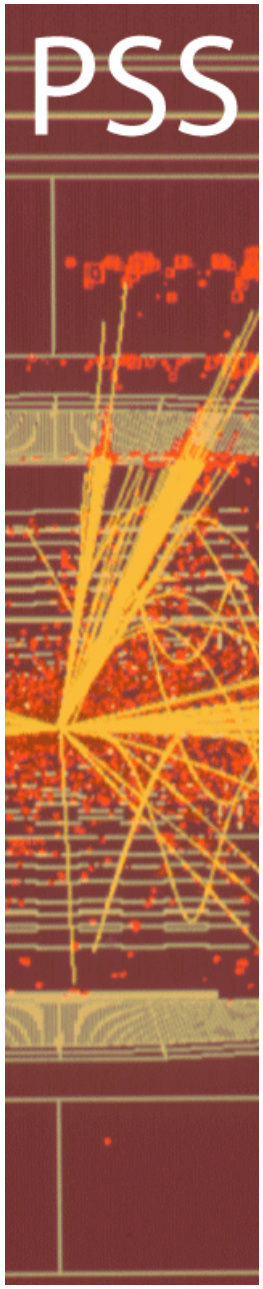
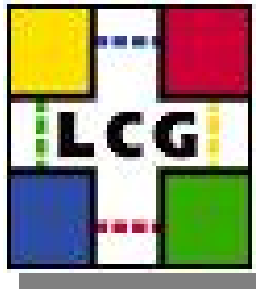




CORAL

COmmon Relational Abstraction Layer

Radovan Chytracek, Ioannis Papadopoulos

(CERN IT-PSS-DP)

Applications Area Internal Review, 18-20 September 2006

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

(http://pool.cern.ch/coral/)

The CORAL Team

- The Development Team
  - IT-PSS-DP
    - Radovan Chytracek
    - Giacomo Govi
    - Ioannis Papadopoulos
  - Experiments
    - Zhen Xie (CMS)
- The “Invisible” Team (within and outside IT)
  - Database administrators and experts
  - Other computing service developers and providers
  - SPI (for the development and release infrastructure)
  - Beta testers (from experiments, COOL developers,…)

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CORAL Project Scope

- CORAL is:
  - a software system providing interfaces and implementations for accessing relational databases
    - exposing a technology-independent API
    - serving the use cases from the LCG
    - encapsulating or forcing by design “best practices”
  - serving the needs of distributed deployment
  - the implementation basis of
    - the relational components of POOL
    - COOL
    - experiment-specific software components

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

From the last review until now

- Recommendations in the last review (related to CORAL):
  - "The reviewers welcome the split of the RAL and POOL release cycles"
    - CORAL released independently of POOL.
  - "Improve error handling; error reporting must propagate to end user with clear indication of which component in the complicated stack encountered the error, and provide sufficient description of the error"
    - A complete exception hierarchy in CORAL reporting on the error conditions and the modules throwing the exception.
    - CORAL architecture based on a thin component stack.
  - "(Security) In a distributed (grid) environment, POOL should not be the weakest point in the chain"
    - Using grid certificates in order to retrieve the authentication credentials (via LFC)

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Requirements from Deployment

- CORAL-based applications are expected to run in a grid-enabled environment
  - Multiple RDBMS technologies involved
    - Oracle on Tier 0/1, MySQL, SQLite on Tier 2, …
    - ➡ Technology-independent C++, SQL-free API
  - Multiple replicas of the relational data
    - ➡ Indirection mechanisms
  - Use of shared database services
    - ➡ Flexible and secure authentication mechanisms
    - ➡ Smart client-side database connection management (fail-over, retrial, pooling,…)
    - ➡ Hooks for client-side monitoring

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

C++, SQL-free API (I)

Example 1: Table creation

```
coral::ISchema& schema =
session.nominalSchema();
coral::TableDescription tableDescription;
tableDescription.setName( “T_t” );
tableDescription.insertColumn( “I”, “long long” );
tableDescription.insertColumn( “X”, “double” );
schema.createTable( tableDescription);
```

Oracle

```
CREATE TABLE “T_t”      ( I NUMBER(20),
                          X BINARY_DOUBLE)
```

MySQL

```
CREATE TABLE “T_t”      ( I BIGINT,
                          X DOUBLE PRECISION)
```

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

C++, SQL-free API (II)

Example 2: Issuing a query

```
coral::ITable& table = schema.tableHandle( “T_t” );
coral::IQuery* query = table.newQuery();
query->addToOutputList( “X” );
query->addToOrderList( “I” );
query->limitReturnedRows( 5 );
coral::ICursor& cursor = query->execute();
```

Oracle

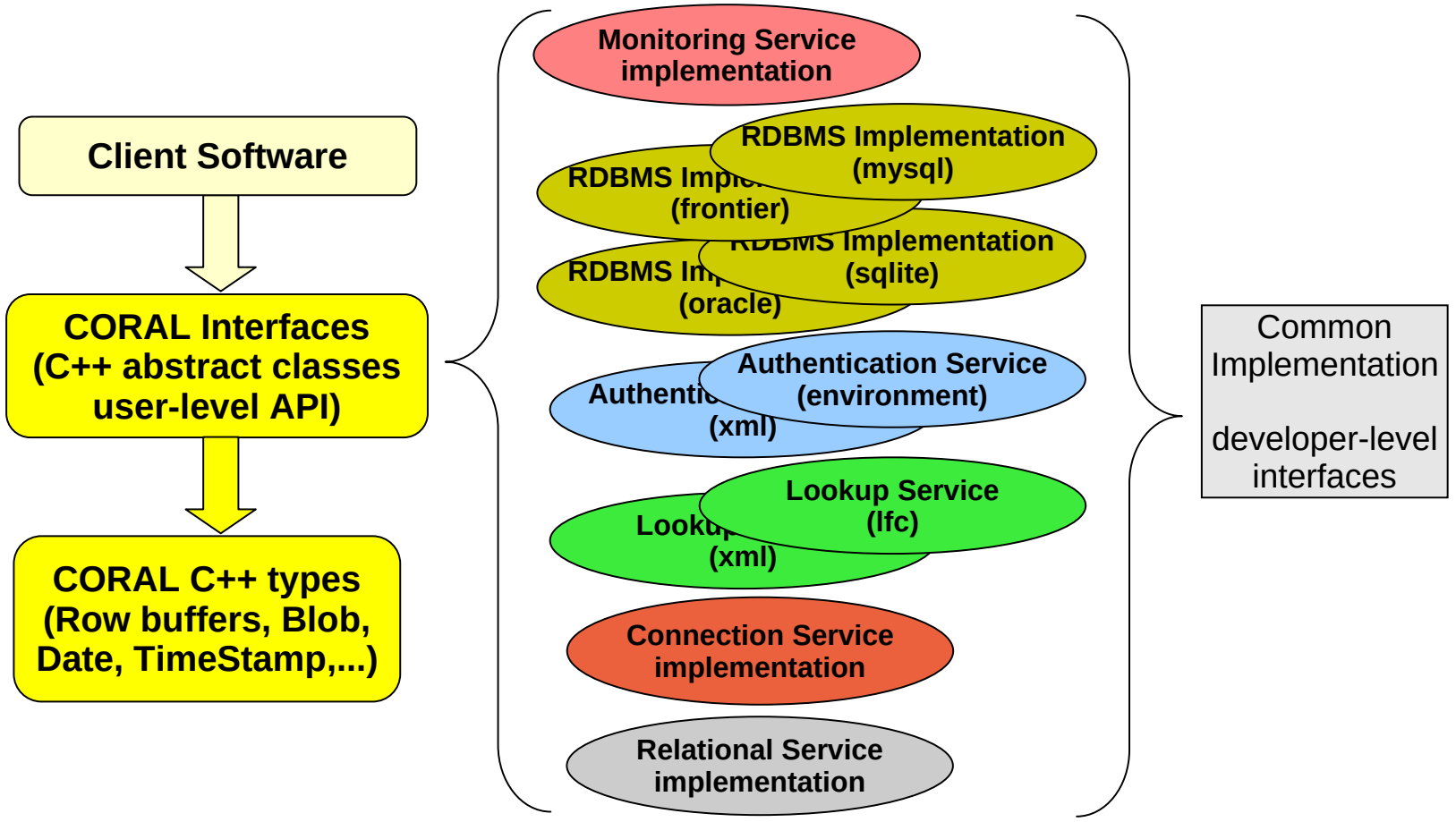
```
SELECT * FROM
      ( SELECT X FROM “T_t” ORDER BY I )
WHERE ROWNUM < 6
```

MySQL

```
SELECT X FROM “T_t” ORDER BY I LIMIT 5
```

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Component Architecture (I)



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

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Component Architecture (II)

- Interactions across the various models is based on the SEAL component model.
- Advantages of this architectural choice
  - Client code depends on a very thin and lightweight software stack
    - CoralBase (Data Types, Row Buffers, 195 KB)
    - RelationalAccess (API, Exception Hierarchy, 115 KB)
  - Efficient unit testing and bug tracking
  - Most releases are binary compatible to the previous one:
    - The core of the implementation is in run-time loadable components
    - So far 16 releases
      - All of them backwards compatible (no need for change in user code, incremental extensions of the API)
      - 11 of them fully binary compatible (no need even to recompile the user code)

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

RDBMS Functionality

- API Highlights
  - Schema definition/manipulation
    - multicolumn indices, keys, constraints, views
    - BLOB and variable-size string variables
  - Data manipulation
    - Use of “bind variables”: Forced/internally implemented (in row insertions) or encouraged by design (row updates/deletions)
    - “Bulk” operations
    - “INSERT-SELECT” operations
  - Queries
    - Row prefetching (client-side caching)
    - Nested queries (“subselects”)
    - …

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

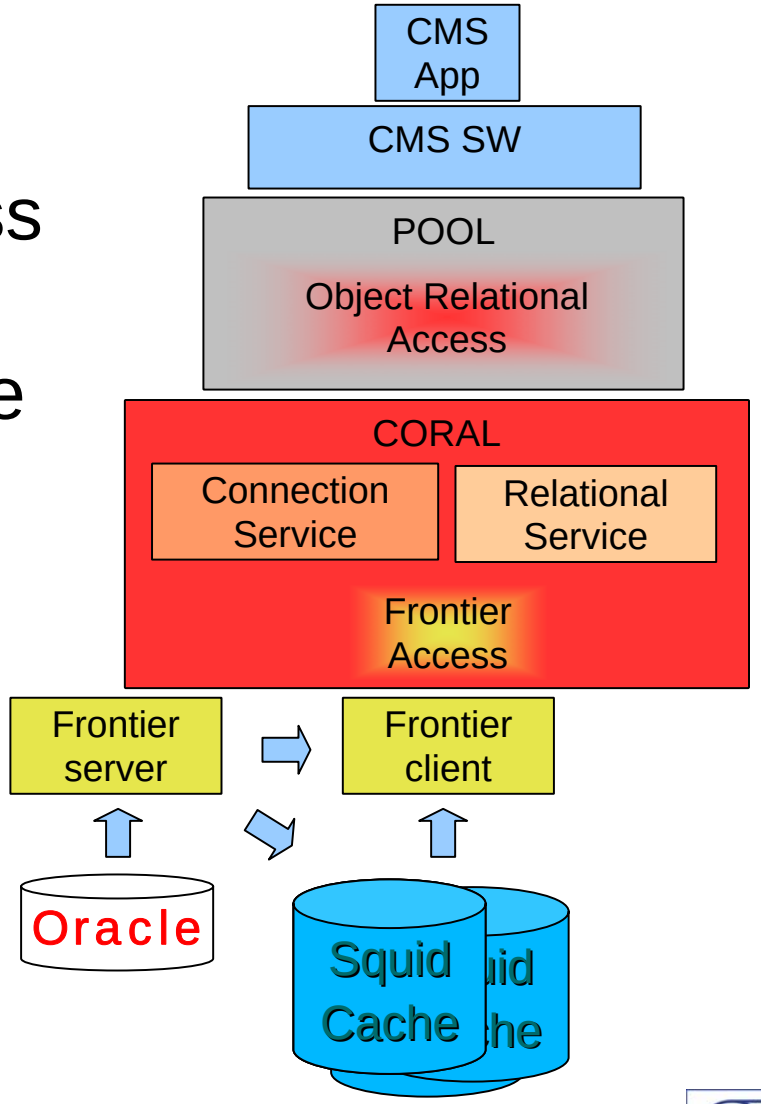
RDBMS Back-End Technologies

- Oracle
  - Fully implements the CORAL API and all internal optimizations:
    - Row prefetching, bind variables, server-side cursors,…
  - Based on OCI (C client library) 10.2.0.2
- MySQL
  - Better suited where only a low level of administration can be afforded
  - Based on the C client library version 5.0
- SQLite
  - File-based; no administration required
  - Based on the C client library version 3.3.5
- Frontier
  - Squid caches between client and Oracle database
  - Suitable for read-only data
  - Implies constraints on the data deployment model (as data may become stale)

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

FrontierAccess plug-in

- Co-developed with Fermilab/CMS
- Deployment in progress
  - 3D project, CMS
- Squid web proxy cache based data access for scalability
- Intensive round-trip development
- Still under active development
- Performance studies
  - CMS, ATLAS, COOL



CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Service Indirection (Lookup Service)

- A database service catalog
  - An “LFN”: */my/conditions/data*
  - A “PFN”/replica: *oracle://HostName/SchemaName*
- Single current implementation based on XML file
  - HTTP based access possible too
- A prototype implementation based on LFC
  - Developed in collaboration with RRCAT India
  - Released last week
  - No need for an extra service!
- A typical user never interacts directly with these components; simply specifies the logical connection string

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Authentication Mechanisms

- CORAL connection strings tell only about the location of a data source
  - mysql://HostName/DatabaseName
- Data source access credentials not exposed
  - UserName="Me", Password="ThEpAsSwOrD"
  - Credential parameters are retrieved from different software components given the bare connection string
- Two simple implementations currently:
  - Credentials from environment variables
  - Credentials stored in XML files
  - Used for application development and prototyping
- Secure implementation based on LFC is being developed
  - Authentication based using GRID certificates
  - Credentials linked to "database roles" and controlled by the LFC ACL mechanisms
  - Extension of the LFCLookupService
  - No need for an extra service (again…)!

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Client-Side Connection Handling

- ConnectionService component
  - Original implementation inherited from ATLAS
- Coordinates Lookup, Authentication, Relational and Monitoring services
  - Ensures that at any given moment a user has a valid connection handle to the schema described by the logical name
    - Connection re-trial and replica fail-over
- Minimizes resources on server from the client
  - Handles to read-only sessions on the same server share the same physical connection
  - Idle connections are pooled for a possible re-use
- Fully configurable parameters
  - To be largely determined by the service providers (DBAs)
- Extensive test plan for probing the behavior of the system
  - Under various imposed failures with an Oracle database:
  - http://pool.cern.ch/coral/internal/connectivity_test_plan.html

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Client-Side Monitoring

- Reasons for client-side monitoring
  - complements the server-side monitoring
  - assists application debugging and tuning
- Implementation strategy
  - CORAL API defines the interfaces for the call-back objects that are called by the RDBMS plugins
  - Information is pushed from the system to the monitoring implementation
    - Session and transaction time boundaries
    - Time duration that the client waits on the server to execute an SQL command
    - The SQL commands themselves
  - Current implementation serves as an example
    - Experiments are expected to implement their own plugins that are coupled to specific monitoring systems

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CORAL and DB Services

- Advantages (for the service providers) from CORAL-based applications
  - Best practices communicated to the development team and implemented centrally
  - Have to deal with a limited set of access patterns
  - Application system testing using the “integration service” makes deployment easier
  - Evolution of the database services and software can be quickly captured by (CORAL) developers and reflected by the rapid database access software upgrades
- Having the CORAL developers close to the service providers is COOL :-)

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CORAL in Use and Outlook

- CORAL users
  - ATLAS
    - Via POOL and COOL
    - Direct usage from detector geometry and on-line applications
  - CMS
    - Via POOL (RelationalFileCatalog, Object-Relational Access)
    - Direct usage from conditions database and on-line applications
  - LHCb
    - Via POOL and COOL
- Main priorities
  - LFC-based component for authentication
  - Focus on connection handling policies
  - Make CORAL thread-safe
  - Follow requirements from the 3D activities and implement the necessary solutions
  - Python interface to CORAL
    - In cooperation with RRCAT, India

CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it