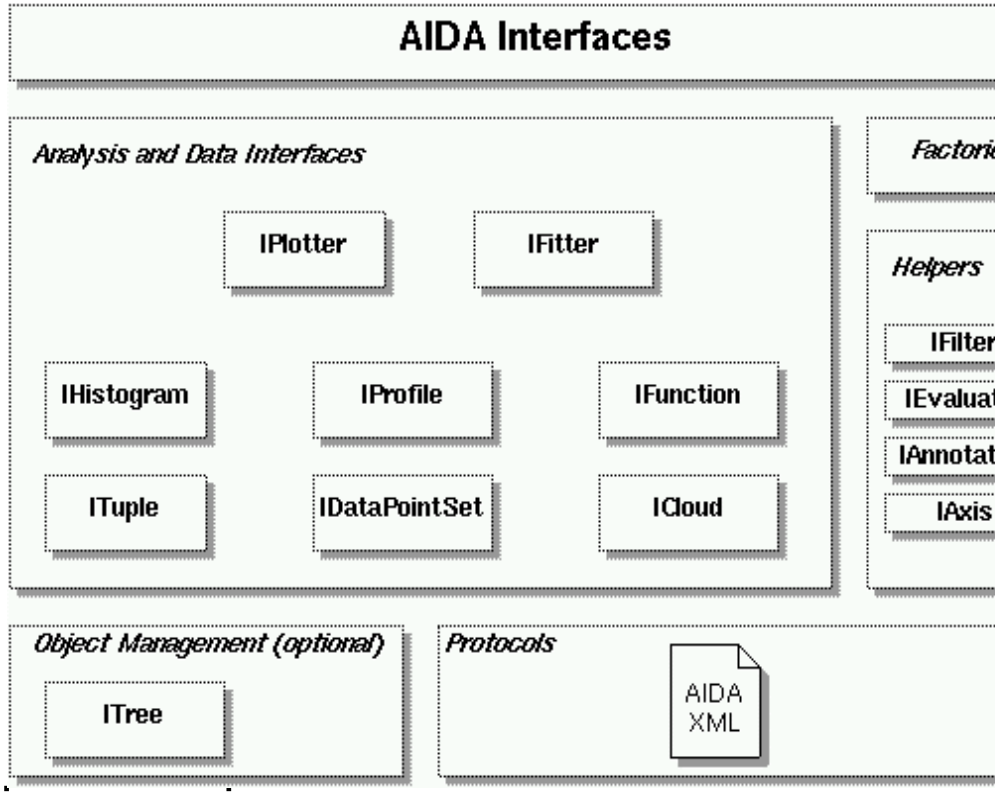
- WP 1 – Analysis Services
  - AIDA interfaces: revision and evaluation
  - Implementation:
    - existing: C++ and Java
    - a new one: ROOT-based.
  - New “AIDA” developer level interfaces for SEAL and POOL components: whiteboard, collections... (*see comments on WP2*)
  - Blueprint compliant analysis toolset
    - Statistics analysis tools based on AIDA interface
    - Mainly external contributions
  - first release 0.1.0, May 2003
  - 0.3.0, July 2003: Full implementation of Root back-end
  - latest update: 1.0.0 (last week)

Work plan approved by SC2 essentially completed

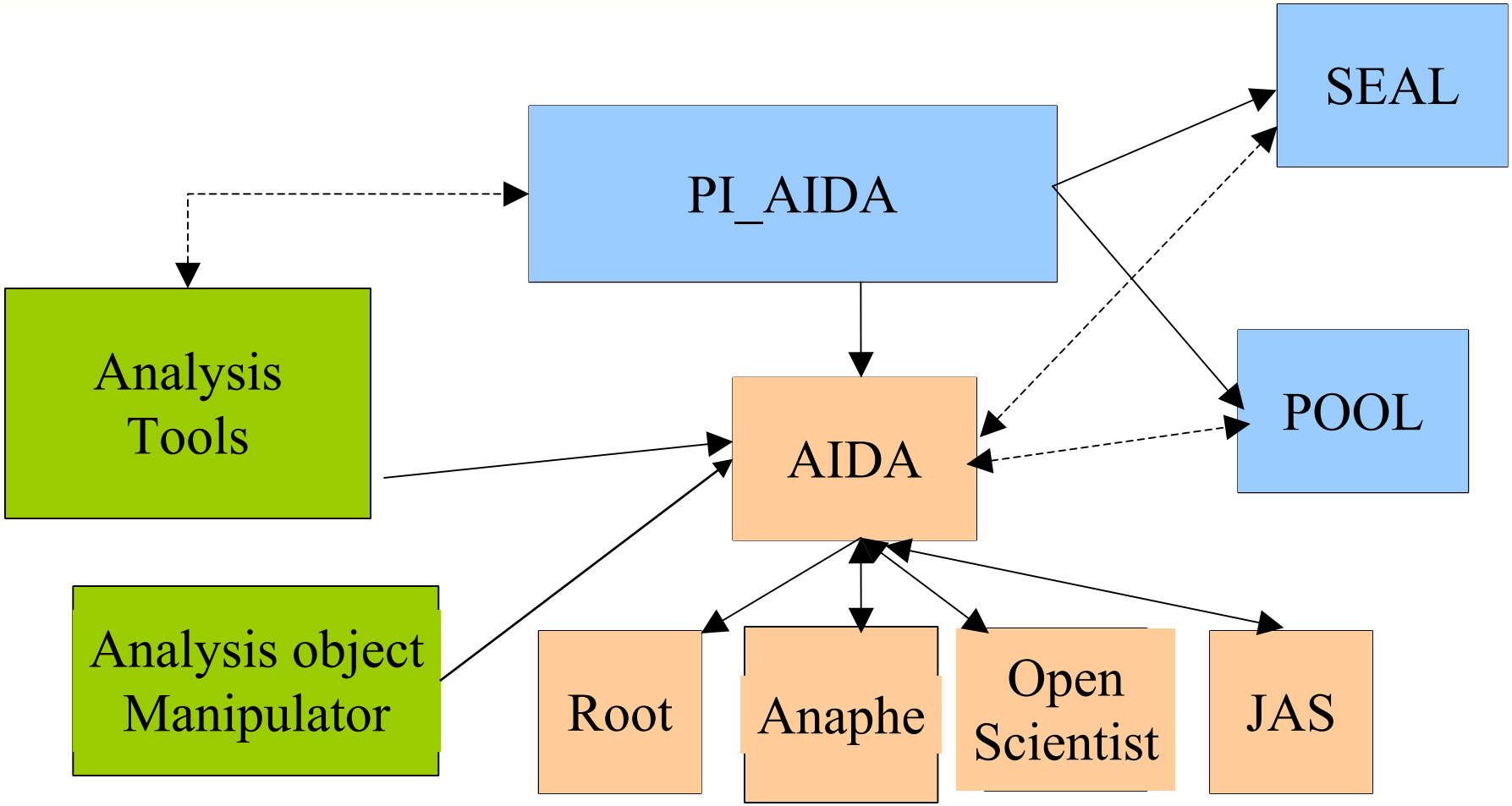
AIDA

AIDA – Abstract Interfaces for Data Analysis

- Started in late 1999 (HepVis Workshop)
- Open collaboration, teams from
  - PI (CERN)
  - JAS (SLAC)
  - OpenScientist (LAL)
- Version 3.0 since Oct. 2002
  - User level interfaces
  - Pointers to objects with factories
  - XML protocol for data exchange
- Missing
  - Simplified value-semantic layer with constructors and operators
  - Developer interface to ease building generic manipulators and tools



PI Analysis Services



The Goals

- **Interoperability**
  - Project Root 2D histogram on Anaphe 1d histogram
  - Use Anaphe fitter in OpenScientist
  - Exchange histograms between ROOT (C++) and Jas (Java)
- **Extensions**
  - Generic manipulator such as projectors, rebinners, etc
  - Build and manipulate aggregate of objects (CANs)
- **Interface to external applications**
  - Store AIDA histograms in POOL (connected to experiment specific data)
  - Display AIDA histograms using external tools such as HippoDraw or EXCEL
- **Framework to develop complex analysis tools**
  - Statistical comparison of data-sets
  - Modelling parametric fit problems using a MonteCarlo approach

Milestone 1: AIDA Proxy layer

- "Value semantics" for AIDA objects
  - Implemented using the "Proxy" pattern, very easy !
  - Based only on AIDA Interfaces
    ➔***no dependency on a given implementation***
  - Initially "hiding" of AIDA object management, later: use of SEAL whiteboard
- Keeping the functionality and signatures of AIDA
  - "re-shuffling" of factory methods to object ctors
  - Use of SEAL plugin mechanism to select implementation
- Examples on how to use with web-docs
- It Has been the base for the user-review and further evaluation
  - Any feedback will be propagated to AIDA team

User Review of AIDA

- Enhance AIDA
  - **ErrorPropagation Functor**
    - to allow correct (and/or user specified) treatment of error propagation in profile histos and DataPointSets
  - **Notification of object modifications**
  - **Bin iterators**
  - **New “bindings” (SQL…)**
- Review of AIDA (Proxies) by users in the experiments
  - **First feedback from CMS and LHCB**
  - **Validity of the AIDA approach confirmed**
  - **Interest in new developments**
    - **HippoDraw of Clouds was found particularly impressive**
  - **Request of new features connected to online applications**
- Integration with Root
  - **direct access to AIDA object from root is still missing**
  - **Looser binding (pyRoot) can satisfy only trivial use-cases**

Latest Release: 1.0.0

- Dynamic loading of implementation:
  - **AnalysisServices/AIDA_Proxy is a proxy of the AIDA interfaces. It uses now the SEAL PluginManager to choose at runtime the corresponding implementation of the AIDA interfaces.**
- ROOT implementation and ROOT I/O
  - **AnalysisServices/AIDA_Root provides an AIDA implementation for Roo Histograms. Available for all types of binned Histograms and Profiles.**
- XML I/O
  - **AnalysisServices/AIDA_Proxy implements Proxy_Store, which gives the user the possibility to read/write AIDA objects (all types of Histograms, plus DataPointSets and Tuples) in a compressed XML file following a format specified by AIDA. (AIDA XML format)**
- Test Framework
- Configuration
  - **based on SEAL_1_2_0.**

AIDA testing Framework

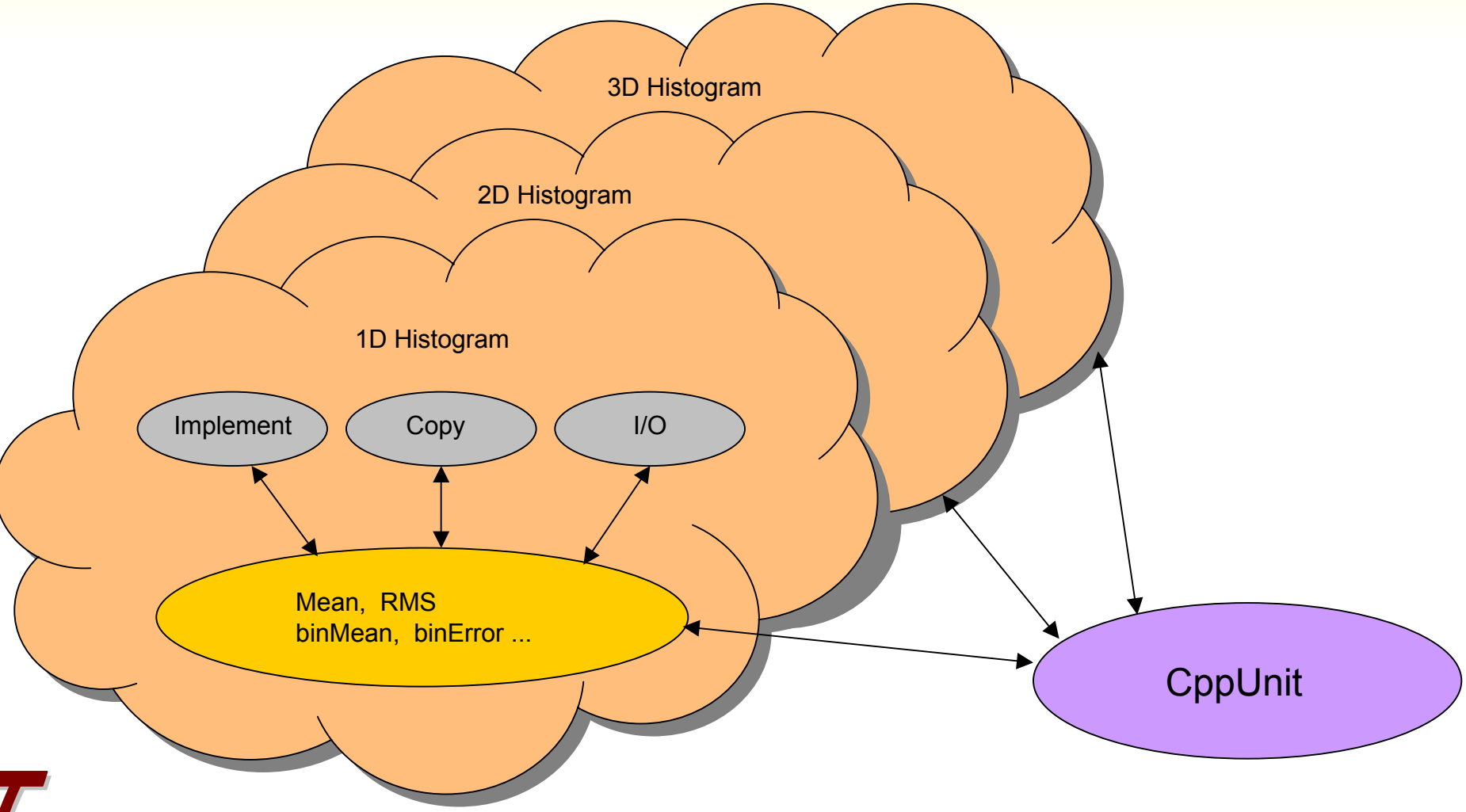
Hurng-Chun Lee (Academia Sinica Computing Centre, Taiwan

- **Purpose**
  - Check the functionalities of AIDA Proxy
  - Check the consistencies between different histogram implementations through AIDA Proxy
  - Develop a unit test framework for AIDA Proxy on which test-cases can be easily adapted

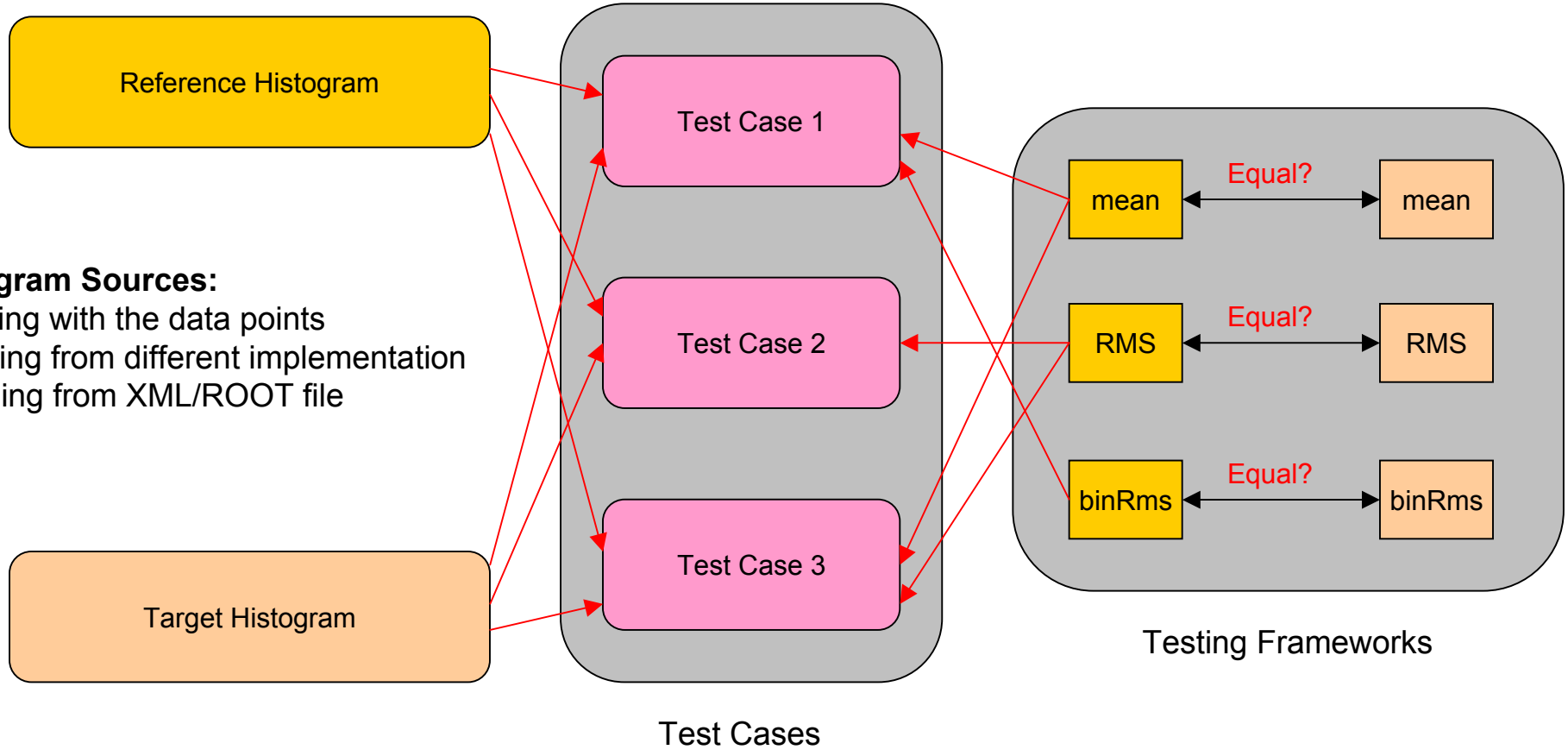
- **First version released in PI 1.0.0**
  - 25% of tests fails
  - Some due to outstanding bugs (to be corrected in next release)
  - Others under investigation
    - difference of behavior between implementations in using under/overflows in globa statistics

Hurng-Chun will give a full report on November 12 AAM

Unit tests of AIDA_Proxy



Testing Logic



Histogram Sources:

- Feeding with the data points
- Copying from different implementation
- Reading from XML/ROOT file

Possible further work-items

- Test integration with other frameworks/implementations
  - External visualisation tools: hippoDraw, exel?
  - Experimental frameworks: started with CMS, LHCb
  - fitting: use minuit C++ from SEAL
- Interoperability via “developer level” Interfaces
  - SEALs object plugin manager, whiteboard
  - POOL persistency and collections
- New functionalities
  - Container to hold various “related” AIDA objects
    - Histo(s) for data, Histo(s) for MC, Fit(s) to either ...
  - Sliding windows clouds

Closer look to real-life use-case first

User involvement essential!

Future?

- **WP1: The program approved by SC2 is essentially completed**
  - User feedback suggests further developments
    - Some in areas explicitly censored by SC2 and EP management
- **WP2: Support of interactive services seem to be in the mandate of SEAL**
- **WP3: Distributed Analysis still in RTAG phase**
- **WP4: No interest in common project in Visualization**

- **PI needs more clear recommendations on how to proceed**
  - It is my personal opinion that anticipation of user-needs is essential in the development of any product, particularly if at the end of the food-chain.
    - Excluding it from the mandate of LCG removes motivation from developers and makes timely deployment impossible
    - Prototyping inside an experiment may help in focusing on real use cases, in easing integration and in avoiding anarchic development

Summary

- Development of high priority items, such as analysis services based on AIDA, on schedule
  - First release in May, production version 1.0.0 last week
  - Steady progress, compatible with other priorities of the developers
- Review of AIDA completed
  - **Goal: provide a fully consistent interface and a set of low level tools that satisfy the requirements of both end-users and developers of high-level tools**
- ARDA RTAG almost completed
  - **Covers also the interface to GRID and data-storage services**
  - **PI eventual involvement unclear**
- Lower priority items (such as Visualization) waiting for input by the experiments