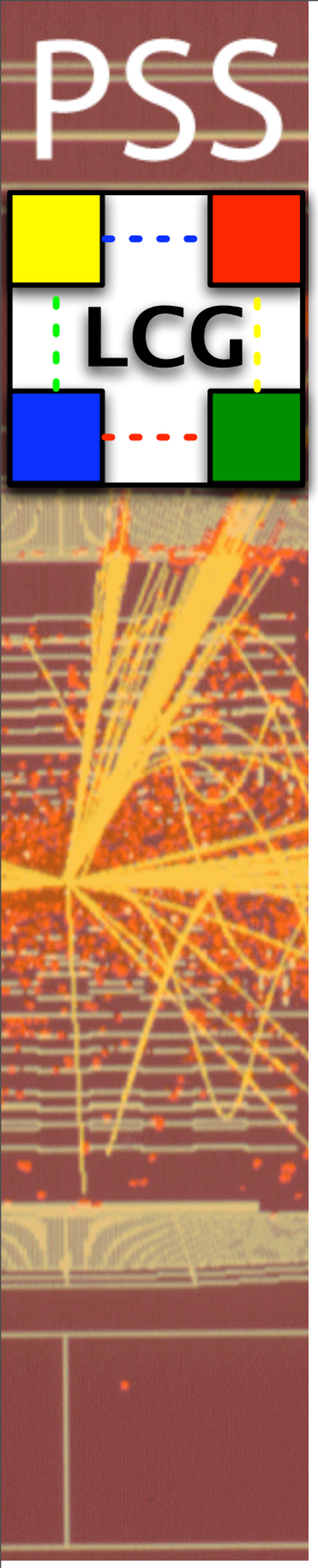


Development Status and Plans for the LCG Common Database Access Layer CORAL

Dirk Duellmann, CERN IT

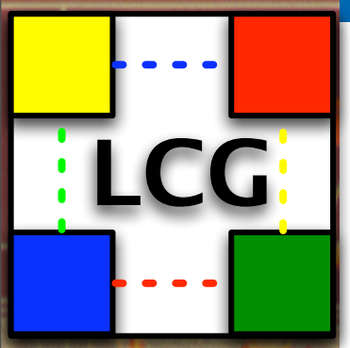
on behalf of the LCG Persistency Framework Project

http://pool.cern.ch & http://pool.cern.ch/coral

CHEP '07, September 3rd

Victoria, Canada

PSS

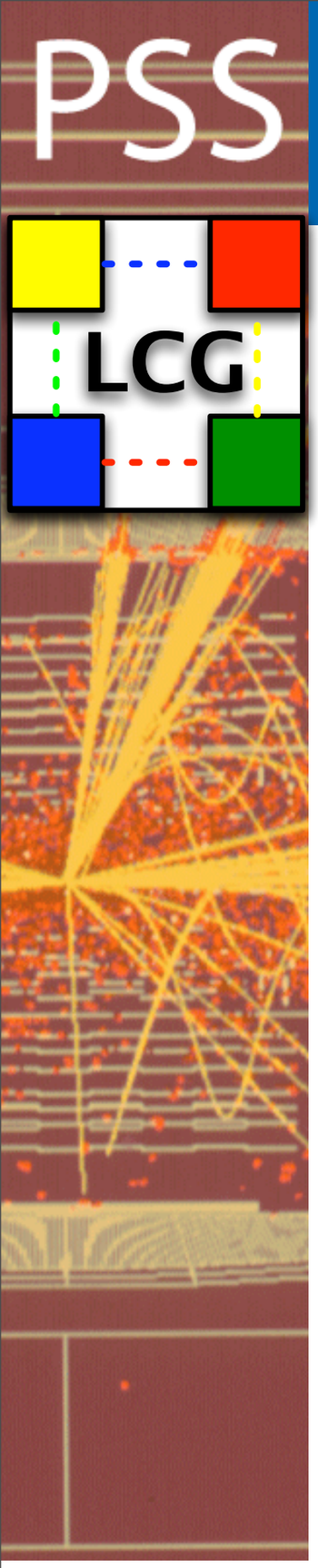


Other connected contributions at CHEP'07

CERN IT Department

- 171 - Production Experience with Distributed Deployment of Databases for the LHC Computing Grid
- 172 - CERN Database Services for the LHC Computing Grid
- 122 - Relational databases for conditions data and event selection in ATLAS
- 161 - Building a Scalable Event-Level Metadata System for ATLAS
- 186 - Development, Deployment and Operations of ATLAS Databases
- 319 - Alignment data streams for the ATLAS Inner Detector
- 333 - Large Scale Access Tests and Online Interfaces to ATLAS Conditions Databases
- 90 - ATLAS Conditions Database Experience with the LCG COOL Conditions Database Project
- 236 - Control and monitoring of alignment data for the ATLAS endcap Muon Spectrometer at the LHC
- 294 - An Inconvenient Truth: file-level metadata and in-file metadata caching in the (file-agnostic) ATLAS distributed event store
- 462 - Database architecture for the calibration of ATLAS Monitored Drift Tube Chambers
- 358 - Distributed Interactive Access to Large Amount of Relational Data
- 430 - The ATLAS METADATA INTERFACE
- 110 - Experience and Lessons learnt from running high availability databases on Network Attached Storage
- 109 - Oracle RAC (Real Application Cluster) application scalability, experience with PVSS and methodology
- 265 - LHCb Distributed Conditions Database
- 213 - LHCb experience with LFC database replication
- 89 - LHCb Online Interface to a Conditions Database
- 322 - CMS Conditions Data Access using FroNTier
- 325 - The CMS Dataset Bookkeeping Service
- 182 - Distributed Database Access in the LHC Computing Grid with CORAL
- 181 - Development Status and Plans for the LCG Common Database Access Layer (CORAL)
- 204 - COOL Software Development and Service Deployment Status
- 205 - COOL Performance Tests and Optimization
- 350 - Nightly builds and software distribution in the LCG / AA / SPI project
- 447 - Implementing a Modular Framework in a Conditions Database Explorer for ATLAS
- 292 - Explicit state representation and the ATLAS event data model: theory and practice

CORAL - A Database Foundation Layer



- Started as component of POOL- now packaged independently
  - POOL/COOL use CORAL
  - CORAL does not require either (online use)
- CORAL goals
  - database foundation for physics applications
    - local area and grid
    - all LCG AA platforms and database back-ends
  - high level interface
    - C++ and Python API
    - abstraction from SQL dialects of db vendors and connection technologies, avoids risk of vendor binding
  - high level services
    - authentication, authorisation, db service look-up, retry and fail-over (see talk #182 for details)

Dirk.Duellmann@cern.ch

Physics Applications
common or experiment specific

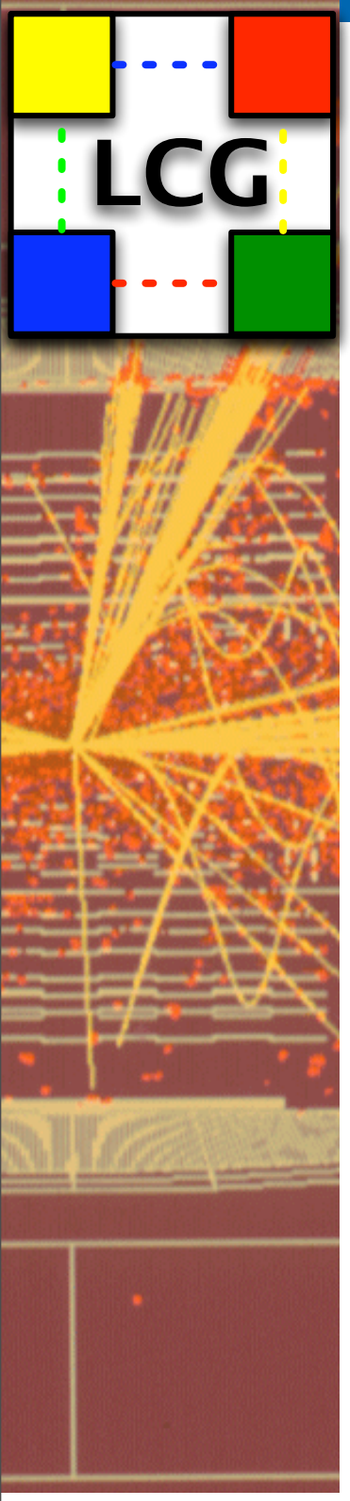
COOL
Validity Intervals,
Conditions Data

POOL
Object Storage,
Meta Data and Collections,
File Catalogs

CORAL
RDBMS Access, Indirection, Authentication/Authorisation

Computing Services
Files, Databases, Grid Services

Applications Using CORAL

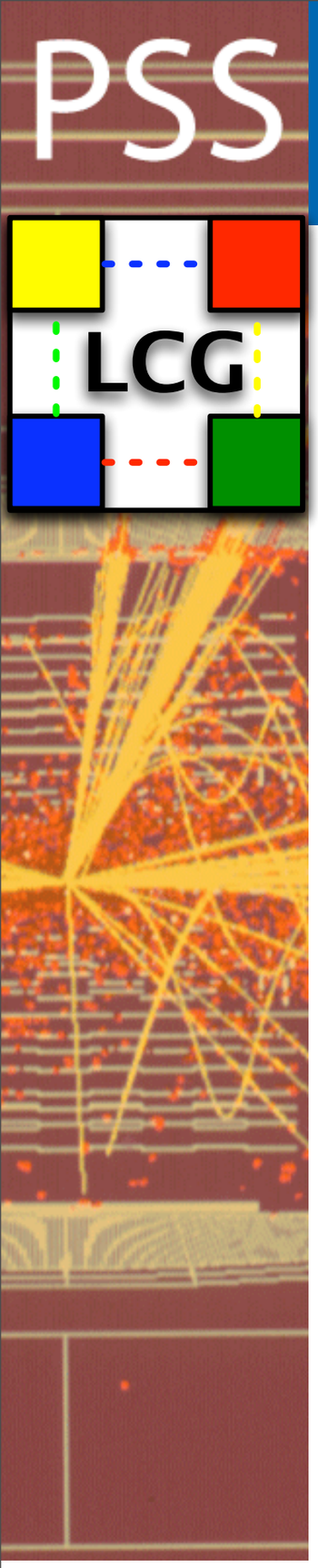


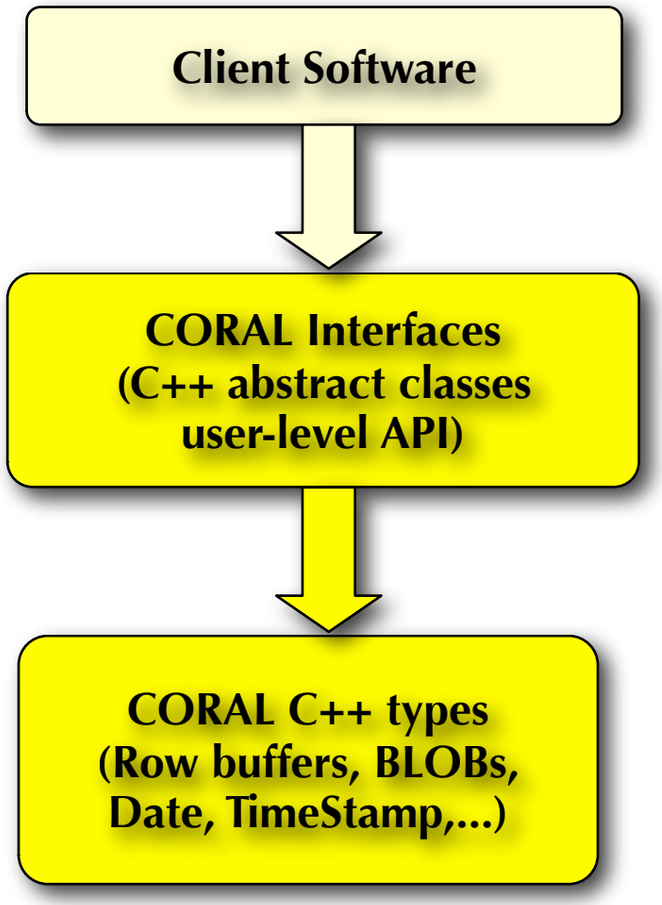
- LCG Persistency framework components POOL and COOL
  - fully replaced direct database dependencies with CORAL
    - POOL File catalogs
    - POOL Event collections
    - COOL conditions database
- CORAL is also used directly
  - ATLAS: detector geometry, several online apps
  - CMS: conditions data, POPCON framework
  - LHCb: online applications
  - Astrophysics: LSST project is evaluating CORAL for storage of large volume image data

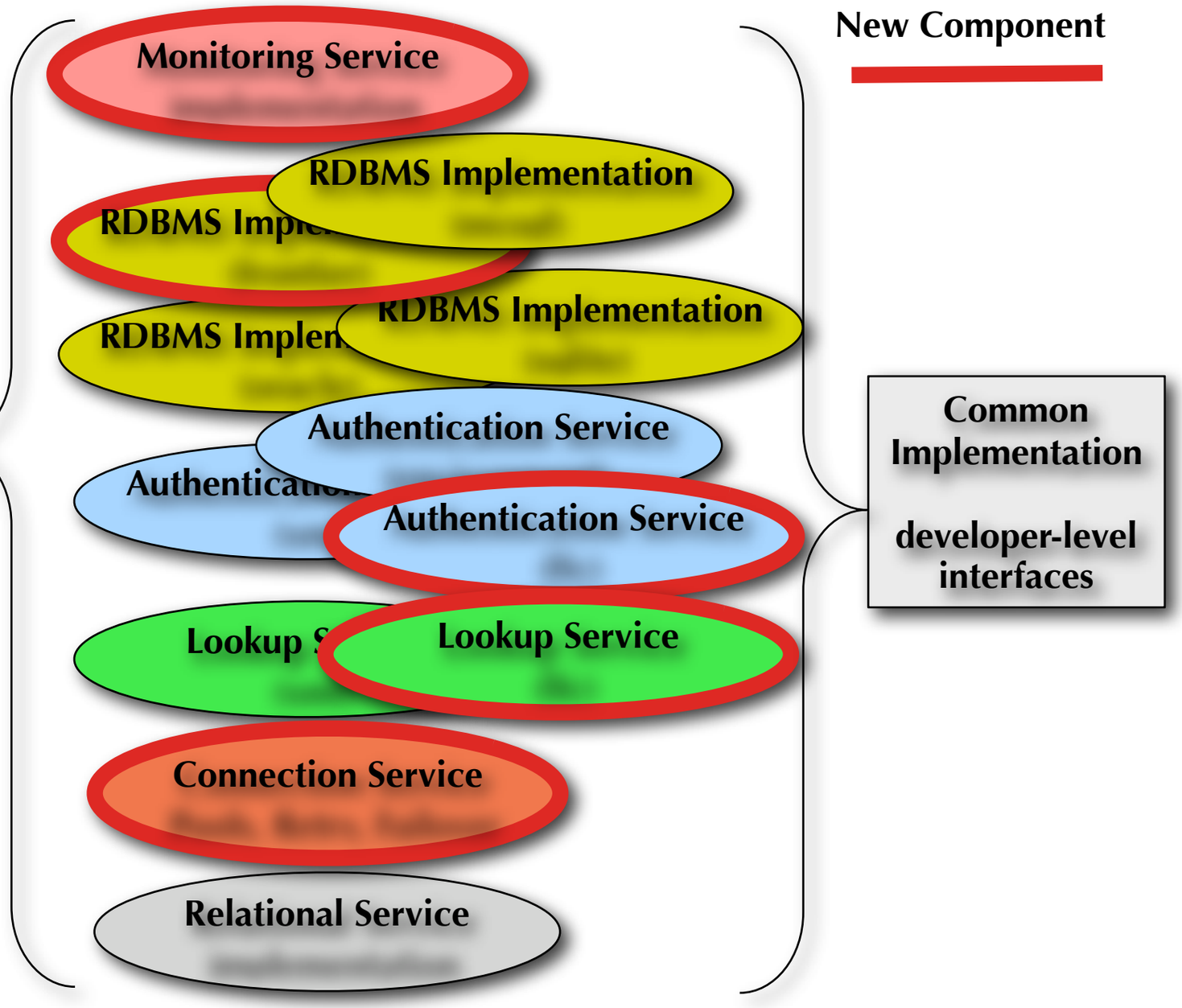
Database Back-ends

- Oracle - 10g R2
  - based on C level interface (OCI)
    - performance and flexibility in compiler choice
  - all API concepts / optimisations supported natively
    - bind variables, bulk operations, session pooling
- MySQL - V5 (and 4)
  - small to medium sized services
  - standalone development set-ups
- SQLight - V3.4
  - server-less, file based access
  - popular transport/replication medium for read-only data
- FroNTier/SQUID - caching layer
  - read-only access to Oracle via http
  - multi-level caching between server and client to avoid network and server latencies for repeated queries

CORAL Components







Plug-in libraries, loaded at run-time, interacting only through the interfaces

CORAL Python Binding

- Several new developments in cooperation with RRCAT, Indore (India)
- Python access to CORAL user level API
  - Same semantics as C++ components...
    - Python loads standard CORAL plug-in libraries
  - ...but maintaining Python look-and-feel
    - e.g. for sequences and iterators
- Quickest way to prototype/develop CORAL applications
  - e.g. for integrating database access methods into python based experiment scripts
  - e.g. develop database administration tools, web based applications

CORAL Database Copy Tool

- Copies either specific tables or full schemata between any of the CORAL back-ends
  - Handles referential integrity constraints between data in different tables
  - Support for data slices based on data queries
- Very useful utility to allow users to move relational data around between the different development, validation and production instances
- The copy tool does not
  - replace specialised applications for data exchange between complex application schemata
  - replace the need for planning big data movements together with the DB service administrators

LCG AA Internal Migration & Next Milestones

- Followed Application Area build system migration from SCRAM to CMT
  - Moved existing functional and regression tests to new nightly build framework (see also Roiser #350)
- Major milestone: remove SEAL dependency from Persistency Framework
  - replace SEAL plug-in loading and utility classes
    - showed up as problematic in multi-threaded environments (outside of initial SEAL design scope)
    - SEAL maintenance is not staffed, SEAL deprecation planned
  - POOL and COOL will follow consistently
- No external changes for package users
  - but of course significant effort for the project to achieve this transparency

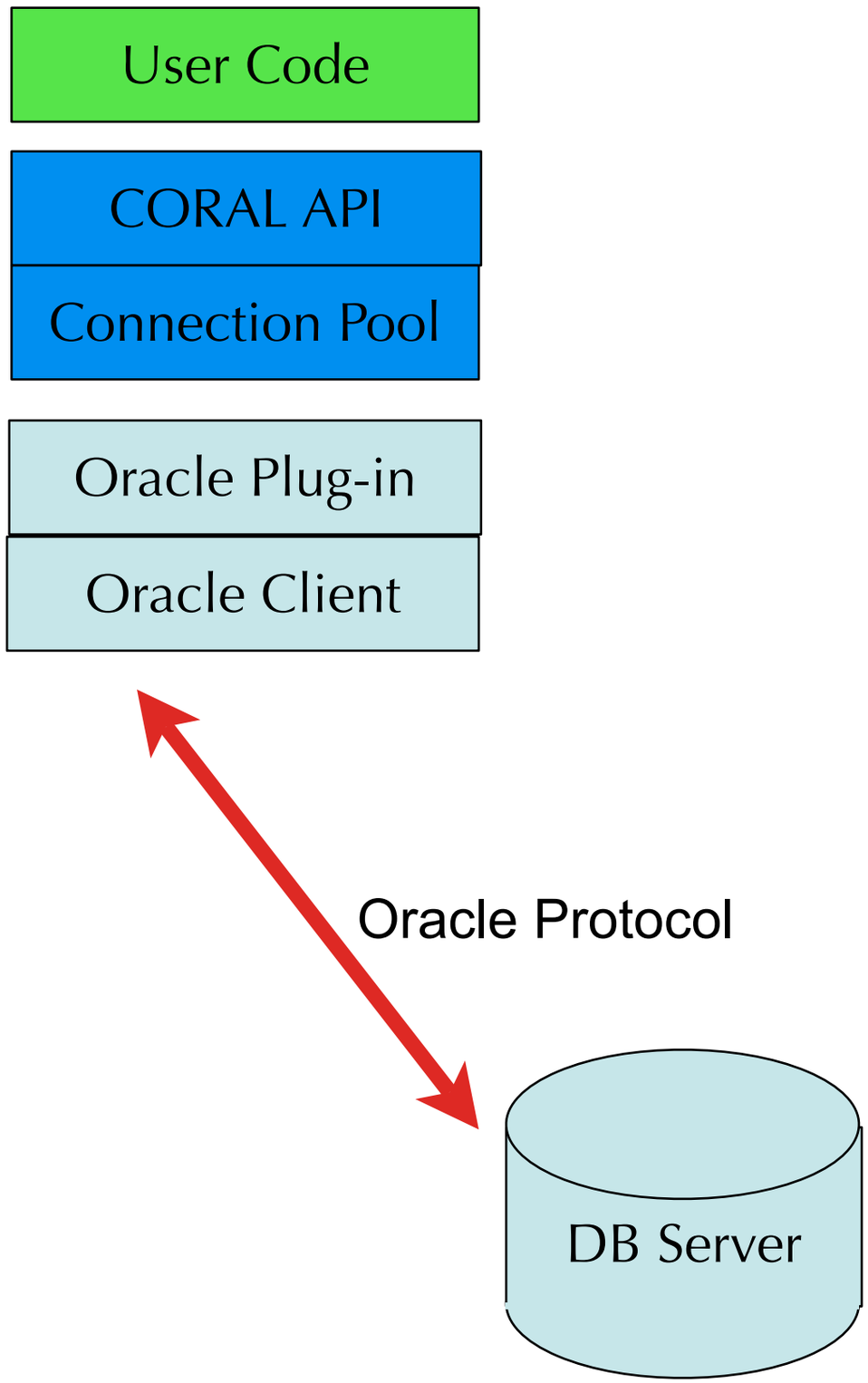
Residual DB Deployment Issues

- Offline DB access
  - many long standing client connections with limited DB activity
  - current 2-tier model leads to ineffective use of the database server resources (many, but almost idle server processes)
- Online DB access
  - many concentrated DB requests for the same data
  - ATLAS and CMS use different cache servers to achieve stringent latency requirements in their online environment
- Need to ship/maintain Oracle/MySQL client binaries to Tier sites
- Security concerns about internet access to DB servers at T0 and T1
  - open database ports require significant precautions (eg firewall closure procedures, frequent security patches) to be protected against security threats
- CORAL authentication and authorisation relies on LFC catalog for evaluation of LCG proxy certificates
  - dependency on LFC, risk of user credential exposure

A possible solution under study: a CORAL Proxy Server

- A CORAL server deployed close to the database servers could resolve many of the above issues
  - Many incoming (idle) CORAL connections can be combined into few DB server connections
    - more effective use of DB server resources
  - LCG certificates could be securely evaluated at the CORAL server
    - improved security and reduced dependency on other services
  - No vendor client libraries are required on the client machine
    - simplified s/w distribution
  - Database server ports can be hidden behind firewall
    - increased service security
  - Caching could be integrated in a general and database independent way
- The stable CORAL API and existing s/w components should allow this without affecting existing code based on already developed CORAL components

An Implementation Scenario

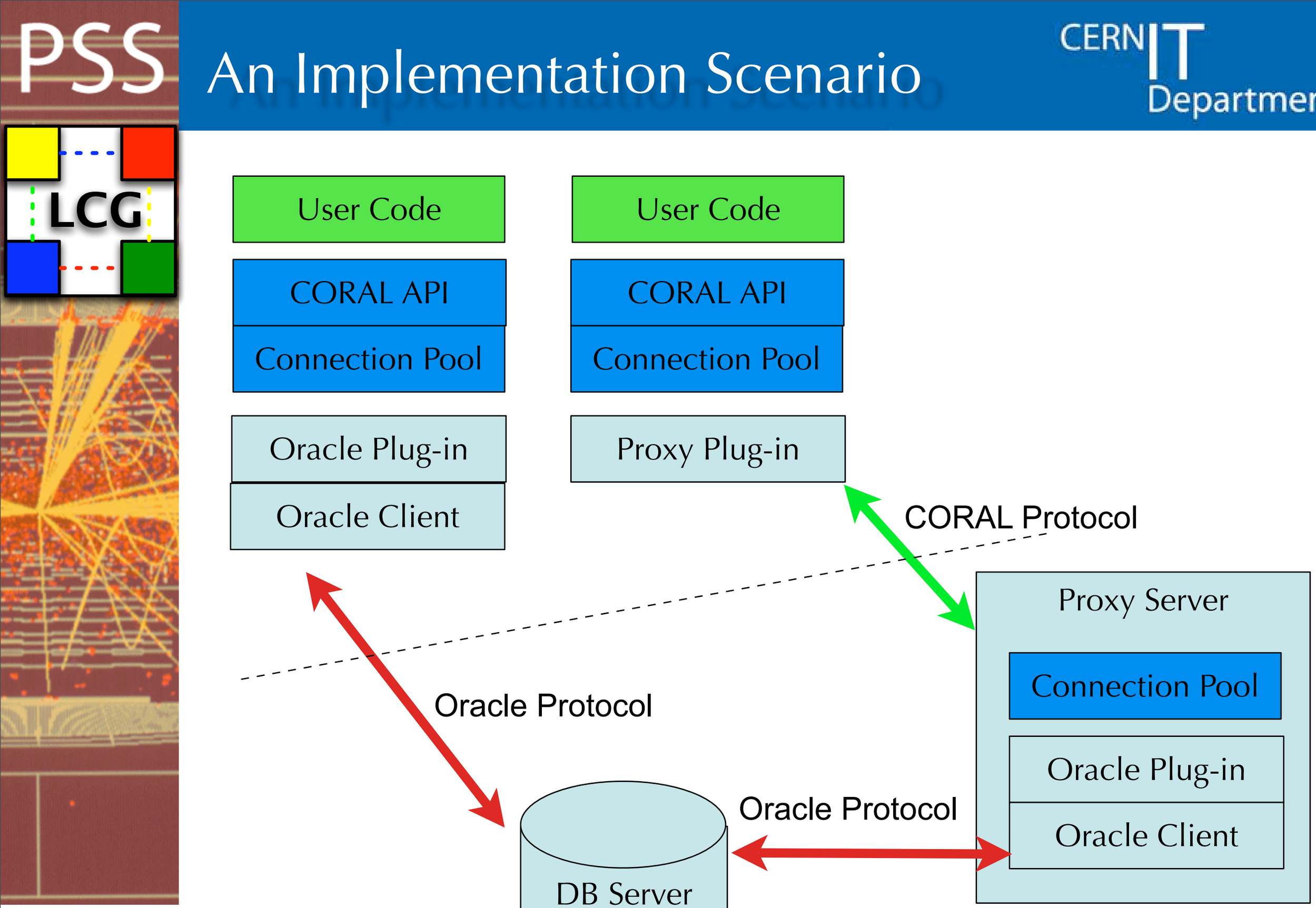


PSS

LCG

An Implementation Scenario

CERN IT Department



Summary

- Since last CHEP CORAL passed a phase of active development
  - The user API has been very stable, but many DB deployment improvements have been implemented
  - Client-side connection pool, LFC based service look-up and authentication, Python binding and DB Copy Tools
- The package is widely used by Persistency Framework and experiment code, is well integrated with the LCG database and grid services and provides a complete foundation for database applications
- Next Steps and future improvements
  - Remove dependency on SEAL from Persistency Framework
  - Investigating a multi-threaded CORAL server to further improve scalability and security for large scale LHC database deployments
  - Existing CORAL applications are expected to profit with minimal/ no code changes