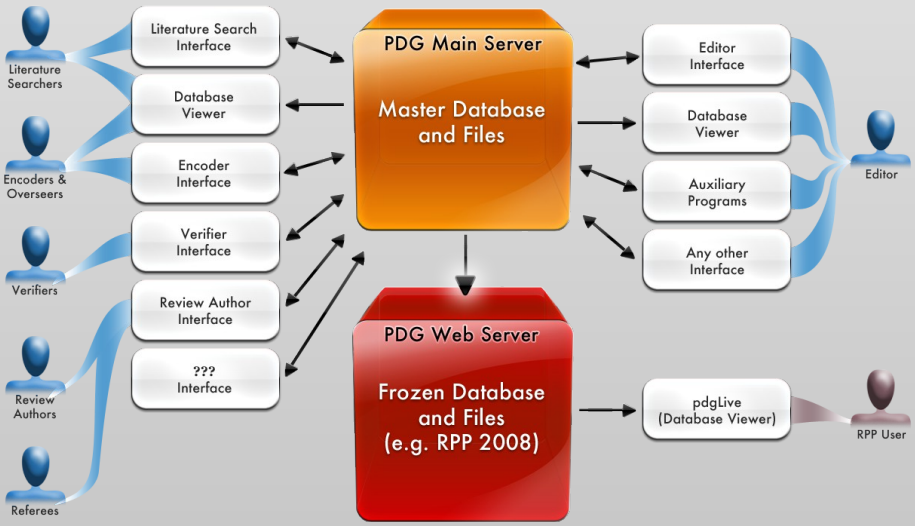
Status of the New PDG Computing System

Juerg Beringer
Physics Division
Lawrence Berkeley National Laboratory



Outline:

- Project status
- **Major success: V0 Release**
- Feedback from user testing
- Roadmap towards completion
- Innovative new features

Two Years Ago

- **Had submitted plan for completing computing upgrade to DOE**

Written in 2006

High-Level Requirements and Roadmap for PDG Computing

Juerg Beringer
Particle Data Group
Lawrence Berkeley National Laboratory

This document summarizes the high-level requirements for the upgraded PDG computing system and proposes a roadmap for completing the upgrade. It is intended to serve as a starting point for a cost estimate for the completion of the upgrade project.

- **Comprehensive and very successful DOE review of PDG in September 2008**
  - **Vital role of PDG reaffirmed**
    - "The PDG publications are crucial to the field ..." (DOE reviewer)
  - **DOE agreed to plan and asked us to increase request for resources for computing upgrade to ensure we will succeed**
    - Now 2 FTE for 3 years (until end of FY11)
    - 0.5 FTE for ongoing support after initial development
- **NSF agreed to contribute to computing upgrade according to its overall share of PDG funding**

Development work funded and in full swing by end of 2008

Today

- **About two-thirds of the project completed**
- **Successfully deployed initial version (V0 Release) of new system on August 11, 2010**
  - Now our production system for ongoing PDG work
- **Full-day DOE review of Computing Upgrade on September 17**
  - http://pdg.lbl.gov/computingreview2010/index.html

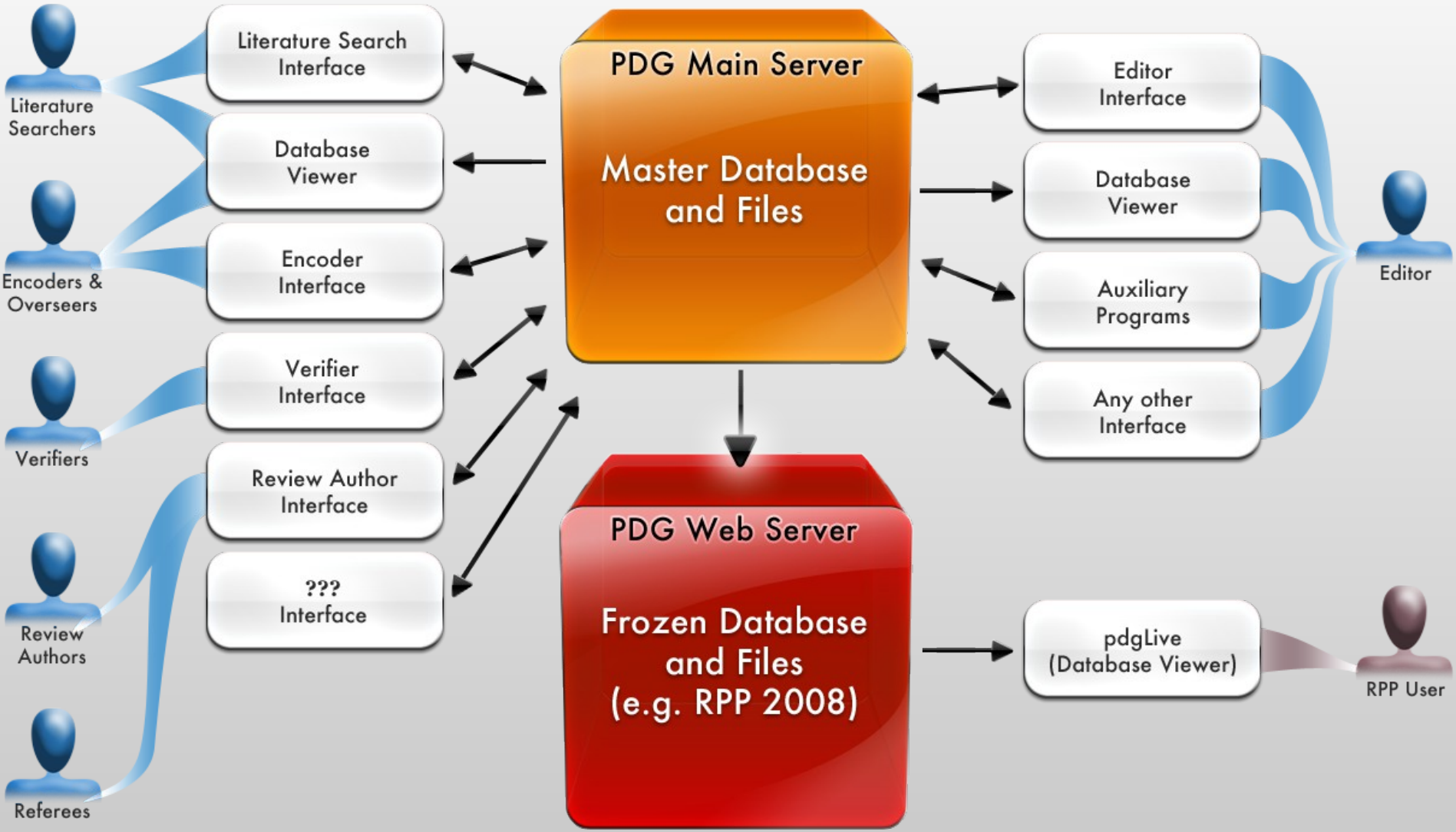
Quotes from the review report (October 20, 2010):

- **Computer system upgrade is proceeding on schedule and within cost**
- **Satisfying all the requirements laid out for it in the 2008 review**
- **Upgraded system will clearly make a dramatic improvement in the PDG production workflow**

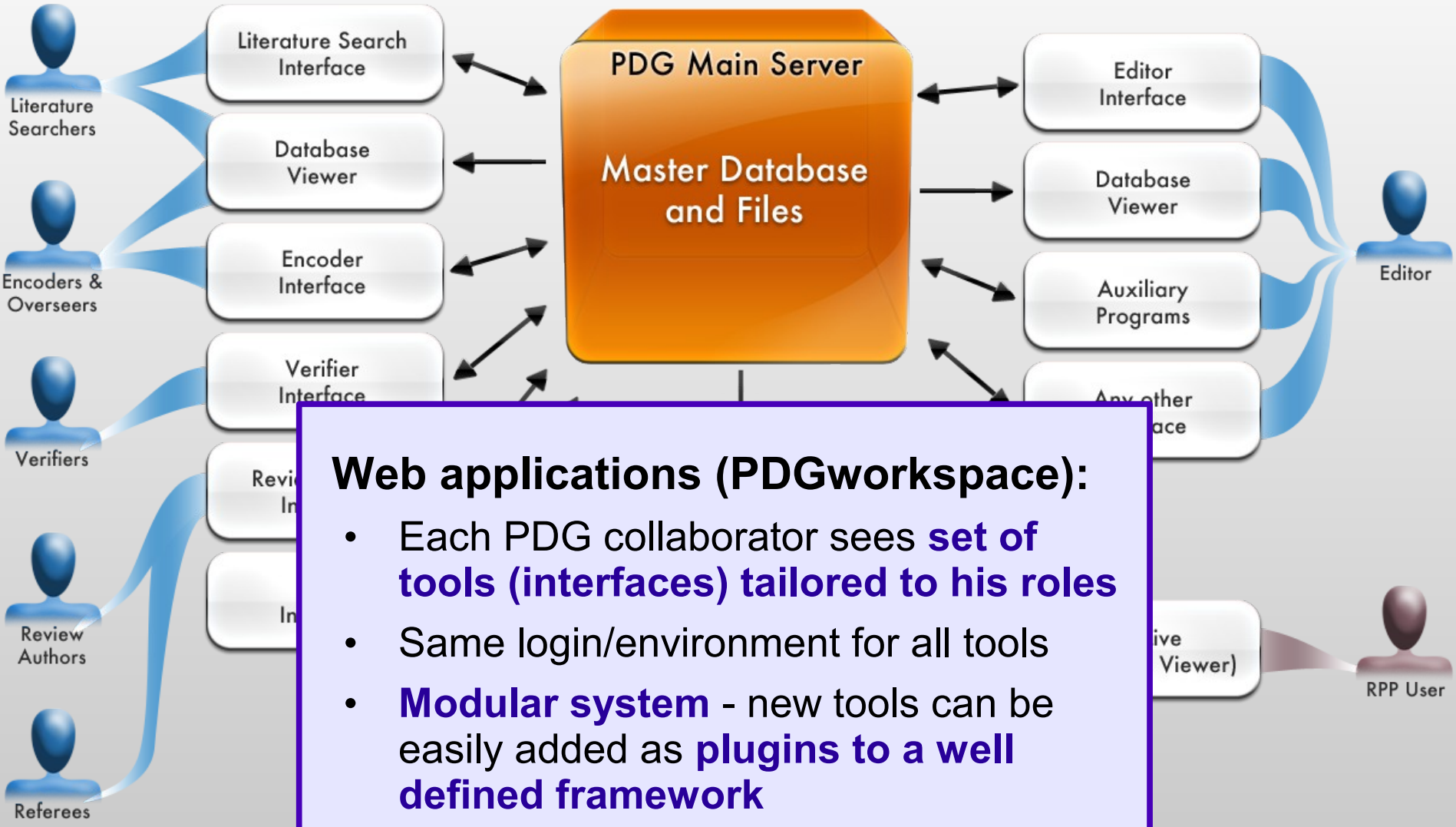
Goals and Scope

- **Primary goal: ensure that PDG can continue to function well**
  - System must be modern, modular, extendable, easy-to-use, maintainable and well-documented
- **Computing system must support all areas of our work**
  - **Decentralized, web-based data entry** and verification for Listings
  - Tools for authoring and refereeing **reviews**
  - **Monitoring** of progress in RPP production
  - Programs for evaluation of data (**fits, averages, plots**, …)
  - **Expert tools for editor**, including creation of book manuscript and static web pages (PDF files)
  - Interactive browsing of PDG database similar to **pdgLive**
- **Suitable platform for future extensions**

New System


Literature Searchers
Encoders & Overseers
Verifiers
Review Authors
Referees
Literature Search Interface
Database Viewer
Encoder Interface
Verifier Interface
Review Author Interface
??? Interface
PDG Main Server
Master Database and Files
PDG Web Server
Frozen Database and Files (e.g. RPP 2008)
Editor Interface
Database Viewer
Auxiliary Programs
Any other Interface
Editor
pdgLive (Database Viewer)
RPP User

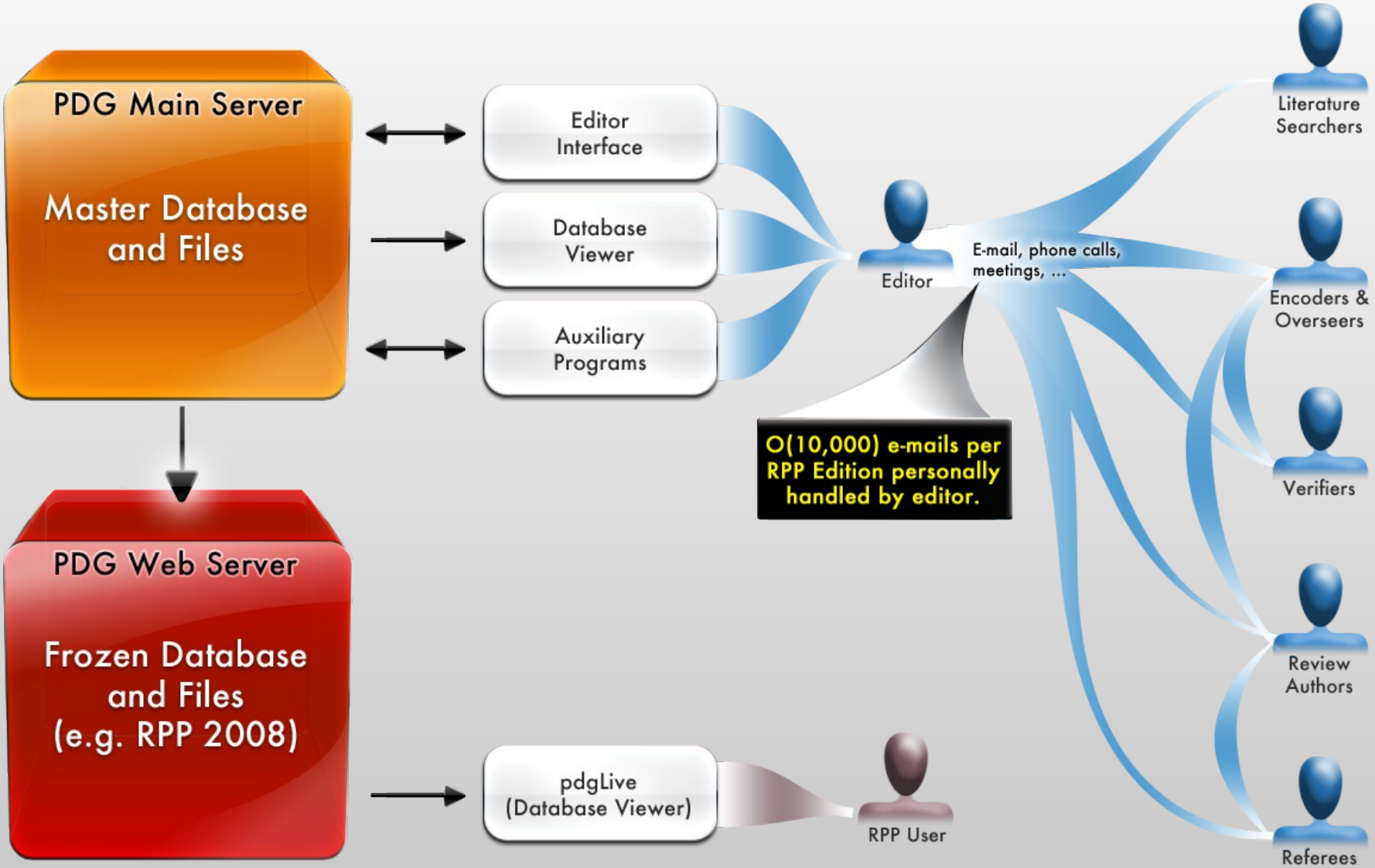
New System



Web applications (PDGworkspace):

- Each PDG collaborator sees **set of tools (interfaces) tailored to his roles**
- Same login/environment for all tools
- **Modular system** - new tools can be easily added as **plugins to a well defined framework**

In Contrast: Old System



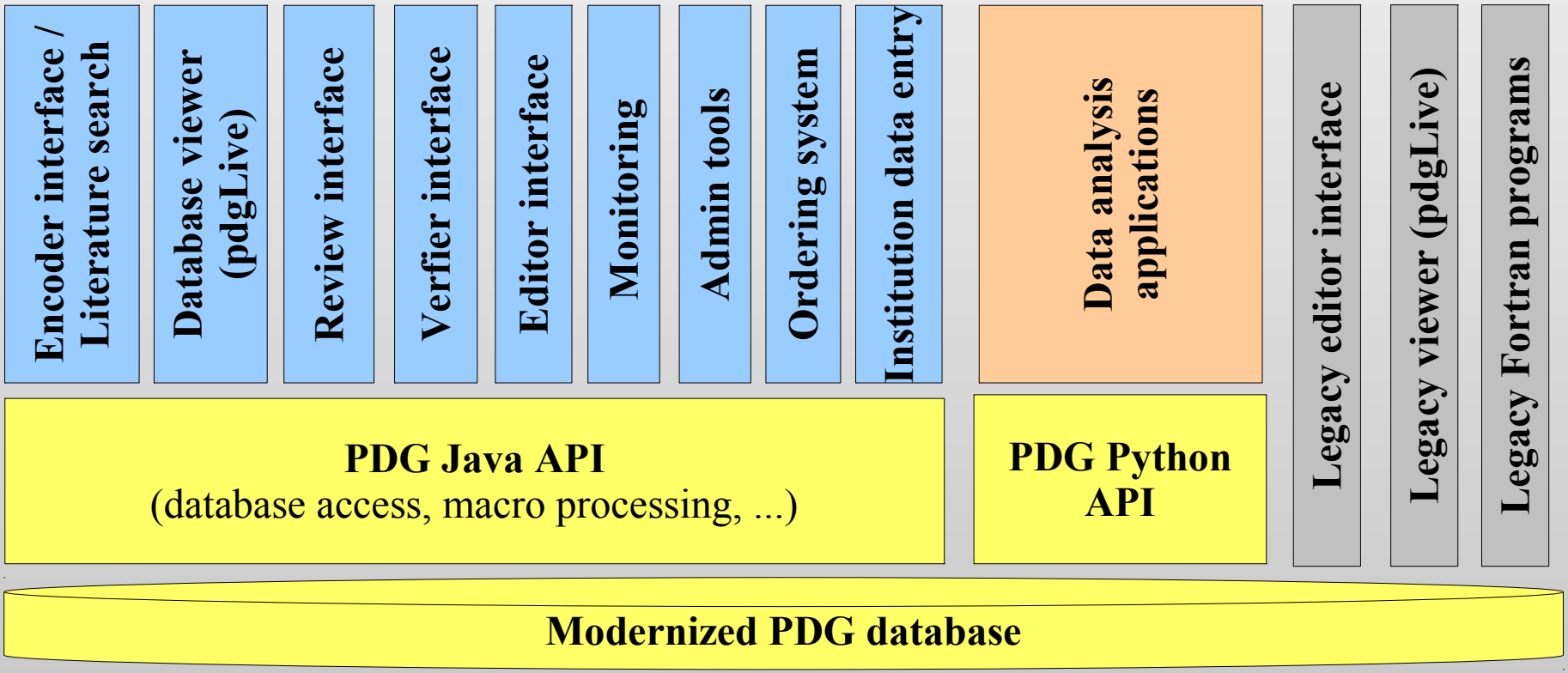
Architecture

= Web applications for collaborators (PDGworkspace) and the public (pdgLive)

= Data analysis applications

= Legacy applications (to be phased out eventually)

= Infrastructure (APIs, database)



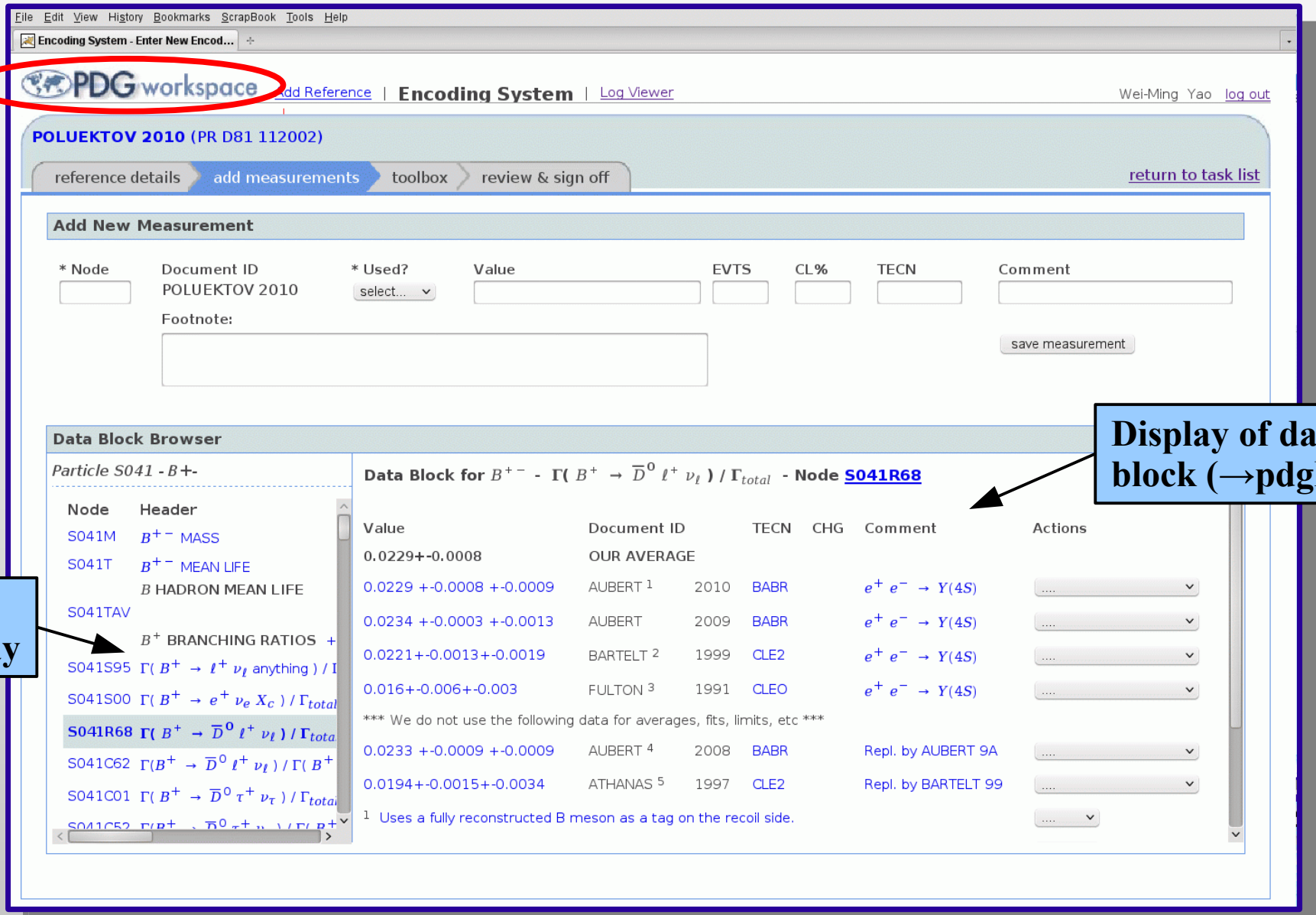
Initial Release (V0 Release)

- **Represents backbone of new system, including**
  - **Modernized PDG database**
  - **Java and Python APIs**
  - **All technologies included and working together**
- **All (updated) legacy applications run in V0 Release system**
  - Full **production system** – **now used for ongoing PDG work**
- **Provides modular framework into which applications can be easily and incrementally included (during ongoing PDG work)**
- **Includes alpha release of encoding interface**
  - **By far most difficult and complex application**
  - Includes main building blocks required by other applications
  - Supports complete standard encoding cycle plus advanced tools

Successfully deployed August 11, 2010

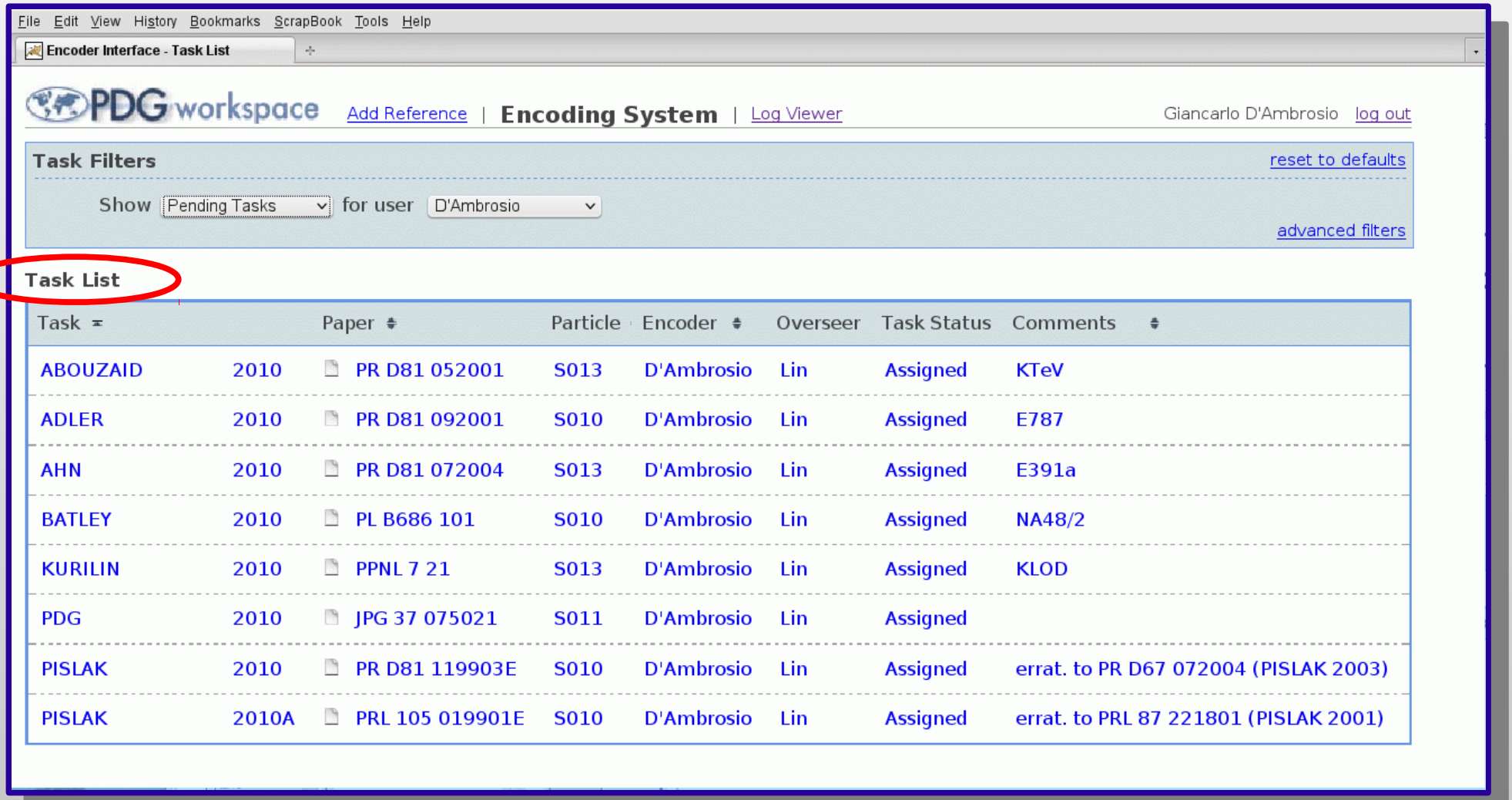
Sneak Preview I

- **Encoding interface (alpha release):**



Workflow Management

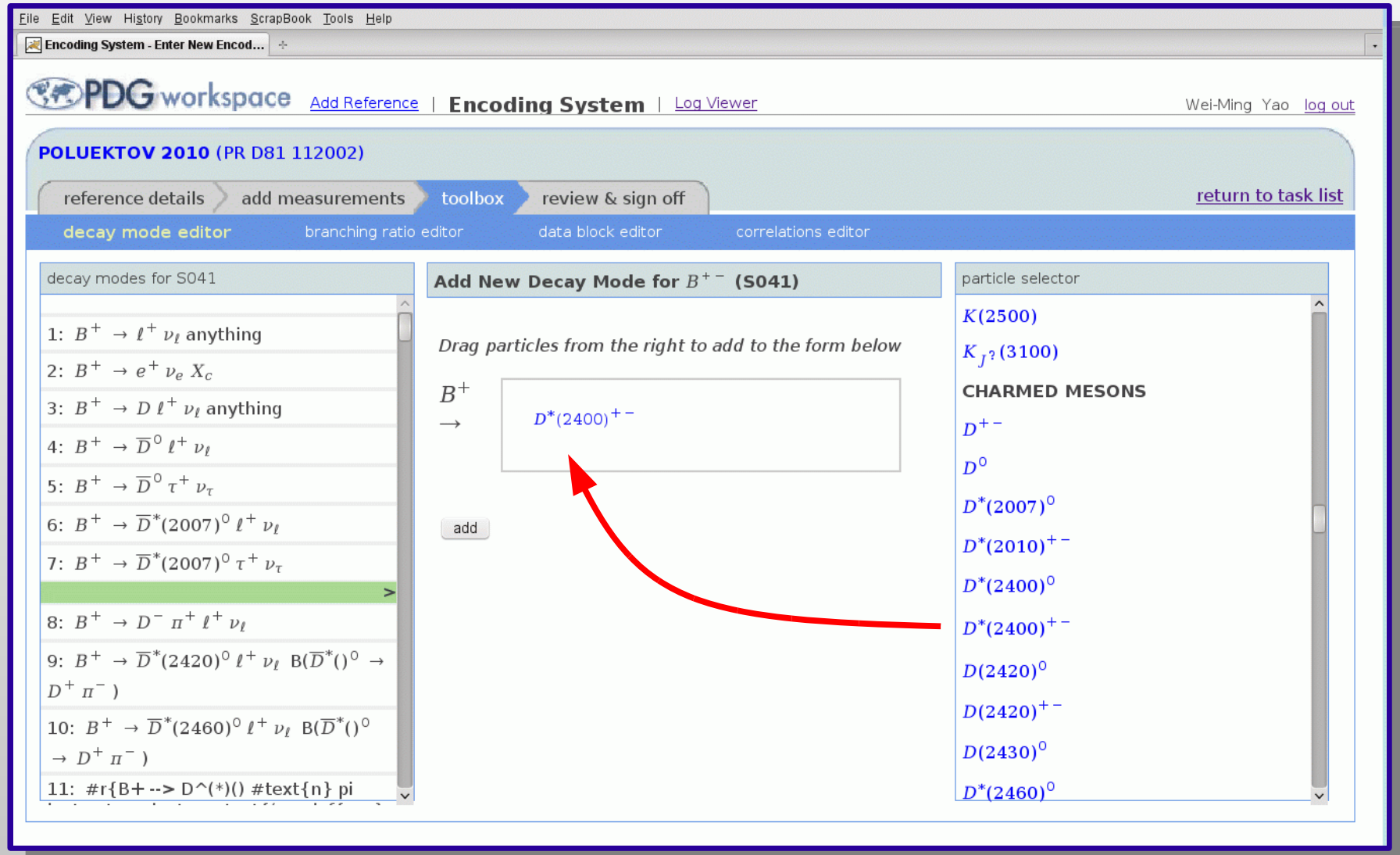
- **Manages work flow of data entry and verification for each measurement entered into RPP**



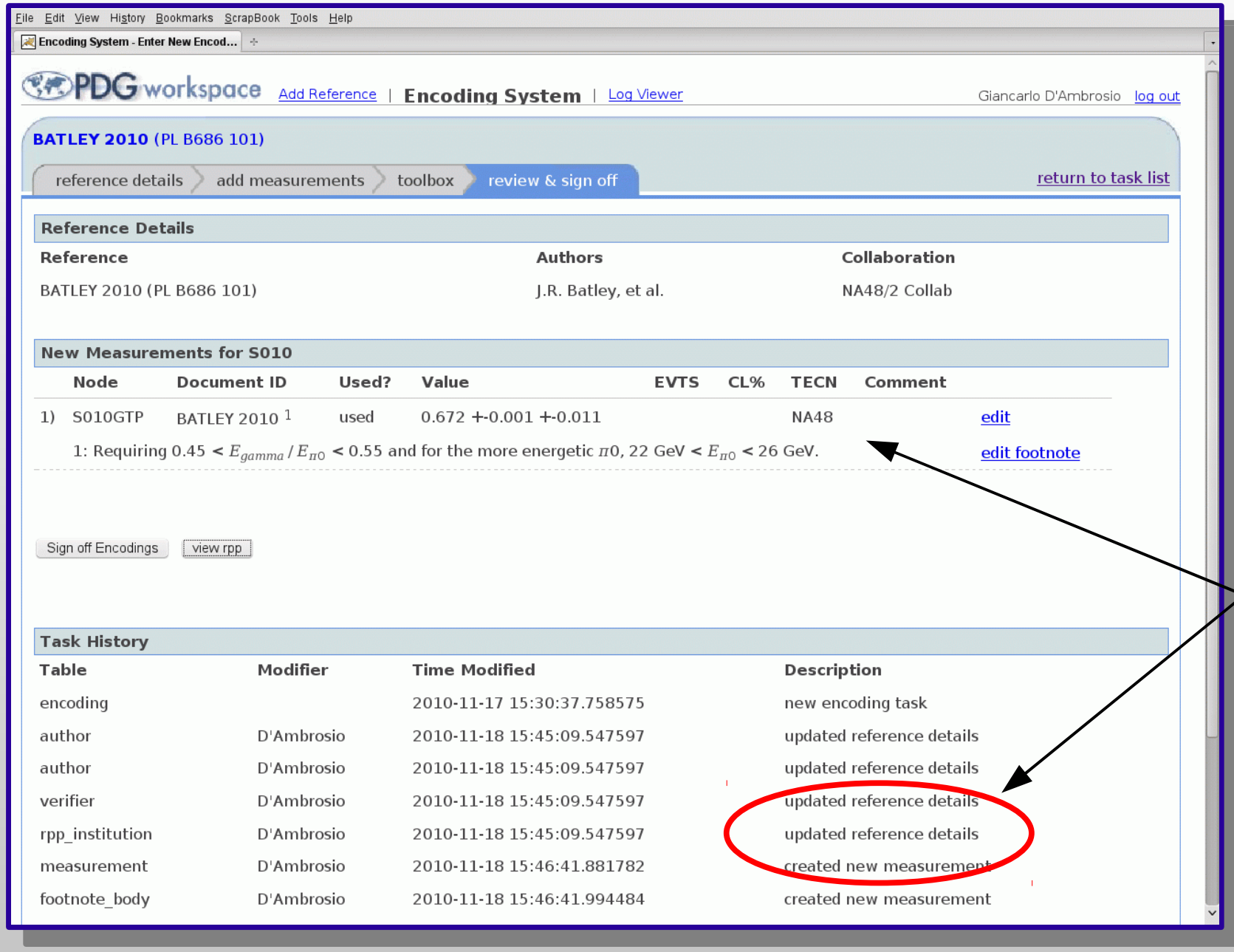
Task		Paper	Particle	Encoder	Overseer	Task Status	Comments
ABOUZAID	2010	PR D81 052001	S013	D'Ambrosio	Lin	Assigned	KTeV
ADLER	2010	PR D81 092001	S010	D'Ambrosio	Lin	Assigned	E787
AHN	2010	PR D81 072004	S013	D'Ambrosio	Lin	Assigned	E391a
BATLEY	2010	PL B686 101	S010	D'Ambrosio	Lin	Assigned	NA48/2
KURILIN	2010	PPNL 7 21	S013	D'Ambrosio	Lin	Assigned	KLOD
PDG	2010	JPG 37 075021	S011	D'Ambrosio	Lin	Assigned	
PISLAK	2010	PR D81 119903E	S010	D'Ambrosio	Lin	Assigned	errat. to PR D67 072004 (PISLAK 2003)
PISLAK	2010A	PRL 105 019901E	S010	D'Ambrosio	Lin	Assigned	errat. to PRL 87 221801 (PISLAK 2001)

Defining Decays

- **How can a casual user correctly enter items involving math?**
  - Solution: drag and drop

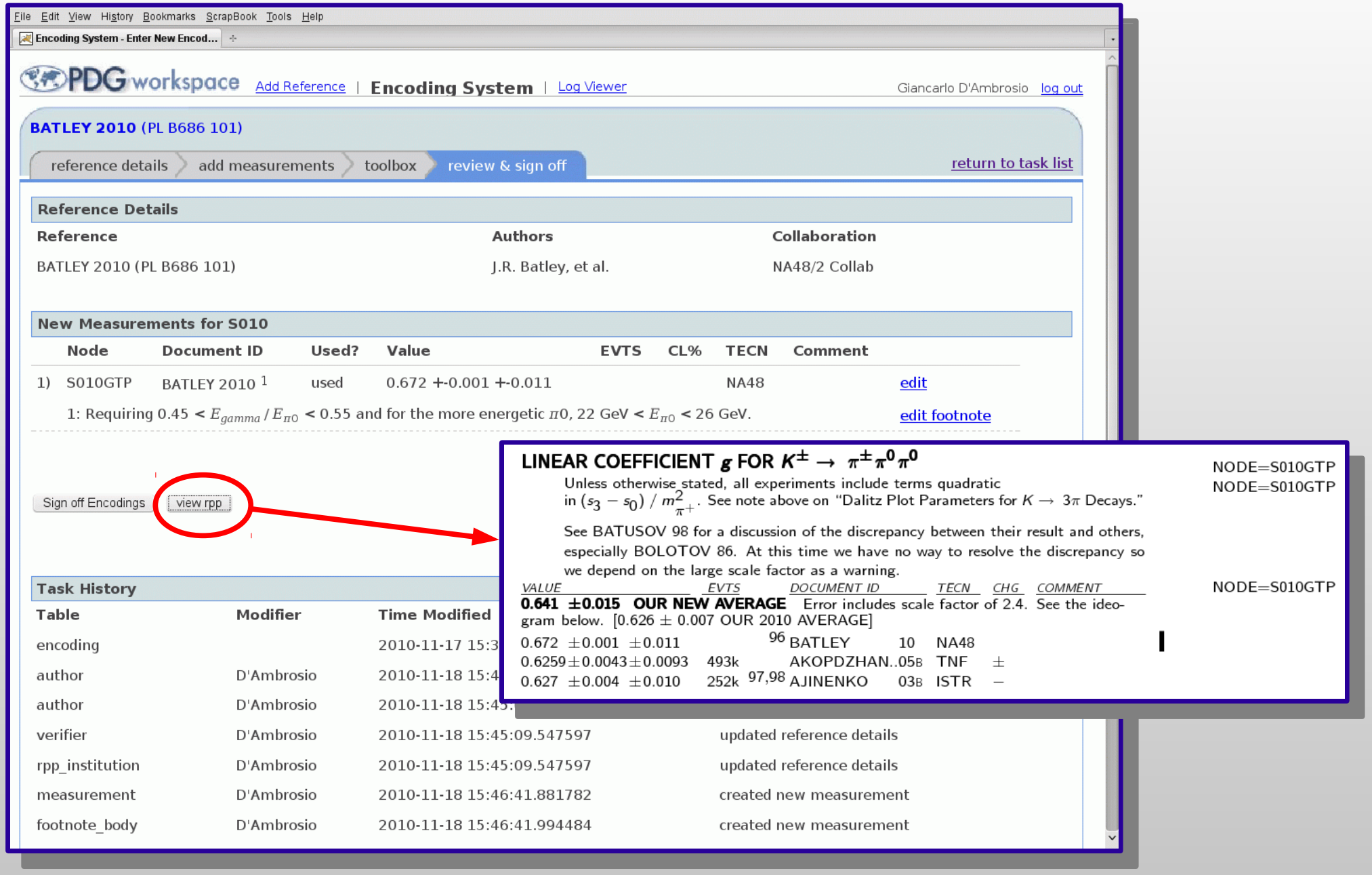


Detailed Logging



Detailed task-based logging of all changes

Preview of RPP Pages w/Fits



File Edit View History Bookmarks ScrapBook Tools Help

Encoding System - Enter New Encod...

PDG workspace | Add Reference | **Encoding System** | Log Viewer | Giancarlo D'Ambrosio log out

BATLEY 2010 (PL B686 101)

reference details | add measurements | toolbox | review & sign off | return to task list

Reference Details

Reference	Authors	Collaboration
BATLEY 2010 (PL B686 101)	J.R. Batley, et al.	NA48/2 Collab

New Measurements for S010

	Node	Document ID	Used?	Value	EVTS	CL%	TECN	Comment	
1)	S010GTP	BATLEY 2010 [1]	used	0.672 +-0.001 +-0.011			NA48		edit

1: Requiring $0.45 < E_{gamma} / E_{\pi 0} < 0.55$ and for the more energetic $\pi 0$, 22 GeV $< E_{\pi 0} <$ 26 GeV. edit footnote

Sign off Encodings | view rpp

Task History

Table	Modifier	Time Modified	
encoding		2010-11-17 15:3	
author	D'Ambrosio	2010-11-18 15:4	
author	D'Ambrosio	2010-11-18 15:45:	
verifier	D'Ambrosio	2010-11-18 15:45:09.547597	updated reference details
rpp_institution	D'Ambrosio	2010-11-18 15:45:09.547597	updated reference details
measurement	D'Ambrosio	2010-11-18 15:46:41.881782	created new measurement
footnote_body	D'Ambrosio	2010-11-18 15:46:41.994484	created new measurement

LINEAR COEFFICIENT g FOR $K^{\pm} \to \pi^{\pm}\pi^0\pi^0$ NODE=S010GTP

Unless otherwise stated, all experiments include terms quadratic in $(s_3 - s_0) / m^2_{\pi^+}$. See note above on "Dalitz Plot Parameters for $K \to 3\pi$ Decays." NODE=S010GTP

See BATUSOV 98 for a discussion of the discrepancy between their result and others, especially BOLOTOV 86. At this time we have no way to resolve the discrepancy so we depend on the large scale factor as a warning.

VALUE	EVTS	DOCUMENT ID	TECN	CHG	COMMENT
0.641 ±0.015 OUR NEW AVERAGE Error includes scale factor of 2.4. See the ideogram below. [0.626 ± 0.007 OUR 2010 AVERAGE]					
0.672 ±0.001 ±0.011		[96] BATLEY	10	NA48	
0.6259±0.0043±0.0093	493k	AKOPDZHAN..05B	TNF	±	
0.627 ±0.004 ±0.010	252k	[97,98] AJINENKO	03B	ISTR	−

NODE=S010GTP

User Feedback

- **At PDG Collaboration meeting yesterday, our collaborators could test-drive the alpha release of the encoding interface**
  - **Received lots of very useful feedback, many detailed suggestions**
  - As expected from alpha release, some hickups and problems (all of which are relatively easy to fix, now that we know of them), and some features are not yet implemented (e.g. online help)
  - Database and server held up well
- **Some feedback:**
  - **Generally very positive feedback**
    - Particularly given the caveats of the present alpha release
  - People indicated they would use the new system once available
  - Questionnaire asking people to rate different aspects on a scale of: “very much”, “somewhat”, “undecided”, “not really”, “no”
  - Functionaliy and workflow rated “somewhat” to “very much” intuitive and easy to use
  - "... **my first impression is that it is just what I wanted.**"
  - "thanks! that will be very useful for all of us!"

Sneak Preview II

- **Interactive access to PDG database in Python**
  - For now primarily aimed at PDG-internal use, but programmatic user access to PDG database will open whole new world of possibilities

jb on crabli: /home/jb - Shell - Konsole

Session Edit View Bookmarks Settings Help

```
[pdg6] ~ $ python
Python 2.5.1 (r251:54863, Sep  3 2010, 12:53:37)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-46)] on linux2
Type "help", "copyright", "credits" or "license" fo
>>> import pdg
>>> db = pdg.connect()
>>> help(db)

>>> topMass = db.getDataBlock('Q007TP')
>>> print topMass

Q007TP MEASUREMENTS

+------------------------+----------------+-----------------+
|       VALUE(GeV)       |     Used?      |   DOCU          |
+------------------------+----------------+-----------------+
| 172.0 +-0.9 +-1.3      | OUR EVALUATION |                 |
| 186+-10+-5.7           |                | ABE 1997R       |
| 168.4+-12.3+-3.6       |                | ABBOTT 1998D    |
| 167.4+-10.3+-4.8       |                | ABE 1999B       |
| 176.1+-5.1+-5.3        |                | AFFOLDER 2001   |
| 180.1 +-3.6 +-3.9      |                | ABAZOV 2004G    |
| 180.7 +15.5-13.4 +-8.6 |                | ABULENCIA 2007J |
| 171.5 +-1.8 +-1.1      |                | ABAZOV 2008AH   |
| 172.7 +-1.8 +-1.2      |                | AALTONEN 2009J  |
| 171.1 +-3.7 +-2.1      |                | AALTONEN 2009K  |
| 171.2 +-2.7 +-2.9      |                | AALTONEN 2009O  |
| 174.7 +-4.4 +-2.0      |                | ABAZOV 2009AH   |
| 174+-10+13-12          | NOT USED       | ABE 1994E       |
| 199+19-21+-22          | NOT USED       | ABACHI 1995     |
```

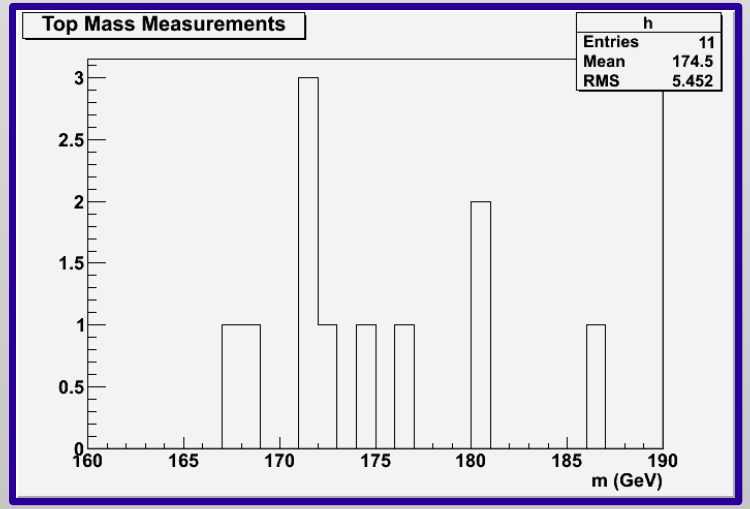
Shell

jb on crabli: /home/jb - Shell - Konsole

Session Edit View Bookmarks Settings Help

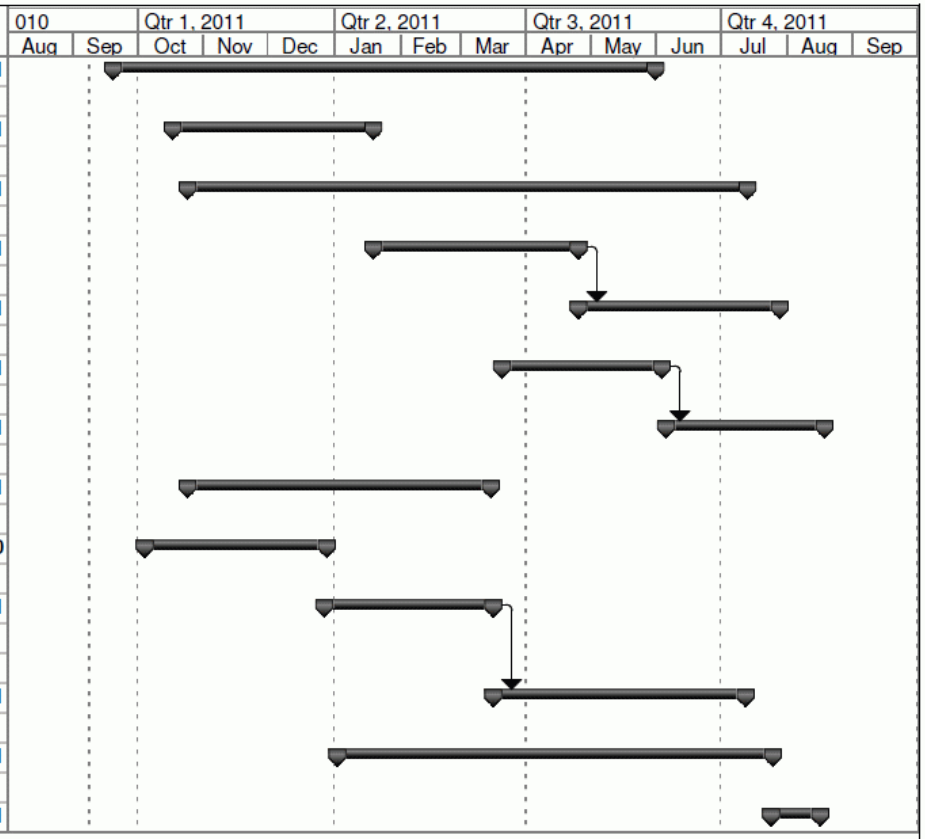
```
>>> import ROOT
>>> h = ROOT.TH1F('h','Top Mass Measurements;m (GeV)',30,160,190)
>>> for m in topMass.getUsedMeasurements():
...     bin = h.Fill(m.getValue())
...
>>> h.Draw()
<TCanvas::MakeDefCanvas>: created default TCanvas with name c1
>>>
```

Shell



Future Plan (Summary)

ID	Task Name	Duration	Start	Finish
1	Consolidation	182 days?	Mon 9/20/10	Tue 5/31/11
13				
14	Encoder interface	68 days?	Mon 10/18/10	Wed 1/19/11
22				
23	API extensions	188 days?	Mon 10/25/10	Wed 7/13/11
27				
28	Database viewer (pdgLive)	68 days?	Thu 1/20/11	Mon 4/25/11
33				
34	Review interface	68 days?	Tue 4/26/11	Thu 7/28/11
40				
41	Verifier interface	55 days?	Mon 3/21/11	Fri 6/3/11
47				
48	Editor interface	54 days?	Mon 6/6/11	Thu 8/18/11
53				
54	Reporting / Monitoring	102 days?	Mon 10/25/10	Tue 3/15/11
61				
62	Ordering system (including user prefere	61 days?	Tue 10/5/10	Tue 12/28/10
71				
72	Data analysis environment	57 days?	Tue 12/28/10	Wed 3/16/11
77				
78				
79	Fortran program migration (partial)	84 days?	Thu 3/17/11	Tue 7/12/11
81				
82	Admin tools	146 days?	Mon 1/3/11	Mon 7/25/11
88				
89	Institution database interface	17 days?	Mon 7/25/11	Tue 8/16/11



Project completion expected mid August 2011

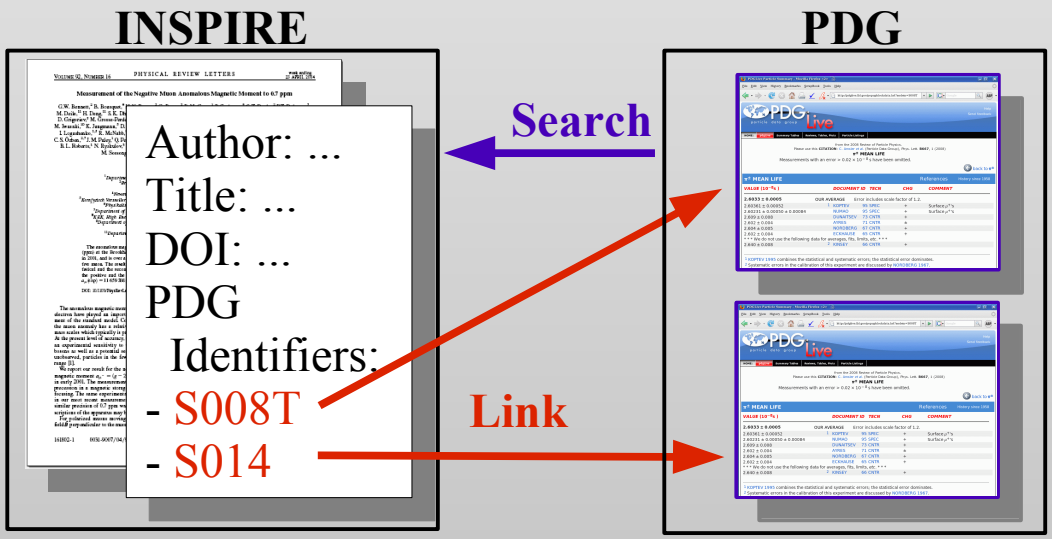
- **Leaves contingency of 3 months of funding (at present level of development effort)**

Beyond the Core Project

- **Immediate and primary goal is to ensure PDG can continue to function well**
  - This has absolute priority over any fancy extensions
- **New computing system also provides platform where innovative new features can be implemented**
- **Several activities already started in this context:**
  - Collaboration with **INSPIRE** on cross-linking using **PDG Identifiers**
  - Participation in **HEP Information Resource Summits**
  - Presentation at CHEP'2010
  - Brain-storming about new features, e.g.
    - pdgLive on smart phones
    - Opening PDG platform to support averaging groups (e.g. HFAG)
    - User tagging of PDG content
    - Allowing programmatic user access to PDG database
    - Providing all PDG data in computer-readable form
    - ...

Cross-Linking

- **Wish list:**
  - From INSPIRE: *"What data does PDG have about this?"*
  - From PDG: *"What are the latest papers on this topic?"*
- **Permanent reference to PDG data items: PDG Identifiers**
  - Essentially PDG nodes (e.g. "Q007TP")
  - PDG will publish authoritative list
- **Map to other classification schemes or use as "pointers"**



- Generate initial set of tags from PDG database
- Could allow authors to tag their articles

Conclusions

- **Thanks to funding received after very successful 2008 DOE Review of PDG, the planned computing upgrade is under way and expected to be completed around mid-August 2011**
  - **"... is proceeding on schedule and within cost"** (DOE review 2010)
- **An initial version of the new system has been successfully deployed and is now the PDG production system**
  - First release of modern**, extendable** and **maintainable** PDG system
  - Represents complete backbone of new system
  - Provides modular framework into which applications can be easily and **incrementally** included (during ongoing PDG work)
- **Remainder of project devoted to building other user interfaces**
  - Main **building blocks needed already available** and working in encoding interface
- **Now have platform for implementing innovative new features**
  - First example will be greatly improved cross-linking with INSPIRE

Backup Slides

Support for Reviews

- **Not yet implemented**
- **Will also be part of PdgWorkspace**
- **Planned features include:**
  - Workflow management (authoring, refereeing; notifications)
  - Two working models supported:
    - Edit review in web-based editor (no need for local TeX/TeXsis)
    - Check out review sources, work locally, check in modified versions
  - Convenient access to different versions (including differences)
  - Consistency checking of related reviews and Listings
  - Status reporting
  - Mailing lists for each review with automatic subscription of review authors and/or referees

Project Team

- **Juerg Beringer (PDG physicist)**
  - Project leader, requirements, system architecture
- **Chuck McParland (computer scientist)**
  - Java API
- **Sarah Poon (computer systems engineer)**
  - Web design, user interfaces, JavaScript
- **David Robertson (computer systems engineer)**
  - Database, Python API, scripts
- **Orin Dahl (PDG physicist, retired)**
  - Legacy Fortran programs
- **Piotr Zyla (PDG editor)**
- **Contributions from Jacob Andreas, Cecilia Aragon, Keith Beattie, Igor Gaponenko, Keith Jackson, Kirill Lugovsky, Slava Lugovsky**

Each member of the team has many years of software development experience

Technologies

- **J2EE-based web application framework**
  - Commonly used industry standard for scalable, distributed web apps
- **Ajax-enabled web pages**
  - User-friendly and highly interactive GUI behavior
- **Relational database (PostgreSQL)**
  - Currently 130 database tables
  - ORM tools (Object-Relational Mapping): Hibernate, JPA; SqlAlchemy
- **Programming languages**
  - Java and JSP for web application framework backend
  - JavaScript and CSS for client-side HTML (Ajax)
  - Python API for programmatic access to database and to interface to numerical libraries and tools
  - Legacy Fortran applications restructured as libraries
- **TeX (print, PDF) and MathML (web)**

Challenges, Risk, and Solutions

- **PDG has special requirements that cannot be addressed by "commodity software"**

  Solution: ✔
  - Identified challenging areas posing potential risk to project
  - Carefully addressed these areas **first** (through **design**, **technology choices**, and **project planning**)

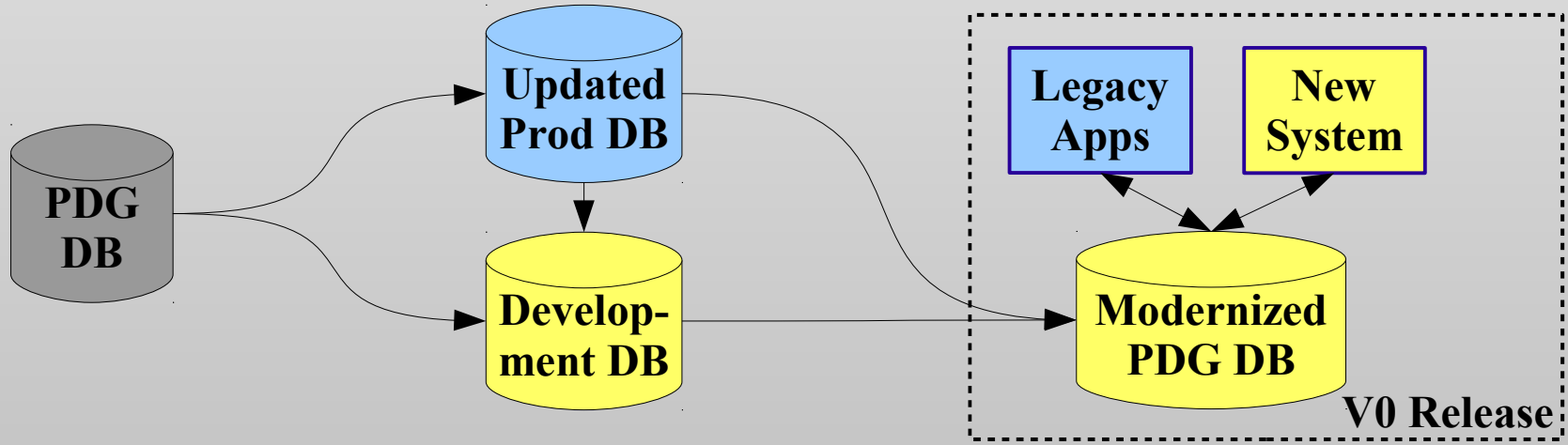
- **Computing upgrade must proceed in parallel to PDG work**
  - Legacy system must continue to run during development
  - Severely limits opportunities for system deployment (once per year)
  - Workload on PDG experts from having to work with two systems

  Solution: ✔
  - Must carefully plan new system deployment
  - Release as early as possible with legacy applications running within new system ("V0 Release", see later)
  - Allows incremental deployment of new components

Challenges, Risk, and Solutions

- **Existing scientific data must be migrated to new system**
  - **Complete redesign of PDG database from scratch impractical from many points of view**
  - Changes to PDG database must be made incrementally
  - Small database changes mandated by ongoing PDG work
    - Conventions on how data is stored in the database (macros, flags, etc)
    - Occasionally need new columns in tables

Solution: ✔
- Modernized PDG database used by both (updated) legacy applications and the new system



Challenges, Risk, and Solutions

- **Scientific output from old and new system must be identical; PDG data must be correct**
  - Inherently difficult to validate tens of thousands of numbers

Solution: ✔

- Nightly builds with unit tests
- Careful and detailed validation before use for PDG production
- Detailed logging of changes at database level
- Version control of database contents by dumping to CVS
- **System validation by producing TeX manuscript of full Review in old and new system, then making sure all changes ("diff") are expected and desired**

Challenges, Risk, and Solutions

- **Distributed data entry**
  - System must take care of complicated distributed work flow
  - Detailed logging of changes (“Why did this number change?”)

Solution: ✔

- Careful design
- Suitable industry-standard technology choices (J2EE)
- Innovative logging scheme using database triggers that keeps track of logical operations and enforces logging at database level for any application (doesn't need any application specific logging support)

Challenges, Risk, and Solutions

- **Use of TeX and display of math on the web**

  Solution: ✔
  - Evaluate existing solutions (MathML, jsMath, mimeTex, TeX-to-MathML translators, ...)
  - Found solution that addresses our needs (see Sarah's talk)

- **Browser and platform diversity among large user base**

  Solution: ✔
  - Use existing extensive JavaScript library where this problem is already solved (see Sarah's talk)

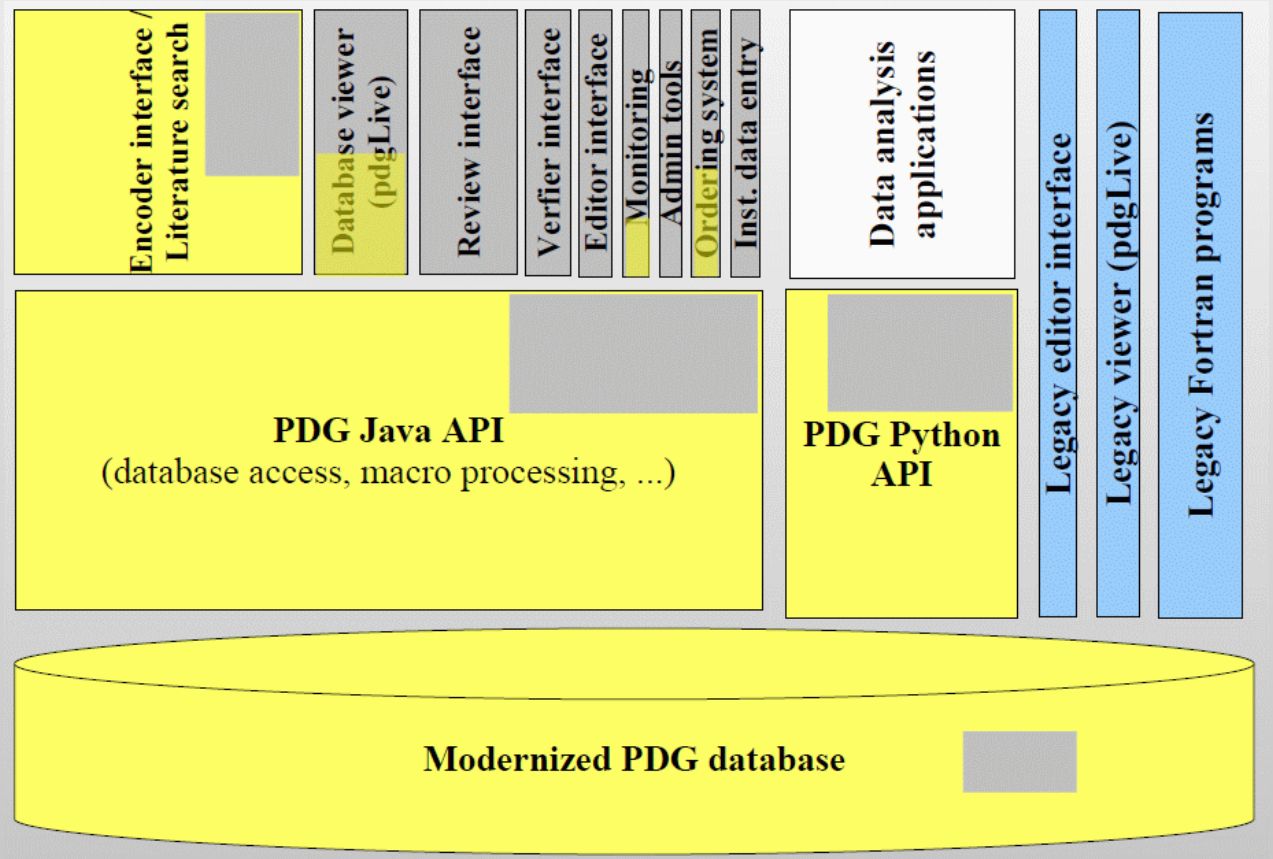
Development Process

- **Follows widely-adopted practices, including**
  - Iterative design process with close interaction with users
  - Ongoing documentation (Wiki, within code, formal manuals)
  - Nightly builds and nightly unit tests
  - Using existing tools, components and libraries to maximize efficiency
- **Frequent communication**
  - Weekly general meetings
  - Weekly individual meetings of developers with project leader
  - Additional meetings as needed
  - Mailing list
- **Close involvement of PDG members**
  - So far through Orin, Piotr and myself (plus occasionally Cheng-Ju Lin and Weiming Yao)
  - As user testing ramps up, will increasingly involve other members of LBNL PDG group plus selected members from PDG collaboration

V0 Release vs Full System

- **Rescaled diagram to reflect approximate development effort**

= updated legacy applications (in V0 release)
= new components included in V0 release
= still to be implemented as part of upgrade (some partly done)



Current Status of Key Tasks

- **Initial design and planning** ✔
- **System architecture** ✔
- **Database abstraction layer** ✔
- **Encoder interface and literature search interface** mostly ✔
- **Database viewer** (main building blocks available)
- **Data analysis environment** partly ✔
- **Review interface**
- **Other system tasks**
  - Refactor existing auxiliary programs ✔
  - Status monitoring
  - System monitoring partly ✔
  - Verifier interface
  - Editor interface
  - Ordering system partly ✔
  - Institution data entry
- **Final acceptance test**

Current Status of Key Tasks

- Initial design and planning ✔
- System architecture ✔
- Database abstraction layer ✔
- Encoder interface and literature search interface mostly ✔
- Databas
- Data ana
- Review i
- Other sy
  - Refact
  - Status
  - Syster
  - Verifie
  - Editor
  - Orderi
  - Institution data entry
- Final acceptance test

- All difficult parts posing potential risk to the project are implemented
- The encoder interface is by far the most complex and difficult application to implement
- The encoder interface includes the building blocks needed for the other applications (e.g. macro processing, math display, etc)
- Therefore, building the remaining applications will be relatively fast

Source Code Size (as of Sep 2010)

To give an approximate measure of the size of the source code developed, here are some numbers of lines of source code:

- **Java API** **75k**
  - Related to database (of which 38k generated) 44k
  - Related to macro processing 22k
  - Related to unit tests 9k
- **Encoder interface** **16k**
  - Java 8k
  - CSS 2k
  - HTML, JSP, JavaScript 6k
- **Python API** **1k**
- **Migration scripts (SQL, some Python)** **3k**
- **Legacy Fortran programs (incl. 45K comment lines)** **110k**