LCG 3D PROJECT STATUS AND PRODUCTION PLANS

L. Albino Nogueira Ramos, L. Betev, L. Canali, J. Casey, R. Chytracek, M. Clemencic,
E. Dafonte Perez, G. Dimitrov, F. Donno, D. Duellmann, M .Frank, M. Girone, B. Koblitz,
P. Kunszt, S. Lemaitre, P. Mato, I. Papadopoulos, N. Sinanis, CERN, Geneva, Switzerland
A. Vaniachine, ANL, USA
S. Stonjek, J. Tseng, Oxford, UK
A. Formica, DAPNIA, Saclay, France
M. Case, U Davis, USA
Z. Xie, Princeton University, Princeton, USA
Meng-Hang Ho, Chi-Wei Wang, Jeng-Hsue Wu, Min-Hong Tsai,
Tao-Sheng Chen, ASGC, Taipe, Taiwan
S. Misawa, R. Popescu, G. Tsai, Y. Smirnov, Y. Wu, BNL, Brookhaven, USA
G. Brown, N. Cullen, A. Sansum, RAL, Didcot, UK
B. Martelli, E. Vilucchi, CNAF, Italy
S. Iqbal, A. Kumar, L. Lueking, J. Trumbo, E. Wicklund, FNAL, Batavia, USA
S. Halstenberg, R. Kupsch, H. Marten, D. Wochele, GridKA, Karlsruhe, Germany
P. Macchi, J.R. Rouet, IN2P3, Lyon, France
G. Merino, M. Rodriguez, PIC, Barcelona, Spain
J. Templon, R. Trompert, A. Verkooijen, NIKHEF/SARA, Netherlands
R. Gardner, A. Zahn, U Chicago, USA
D. Deatrich, R. Tafirout, TRIUMF, Canada
O. Smirnova, A. Waananen, NDGF
A. Domenici, L. Iannone, G. Pucciani, INFN, Pisa, Italy

Abstract

The LCG Distributed Deployment of Databases (LCG 3D) project is a joint activity between LHC experiments and LCG tier sites to co-ordinate the set-up of database services and facilities for relational data transfers as part of the LCG infrastructure. The project goal is to provide a consistent way of accessing database services at CERN tier 0 and collaborating LCG tier sites to achieve a more scalable and available access to non-event data (e.g. conditions, geometry, bookkeeping and possibly event level meta data). Further goals include the co-ordination of the requirement discussions between sites and experiments and to facilitate the technical exchange between database service providers at online, tier 0 and tier 1 sites. This contribution describes the outcome of the first year of requirement gathering and technology tests with the proposed distribution technologies (Streams and FroNtier). We also give a summary the definition of the production set-up for the first 6 months of operation as part of LCG service challenges.

INTRODUCTION

The LCG 3D project [2] has been started in the context of the LHC Computing Grid (LCG [1]) to define a consistent database service setup for experiment applications and grid services, and to investigate distribution techniques to make data stored in relational databases available in a consistent way at the participating sites.

The project has focussed during the first year on the integration of database setups and service teams at tier 0 and tier 1 sites and on the evaluation of distribution technologies for HEP applications. This work is related also to significant preparations on the grid and experiment application software side, which are not covered here but in other contributions to this conference [7,8,9,10,11,12,13,14]

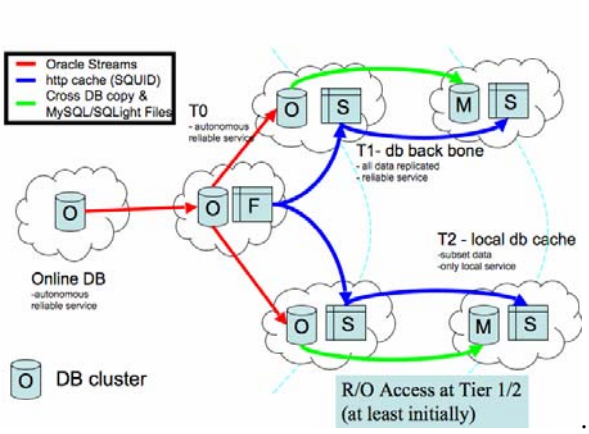

Figure 1: LCG 3D Service Architecture

The service architecture proposed by the project, summarised in Fig 1, deploys database clusters as basic building blocks for the database service at CERN and tier 1 sites.

Database Clusters

Oracle Real Application Clusters (RAC [3]) promise to provide the technology for building database clusters that provide higher availability than single node databases and at the same time allow scaling the server CPU with the application demands. By adding nodes to a cluster, the number of queries and concurrent sessions can be increased together with the total amount of database cache memory, which Oracle RAC shares across all cluster nodes.

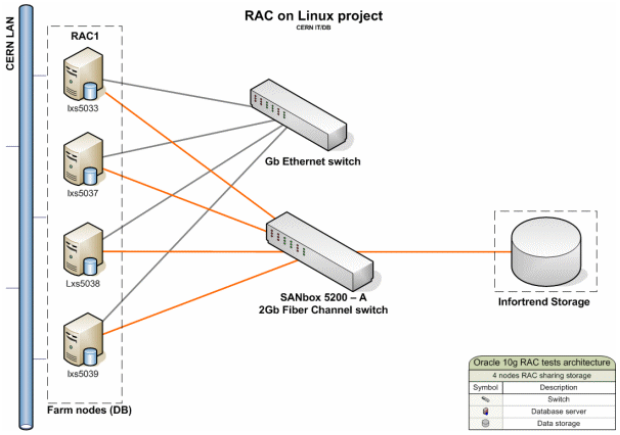

Figure 2: Database Cluster and SAN-based storage

How far a RAC set-up will be able to scale for a given application depends significantly on the application design. Limiting factors are typically inter-node network communication (cache coherency) and application contention on shared resources, which need to be identified in validation tests and can often be avoided by application design. To control these scalability limitations, a close interaction between application developers and database administration team is required especially during the early deployment phase.

In a RAC set-up the Oracle 10g 'service' concept allows one to structure larger database clusters into groups of nodes that are allocated to particular database applications (e.g., online configuration DB, Grid file catalogue). This pre-allocation of cluster resources can be used to limit inter-node communication and to isolate key applications from lower-priority tasks.

In addition to CPU scalability, RAC systems do also provide increased availability. In case of unavailability of a server node (e.g., because of a hardware or software problem or a planned service intervention) the software will redirect incoming client connections automatically to other cluster nodes. Open transactions may still be affected in case of a node failover and will be rolled back. This needs to be taken into account in the retry logic of database applications. An additional significant availability advantage of database cluster setups results from the possibility to perform database software upgrades in a rolling fashion, without requiring an outage of the complete service.

The server nodes in the CERN production set-up consist of mid-range dual CPU machines under Linux, to achieve cost efficiency and integration into the existing fabric infrastructure. The database nodes are connected to a shared storage system based on fibre channel attached disk arrays as shown in Fig 2. This fulfils the requirements of the 'shared-everything' architecture of Oracle and allows scaling of the storage system and CPU requirements independently.

Distribution Techniques

To connect database clusters and propagate data in the distributed setup several alternative approaches exist, which provide different levels of redundancy against network problems and scalability with the transferred data volumes.

For smaller data volumes and in reliable network environments direct synchronous data copying can be perfectly acceptable – e.g. via database links and materialised views. For larger volume read-only data with well-defined physical clustering the exchange of complete tables or tablespaces as files may be an option (e.g. Oracle transportable tablespaces, SQLight or MySQL files).

The general case though is more difficult to handle, as database data may be updatable (at least to one tier) or may contain complex relations between different database schemas or even servers. In these cases database replication as offered by Oracle streams technology [4] can maintain consistency between relational data at different tiers - online, offline (tier 0) and tier 1.

Database Replication with Oracle Streams

The Oracle streams [4] product allows connecting single tables or complete schemas in different databases and keeping them up to date. A schematic description of a streams setup used in the 3D testbed is shown in Fig. 3. Streams utilise the database redo logs to capture data or schema changes on the source database side - in this case the CERN tier 0 database. These changes are then shipped via an asynchronous propagation process to one or more destination databases, e.g. the tier 1 databases. At the destination the logical change records (LCRs) are re-applied in correct transactional order. Complex filter rules can be introduced to select which data and change types should be propagated.

In case a destination database is not reachable (e.g. because of a network outage or database service intervention) change records are kept on the source system and are automatically applied once the connection has been re-established.

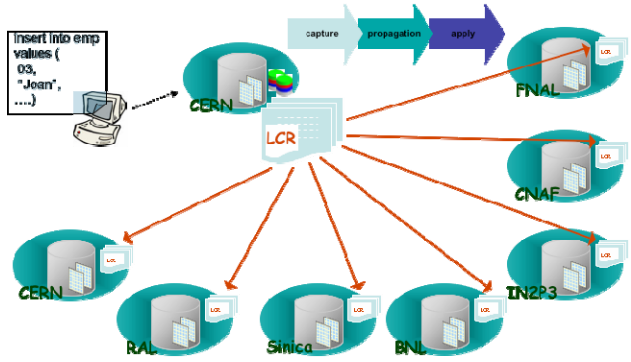

Figure 3: Database Replication with Oracle Streams

The database process which is capturing and queuing changes can optionally be executed on a separate machine (see also Fig 4.) to minimise the impact of the capture process on the source database. Streams can be setup to provide a uni-directional or bi-directional connection between their endpoints. Even though bi-directional streams have been tested successfully, they add significant complexity to the deployment as conflicts between updates on both streams endpoints may arise and need to be handled.

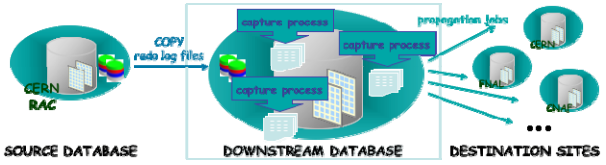

Figure 4: Downstream Capture Setup at Tier 0

A key advantage of streams with respect to other mechanism such as application specific copy tools or data caching is their simple and generic semantics. Many key LCG database applications have been validated in the streams environment and continue to function without any application changes or application specific replication procedures.

Distributed Caching with FroNTier/Squid

The second data distribution technique that has been evaluated is based on read-only caching of query results. Instead of providing a connected set of distributed database servers here a distributed hierarchy of independent cache servers is introduced between database clients and a (central) database server.

In order to deploy a standard web proxy cache server (Squid [6]) to cache database data, the FroNTier package is used to encode the communication between database client and server (e.g. select statements and database result sets) into http requests. A FroNTier plug-in to the CORAL [5] database abstraction component of the LCG Persistency Framework (POOL) has been developed, which maps SQL queries to particular URL request and extracts query result from html documents prepared by the FroNTier server.

The key advantage of the FroNTier/Squid system is its lower administration requirement in comparison to a full database server deployment. As squid servers are easier to configure and maintain, they may simplify the provision of a local database cache with the more limited human resources available at tier 2 sites.

Even if in this approach all database queries need to be performed upstream, currently centrally at tier 0, the setup still scales well for clients that use repeatedly the same queries on the same read-only database tables. These query-data combinations will found in a local cache (either directly in a tier 2 cache or perhaps in the larger cache of the nearest tier 1) and will not result in a real database query.

As squid cache servers are not aware of possible database updates, applications running in a FroNTier/Squid environment need to be carefully designed and tested to avoid possibly subtle consistency issues caused by stale cached data.

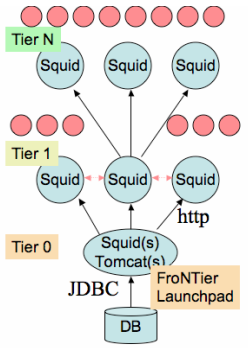

Figure 5: Distributed Caching with FroNTier/Squid

Replication Tests in the 3D Testbed

To test experiment and grid database applications against different replication techniques a testbed of database, FroNTier and squid servers has been setup between CERN and LCG tier 1 sites. This included two database replicas at CERN, which have been connected to experiment online and tier 1 servers. The list of applications and involved tiers is given in Table 1.

Table 1: Application Tests in the 3D Testbed

Experiment / Project	Software Package	Distribution Tiers	Distribution Technique
ATLAS	COOL	Online->T0	Streams
ATLAS	COOL	T0->T1	Streams
ATLAS	AMI	T1->T0	Streams
CMS	Conditions	Online->T0	Streams/ Copy
CMS	Conditions	T0->T1->T2	FroNTier/ Squid
LHCb	COOL	T0->T1	Streams
LCG	AMGA	T0->T0	Streams
LCG	LFC	T0->T1	Streams/ Copy

Production Setup and Schedule

To prepare the production deployment of LCG database services the project has collected estimated database requirements for the first year of production from experiments and grid projects. These estimates still suffer significant uncertainties as full-scale production experience is still missing and in particular the definition of concrete calibration models still evolves as part of the development of detector calibration algorithms.

The project has therefore proposed to start in parallel to the LCG Service Challenge 4 in April 2006 with a pre-service phase with 2-3 node database clusters at seven LCG sites (ASCC, BNL, CNAF, CERN, GridKA, IN2P3, RAL). These database setups have been successfully commissioned by the site database teams and can now be used for a larger scale replication throughput tests by the ATLAS and LHCb experiments. In addition a redundant and load-balanced three-node FroNTier/Squid production system has been setup at CERN, which will be exercised by the CMS experiment that during this initial phase focus on FroNTier/Squid deployment for Tier 0/1/2 distribution. The remaining LCG tier 1 sites (PIC, NIKHEF/SARA, NSG, TRIUMF) have now joined the project to prepare for a full production phase including all sites scheduled for October 2006.

Summary

The 3D project has proposed a service implementation for distributed database deployment in the LHC computing grid, which relies on database clusters for the central tiers (online, T0, T1) to provide the required availability and to scale with still uncertain experiment requirements.

Two complementary database distribution techniques have been evaluated: Database replication via Oracle streams is used for online to offline transfers. Towards the more external tiers (tier 1 and 2) distributed query caching via Frontier/Squid may become an important alternative, which promises to require less deployment effort at the sites. Both techniques have been successfully used in wide area distributed tests and showed promising performance and deployment stability in small to medium scale tests.

The project is now moving into pre-production phase and a significant fraction of the requested database resources have been setup by the participating LCG sites. This allows also large-scale validations by the experiments including a comparison of the wide-area distribution approaches. The outcome of this phase will be used to adjust the current experiment requirements for a first full database service phase which is scheduled for October 2006.

REFERENCES

[1] The LHC Computing grid project http://lcg.web.cern.ch

[2] Distributed Deployment of Databases for LCG - http://lcg3d.cern.ch

[3] http://www.oracle.com/technology/products/database/clustering

[4] http://www.oracle.com/technology/products/dataint/htdocs/streams_fo.html

[5] LCG Persistency Framework, http://pool.cern.ch, http://pool.cern.ch/coral

[6] Squid web proxy cache - http://www.squid-cache.org/

[7] LHCb conditions database framework, CHEP 06, #168, M. Clemencic et al.

[8] Database access in Atlas computing model, CHEP 06 #38, A. Vaniachine et al.

[9] Software for a variable Atlas detector description, CHEP 06, #67, V. Tsulaia et al.

[10] Optimized access to distributed relational database system, CHEP 06, #331, J. Hrivnac et al.

[11] COOL Development and Deployment - Status and Plans, CHEP 06, #337, A.Valassi et al.

[12] COOL performance and distribution tests, CHEP 06, #338, A. Valassi et al. (poster)

[13] CORAL relational database access software, CHEP 06, #329, I. Papadopoulos et al.

[14] POOL object persistency into relational databases, CHEP 06, #330, G. Govi et al. (poster)