CORAL Server

A middle tier for accessing relational database servers from CORAL applications

Andrea Valassi, Alexander Kalkhof (DM Group – CERN IT)
Martin Wache (University of Mainz – ATLAS)
Andy Salnikov, Rainer Bartoldus (SLAC – ATLAS)

LCG Application Area Meeting, 4th November 2009

Thanks for useful suggestions and contributions to Dirk Duellmann, Zsolt Molnar, the Physics Database team and all other members of the CORAL, POOL and COOL teams!

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

Outline

- **Motivation**
- **Development and deployment status**
  - Software architecture design
- **Outlook**
  - Planned functional/performance enhancements
- **Conclusions**

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

Introduction

- **CORAL used to access main physics databases**
  - Both directly and via COOL/POOL
    - Important examples: conditions data of Atlas, LHCb, CMS
  - Oracle is the main deployment technology at T0 and T1
    - Main focus of 3D distributed database operation
- **No middle tier in current client connection model**
  - Simple client/server architecture
    - No Oracle application server
  - One important exception: Frontier (for read-only access)
    - Used by CMS for years, adopted by ATLAS a few months ago
- **Limitations of present deployment model**
  - Security, performance, software distribution (see next slide)
  - *Several issues may be addressed by adding a **CORAL middle tier***
    - *Proposed in PF reports at the 2006 AA review and the 2007 LHCC*

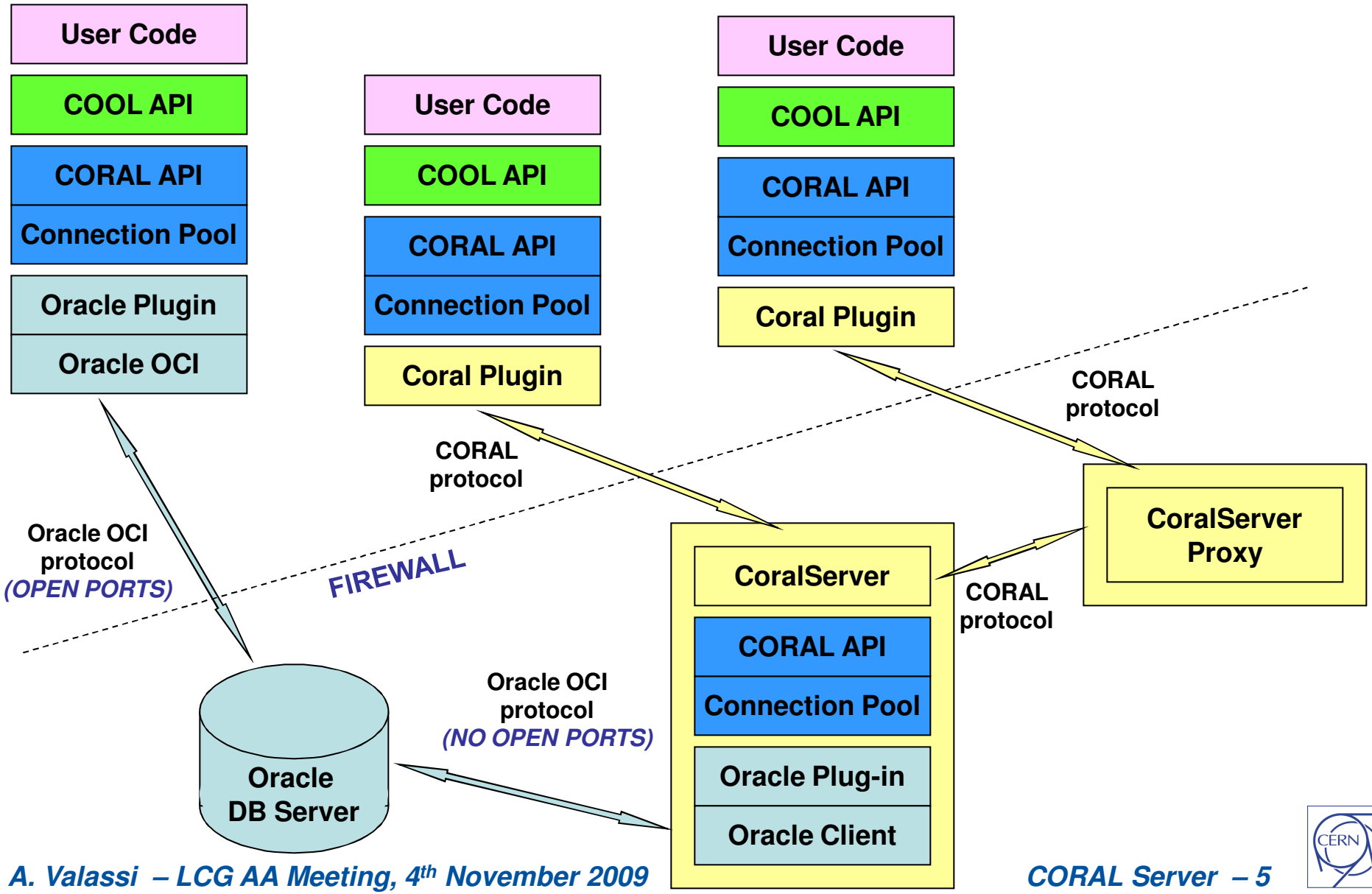
CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Motivation

- **Secure access (R/O and R/W) to database servers**
  - Authentication via Grid certificates
    - No support of database vendor for X.509 proxy certificates
    - Hide database ports within the firewall (reduce vulnerabilities)
  - Authorization via VOMS groups in Grid certificates
- **Efficient and scalable use of database server resources**
  - Multiplex clients using fewer physical connections to DB
  - Option to use additional caching tier (CORAL server proxy)
    - Also useful for further multiplexing of logical client sessions
- **Client software deployment**
  - CoralAccess client plugin using custom network protocol
    - No need for Oracle/MySQL/SQLite client installation
- **Interest from several stakeholders**
  - Service managers in IT: physics DB team, security issues...
  - LHC users: Atlas HLT (replace MySQL-based DbProxy)...

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

Deployment scenario (e.g. COOL)



Development overview

- **First phase of development in 2007-2008**
  - User requirements (Atlas HLT) analyzed in November 2007
  - Main focus of this development phase was on server components
    - Design based on template meta-programming techniques
  - Passed simple COOL R/O tests, failed HLT tests against COOL DB
- **New developments in 2009** ← ***Only focus of this talk***
  - Several changes in the team at the end of 2008
    - Software review in December 2008 to identify areas for improvement
    - Development restarted (from ~scratch) in January 2009 using a new comprehensive architecture design covering both server and client
  - Kept focus on R/O access with caching proxy for Atlas HLT
    - This is only the first customer – but is an excellent benchmark
  - Aggressive schedule led to successful deployment in only 9 months
    - March: pass standalone multi-client HLT tests on COOL DB
    - June: pass standalone tests on trigger and geometry DBs – first release
    - September: software deployed at Point1 and fully validated/operational

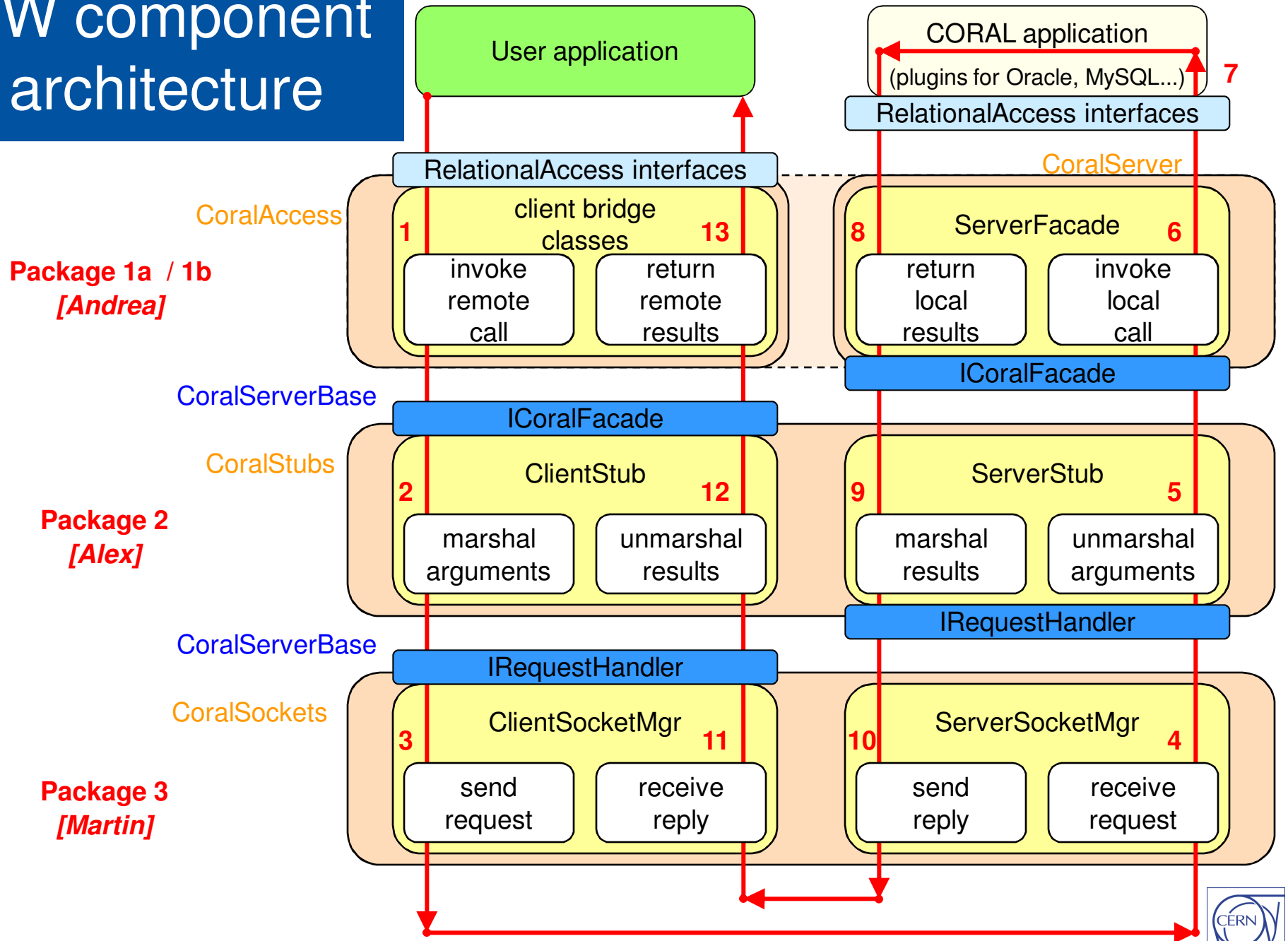
CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Guidelines of present design

- **Joint design of server and client components**
  - Split system into packages 'horizontally' (each package includes both the server-side and the client-side components)
    - Proxy is now standalone but can be reengineered with this design
  - RPC architecture based on December 2007 python/C++ prototype
- **Different people may work in parallel on different packages**
  - Minimize software dependencies and couplings
  - Upgrades in one package should not impact the others
- **Strong emphasis on functional (and performance) tests**
  - Include standalone package tests in the design
    - Aim to intercept issues before they show up in system tests
  - System tests may be decomposed back into incremental subtests
- ***Decouple components using abstract interfaces***
  - Modular architecture based on object-oriented design
  - Thin base package with common abstract interfaces
  - Implementation details encapsulated in concrete derived classes

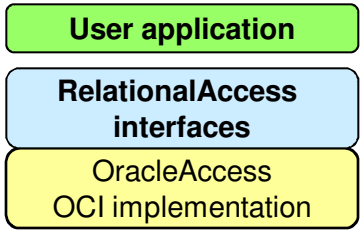
CERN - IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

SW component architecture

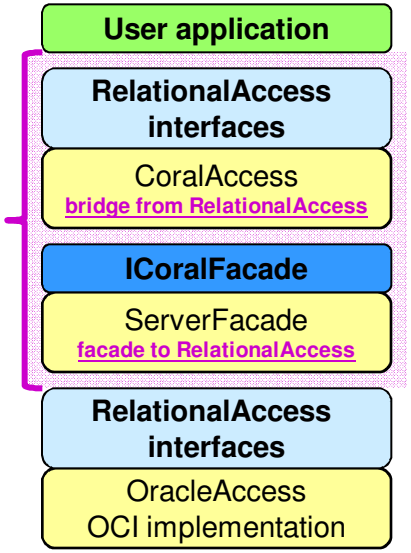


Incremental tests of applications

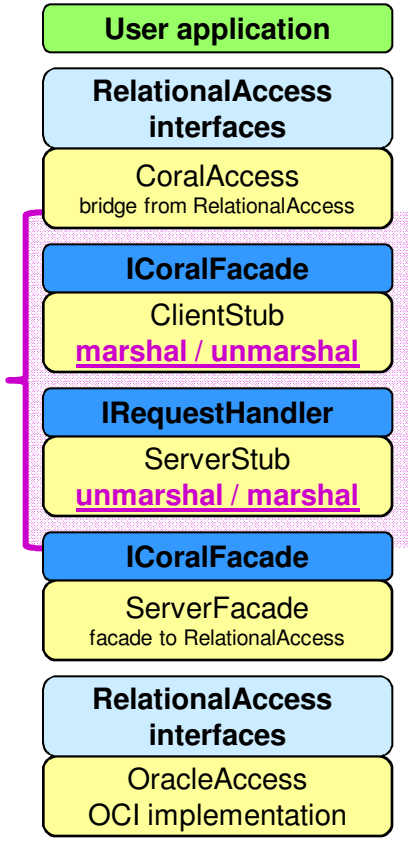
Traditional CORAL
"direct" (OracleAccess)



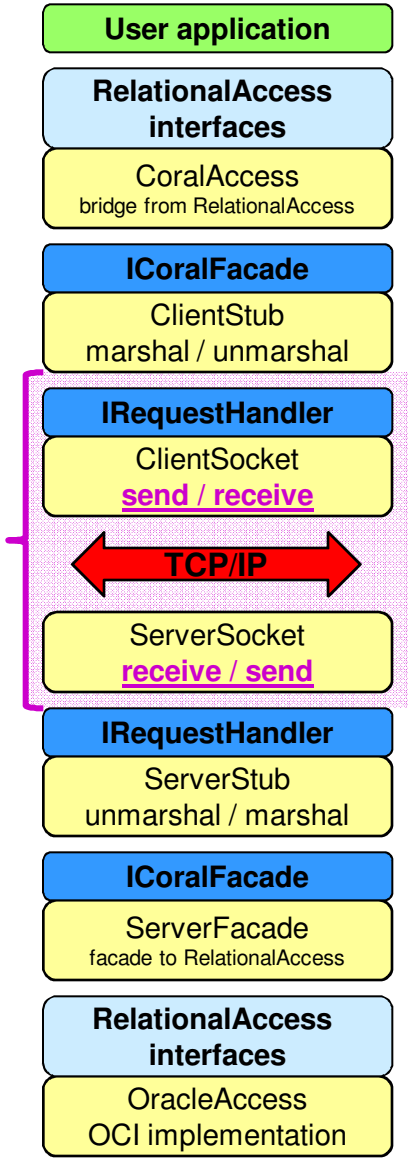
Add package 1 (a/b)
"façade only"



Add package 2
"stub + façade"



Add package 3
"server" (full chain)



A few implementation details

- **Server is multi-threaded**
  - Threads are managed by the SocketServer component
  - One listener thread (to accept new client connections)
  - One socket thread per client connection
    - Pool of handler threads (many per client connection if needed)
- **Network protocol agreed with proxy developers**
  - Weekly meetings (ongoing, now used for regular progress review)
  - Most application-level content is opaque to the proxy
    - Proxy understands transport-level metadata and a few special application-level messages (connect, start transaction…)
    - Most requests are flagged as cacheable: only hit the DB once
  - Server may identify (via a packet flag) client connections from a proxy and establishes a special ‘stateless’ mode
    - Session multiplexing (cache connect and drop disconnect requests)
    - One R/O transaction spans whole session (drop transaction requests)
    - ‘Push all rows’ model for queries (no open cursors in the database)

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

ATLAS HLT proxy specificities

- **Development has largely focused on ATLAS HLT so far**
  - Very complex, but also very simple, depending on the point of view
    - Some specific choices for HLT (mainly in the proxy layer) may need to be changed for the deployment of a general-purpose offline service
- **Oracle connection sharing was initially disabled**
  - Not necessary for HLT, thanks to session multiplexing in the proxy
    - But necessary for connection multiplexing in general-purpose server
  - Hang had been observed with connection sharing
    - Problem identified as Oracle bug in MT mode and solved in 11g client
- **HLT deployment needs non-serializable R/O transactions**
  - Requirement: see data added after the start of the R/O connection
    - Connections are potentially long
  - Presently handled by a 'hidden' env variable in OracleAccess
    - May need a cleaner way out (API extension: three transaction modes)

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

Outlook

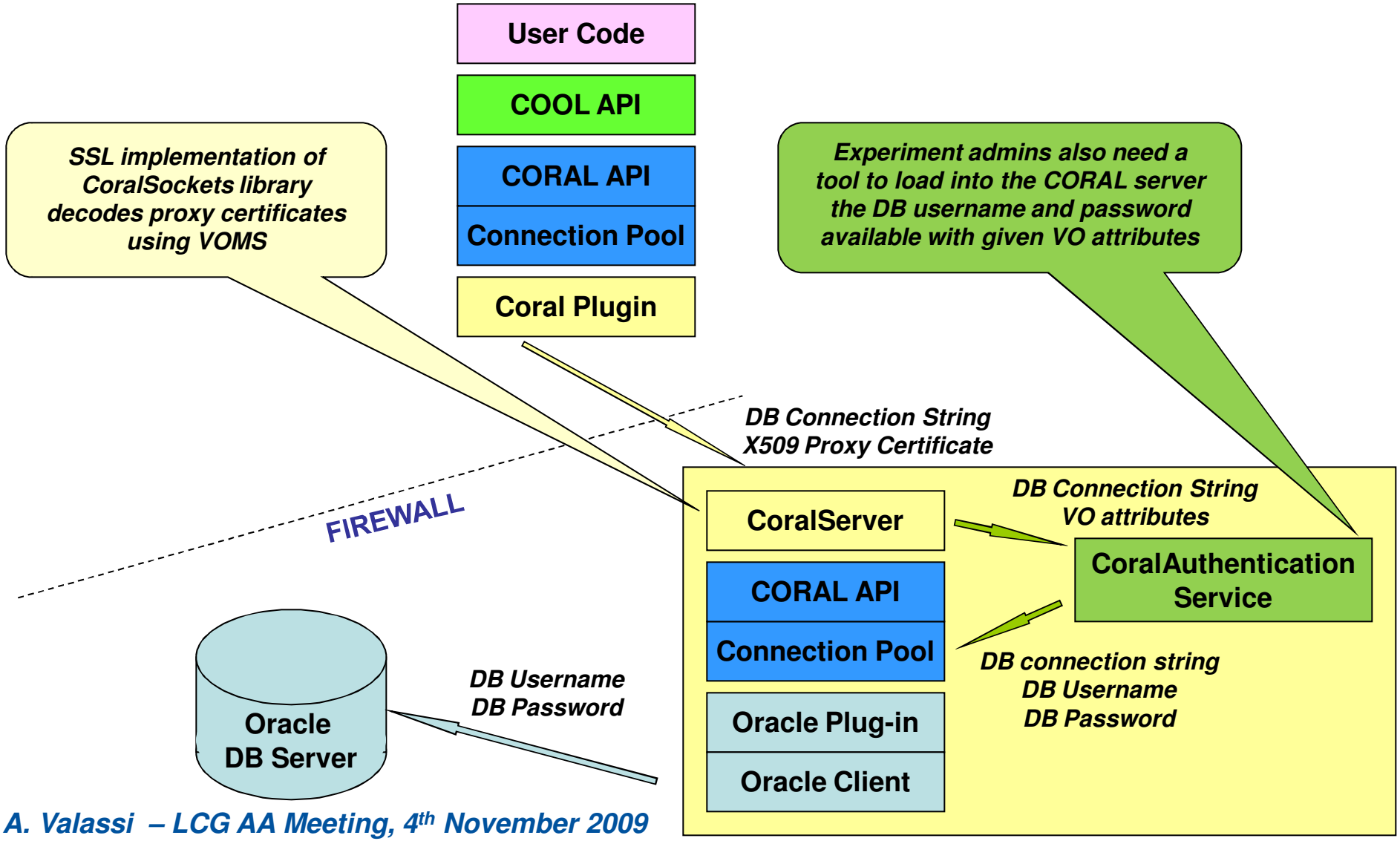
- **Software deployed and validated for Atlas HLT – Q3 2009**
  - Read-Only access with passwords (single authentication.xml file)
- **Complete secure authentication/authorization *– Q4 2009?***
  - Proxy certificates supported in client/server – but not yet released
    - Dependencies on SSL and VOMS versions must be sorted out
    - SSL missing in proxy – might switch to common components
    - Separate SSL and non-SSL sockets for control and data packets
  - In progress: tool to load Oracle passwords into the CORAL server
- **Monitoring enhancements for ATLAS HLT *– Q1 2010?***
  - Need at least equivalent features as M2O and DbProxy monitoring
- **Further performance tests and optimizations *– Q1 2010?***
  - Measure and minimize the overhead w.r.t. Oracle direct access
  - Later: add disk cache to proxy for possible offline usage
- **Full Read-Write functionalities *– Q1 2010?***
  - DML (e.g. insert table rows) and DDL (e.g. create tables)
- **Deployment of test server and proxy at CERN *– Q2 2010?***
  - In parallel, discuss possible deployment at T1 sites

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

Secure access scenario

CERN IT Department

Thanks to A. Frohner, J.P. Baud, V. Ciaschini, M. Alandes, S. Roiser for help with SSL and VOMS!



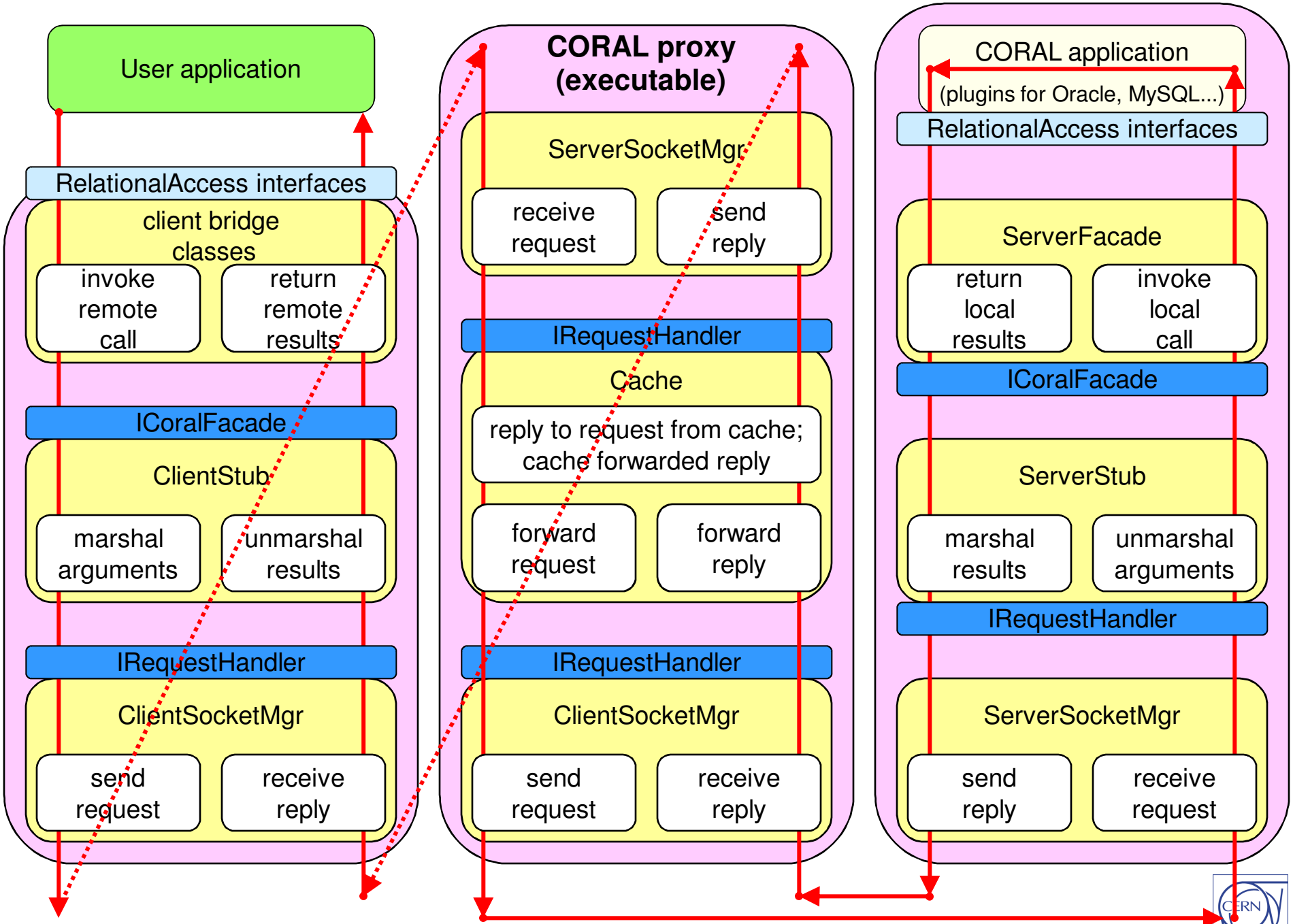
Conclusions

- **CoralServer is validated and fully operational for Atlas HLT**
  - Only 9 months after start of developments with present design
  - Read-Only access with passwords, plus caching proxy
  - Also adopted for Global Monitoring at Atlas Point1
    - Interest for Read-Write access at Point1 too
- **Several developments are planned in the future**
  - Secure authentication, Read-Write access, Monitoring…
  - Precise timescales for achieving them are hard to predict
    - Manpower is limited and needed also for mainstream PF support tasks
- **Mutually beneficial collaboration of CERN/IT and ATLAS**
  - Effective and very pleasant too: thanks to everyone in the team!
  - I believe this software will be useful for other experiments too

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

Reserve slides

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


User application
RelationalAccess interfaces
client bridge classes
invoke remote call
return remote results
ICoralFacade
ClientStub
marshal arguments
unmarshal results
IRequestHandler
ClientSocketMgr
send request
receive reply
CORAL proxy (executable)
ServerSocketMgr
receive request
send reply
IRequestHandler
Cache
reply to request from cache; cache forwarded reply
forward request
forward reply
IRequestHandler
ClientSocketMgr
send request
receive reply
CORAL application
(plugins for Oracle, MySQL...)
RelationalAccess interfaces
ServerFacade
return local results
invoke local call
ICoralFacade
ServerStub
marshal results
unmarshal arguments
IRequestHandler
ServerSocketMgr
send reply
receive request