Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering

Study and testing of the Grid infrastructure for the data processing in the ALICE project at Large Hadron Collider at CERN

Prague 10.1.2009 Čeněk Zach

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne ______________ ____________________

podpis

Poděkování

Chtěl bych poděkovat paní RNDr. Dagmar Adamové CSc. za konzultace a velikou pomoc při psaní této práce.

Název práce: Studium a testování gridové infrastruktury pro zpracování dat projektu ALICE na Large Hadron Collider v CERN

Autor: Čeněk Zach

Obor: Jaderné Inženýrství

Zaměření: Experimentální Jaderná Fyzika

Druh práce: Bakalářská práce

Vedoucí práce: RNDr. Dagmar Adamová CSc., Ústav jaderné fyziky, AV ČR

Abstrakt: V mé práci nastíním aktuální problémy zpracování dat ve vědeckých projektech týkající se potřeby výpočetních a úložných zdrojů a jejich řešení s pomocí výpočetních gridů. Popisuji základní koncept výpočetních gridů a jejich architekturu se zaměřením na výpočetní grid LHC (LCG) a middleware AliEn. Zmiňuji hierarchickou architekturu, služby a aktuální zdroje LCG. Poté se soustředím na architekturu a služby AliEnu. Nakonec prezentuji krátké testování základních funkci AliEnu.

Klíčová slova: Výpočty ve fyzice vysokých energií, výpočetní Grid, LCG, AliEn

Title: Study and testing of the Grid infrastructure for the data processing in the ALICE project at Large Hadron Collider at CERN

Author: Čeněk Zach

Field: Nuclear Engineering

Specialization: Experimental Nuclear Physics

Supervisor: RNDr. Dagmar Adamová CSc., Nuclear Physics Institute, ASCR

Abstract: In my work I present current problems in the scientific data management concerning the need for computation and storage resources and their solution with the help of Computing Grids. I describe the concept and architecture of the Computing Grids with a focus on the LHC Computing Grid (LCG) and the Grid middleware AliEn. I describe the LCG tiered architecture, services and actual resources. After that I concentrate on the architecture and services of AliEn. In the end I present a short testing of some Alien basic functions.

Key words: Computing in High Energy Physics, Computing Grid, LHC, AliEn

Contents

List of Tables

1 Introduction

The High Energy Physics (HEP) experiments such as the experiments at the CERN Large Hadron Collider (LHC) [1] are heavily dependent on the computing resources with the storage and analysis of their data. Given the volume of data taken yearly at the LHC (about 15 PetaBytes), it is impossible to store and analyze all the data at any single computing centre alone. Instead, the infrastructure of a Computing/Computational Grid is necessary to cover the needs of the data management for these projects.

To put things easy lets say, that computers were invented to help people perform difficult calculations, and ever since the first prototype, they have been improving at incredible speed. The same story goes with the Computational Grid, which represents the idea of connecting some machines together to gain enhanced computing power, and then connecting computers all around the world to offer computing power to everyone, similar to electricity or water supply systems.

In the end of the last century, together with the technological progress in computing, also telecommunications and networking technologies were developed at impressive speed. In the early 1990s, the Word Wide Web started at CERN, to make easier exchange of documents among particle physicists around the world. Nowadays physicists and other scientists need to run very massive calculations which involve enormous amounts of data. The concept of Computing Grids brings about the possibility to cover the computational demands of these scientific projects. One can see the analogy: while the Word Wide Web facilitates informations and documents sharing, the basic goal of the Computing Grids is to allow sharing of computing resources owned by many different organizations to access remote computers, storage systems, software and data.

In this paper, I present a short review of the basic features of computing grids, with a special concentration on the LHC Computing Grid (LCG) [2] and the ALICE experiment computing grid in particular. Also, I give a brief example of testing a part of this infrastructure.

2 Computing Grid

"A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities." [3]

Let us examine in detail individual terms in this definition:

hardware infrastructure already exists and is continuously improved to deliver a high performance computing system, the ultimate goal is to unite it into one world-wide entity

software infrastructure consists basically of a set of protocols and services needed to operate the grid, so called middleware, and from software packages and applications developed for the projects using the grid.

dependable - high-level availability and reliability of the services must be ensured to make Computing Grids dependable as the power grid or the water supply

consistent - the grid (architecture, interfaces, etc.) in all parts of the world should me as alike as is possible

pervasive in any place on earth technically possible allowing practicaly everyone to use it

The main goal of Computing Grids is the coordinated resource sharing. By resource is meant computing power, storage capacity, data, software, advanced sensors, networks or any other possible components shared among any potential users.

To characterize different users of computing grid resources, the concept of the Virtual Organization (VO) was introduced [4]. The VO identifies a set of individuals and/or institutions defined by grid resources sharing rules and usually by a common purpose of the grid usage.

A natural tendency of already established grid systems is upscaling and extending of resources. This may be accompanied by many problems. How to effectively manage so many resources spread so far apart? How to take care of the vast number of users, who may use different operating systems (OS) on different platforms? What about security? All this and more of alike problems must be solved to make a computing grid available for users world-wide. The complex requirements given by the interoperability of computing grids motivated the formation of a complex technology called generally the Grid Architecture. More specifically, the term middleware is used for the system rules and services which operate a computing grid.

There exist a number of computing grids serving different projects and as a rule, each of them is developing project specific components of the grid middleware. Let us mention as some examples The Legion Project (discontinued since 2003) [5] or the Globus Project, now the Globus Alliance (GA) [6]. Different projects have their own approach to the building of middleware. While the former has implemented the grid middleware as object-oriented structure where each resource is one object and communicates with others through its methods, the latter developed the Globus Toolkit (GT) [7]. The GT has been designed to use (primarily) existing components (protocols, services, interfaces and software development kits), including vendor-supplied protocols and interfaces. However, if a vendor does not provide the necessary behavior, the GT includes the missing functionality.

2.1 Grid architecture

As mentioned earlier, the complexity of problems arising during the establishment, management and operation of computing grids initiated development of a complex technology generally identified as the Grid Architecture. Due to the distributed and networked nature of the grids, it consists mostly of protocols, defining rules of communication.

Communication is an important word here given the diversity of different Virtual Organizations that are entitled to use the grid services and resources. It would be unacceptable to ask individual institutions to give up any part of control of their resources or to require changes to local policies. The Grid Architecture must enable dynamic, cross-organizational sharing of grid resources. The resulting structure must be interoperable, which in the networked environment means common protocols.

A definition of the Grid Architecture provides description of individual components, the specification of the purpose and function of all the components, and indicates how these components interact with one another. The usual approach is, to define the Grid Architecture in terms of a system of layers, see Figure 1.

Components within each layer have common features which are built on capabilities of lower layers. Any layer can communicate only with a layer above or below within a given system and only with the same layer of another system. The architecture is usually represented in terms of the Hourglass Model where the narrow neck part consists of a small set of base protocols (e.g. TCP and HTTP), to which different high-level services can

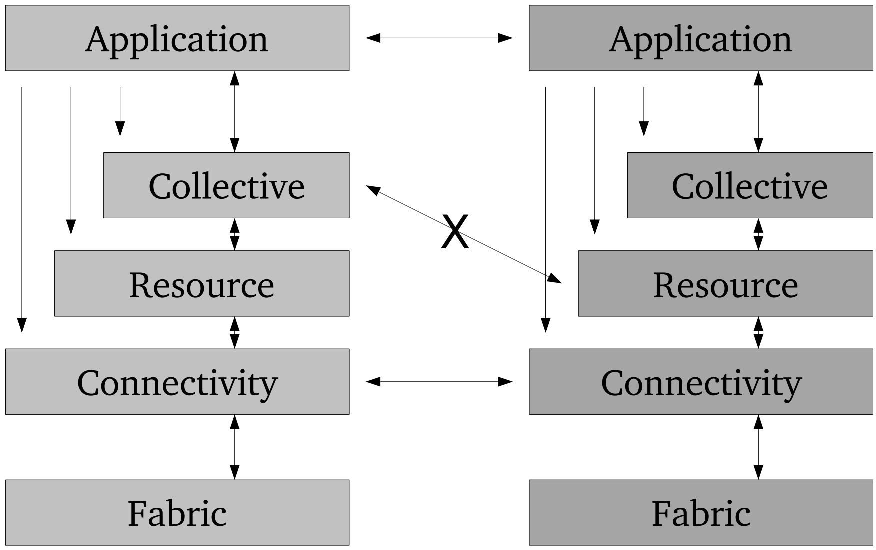

Figure 1: Layered Grid architecture, it is shown that layers build on capabilities of lower layers and can communicate only with layers one level higher or lower or the same layer of another system

refer (the top of the hourglass), and which themselves can be mapped onto underlying technologies (the base of the hourglass).

The neck protocols represent Resource and Connectivity protocols, which provide the sharing of individual resources. They can be implemented on top of resource types, defined at the Fabric layer, and can be used to construct global services and applications at the Collective layer.

2.1.1 Description of layers

Fabric layer provides the resources available for shared access by Grid protocols: for example, computational resources, storage systems, catalogs, network resources, and sensors. There are also internal protocols and interfaces such as computer cluster management protocols. These protocols/interfaces do not belong directly to the Grid architecure but may be used for the implementation of higher-level protocols.

Connectivity layer defines core communication and authentication protocols required for Grid-specific network actions. The communication mostly relies on the protocols derived from the Internet TCP/IP protocol stack. For authentication, communication protection, and authorization, the Grid Security Infrastructure (GSI) protocols are used. The GSI derives from the cryptographic protocols that provide secure communications on the Internet.

Resource layer defines protocols and interfaces for the secure negotiation, initiation, monitoring, control, accounting and payment. These protocols are built on Connectivity layer communication and authentication protocols. Some examples include: Grid Resource Information protocol (GRIP), the HTTP-based Grid Resource Access and Management protocol (GRAM), an extended version of the File Transfer Protocol - GridFTP and the Lightweight Directory Access Protocol (LDAP), used as a catalog access protocol.

Collective layer contains protocols, services and interfaces, that are not associated with any one specific resource but are global in nature. Some examples include:

- Directory services (Grid Resource Registration Protocol (GRRP) and Grid Information Service (GRIS) protocols)
- Co-allocation, scheduling, and brokering services (gLite Resource Broker (RB))
- Monitoring and diagnostics services (Caltech MonALISA package)
- Data replication services (the Globus Data Replication Service (DRS))
- Workload management systems (the gLite Workload Management System (WMS))
- Grid Information Index Servers (GIISs)
- online credential repository service (MyProxy)

An example of a support service for a Grid users is the Global Grid User Support (GGUS) serving the users of WLCG/EGEE projects [8].

Application Layer in the Grid architecture consists of the user applications that operate on the Grid. Applications are constructed utilizing services defined at any layer. In this context, let's mention the Applications Area and Distributed Analysis & Grid Support projects of the LHC Computing Grid [2], that provide the tools and infrastructure for the development and interfacing of the High Energy Physics (HEP) software.

2.1.2 Description of protocols/services

It's not necessary to discuss here all the needed protocols resp. services so I'll pick some important of them and shortly describe them:

Grid Security Infrastructure protocols (GSI) are based on Transport Layer Security TLS thus on public keys

Grid Index Information Service (GIIS) has all the static informations about resources - which, where, how many, capacity, etc.

Grid Resource Registration Protocol (GRRP) serves for registering single resources to GIIS

Grid Resource Information Protocol (GRIP) returns static/dynamic information about requested resource, for example a load of CPU or free resp. maximum capacity of storage element

Grid Resource Access and Management (GRAM) provides tools for allocation, monitoring and control over computational resources

Grid File Transfer Protocol (GridFTP) is extension FTP which implements use of security protocols, partial file access and others

Compute Resource Service/Computing Element (CE) is a grid-enabled computing resource [9]

CE should provide:

- mechanism to submit and monitor jobs
- information about resources and their current state through GRIS
- authentication and authorization mechanisms
- others - depending on concrete definition

Storage Resource Services/Storage Element (SE) is a grid-enabled storage resource [9]

SE should provide:

- Mass Storage System (MSS)
- Storage Resource Manager (SRM) interface that provides the procedures for storing, registration, access, download and removal of files together with security, authentication and authorization services

Workload Management System (WMS) is one of the core grid services made up of a set of components responsible for the distribution and management of tasks across the Grid resources, in such a way that applications are effectively executed. Follows several WMS components:

Workload Manager (WM) is the core component of WMS. Its purpose is to accept and fulfill requests for the management of jobs. For this it uses other components of WMS, depending on the given request.

Resource Broker (RB) (or MatchMaker) when given job description, it takes a list of all available resources (using GIIS and GRIS), filters it according to requirements of the job and then picks the best resource for it.

Logging and Bookkeeping Service (L&B) tracks job flow related events (important points of job life, e.g. submission, finding a matching CE, starting execution etc.) gathered from various WMS components as well as CEs (all those must be able to answer LB calls). The event informations are passed to a physically close component of the LB infrastructure This component stores them in a local disk file and delivers them further, e.g. to the monitoring system.

2.2 Globus Toolkit

Let us characterize in more detail the Globus Toolkit due to its widely usage in different Grid projects, among others including NASAs Information Power Grid [NIPG] or LHC Computing Grid. The GT began in 1994 in the USA where an experiment took place to create temporary links between 11 high-speed research networks to create I-WAY, a national grid, led by Rick Stevens (Argonne National Laboratory) and Tom DeFanti (University of Illinois at Chicago). Users of I-WAY were allowed to run applications on computers across the country. This successful experiment was financed by DARPA [10] and the first version of Globus Toolkit appeared in 1997.

Because it was open source it spread quickly. Due to its versatility, the GT components became used in majority of computing grid projects.

Globus Alliance represents a community of users and developers who work on open source software and its documentation. The GT software includes packages for security, information infrastructure, resource management, data management, communication, fault detection, and portability. For example, inquiry software is provided for discovering structure and state information for various common resource types, such as computers OS version and hardware configuration, scheduler queue status, storage system available space, and networks load.

The GT is packaged as a set of components that can be used either independently or together, also to develop applications. Today, a lot of corporations and institutions are interested in development of project specific grid technologies, often using the GT components. These technologies include for example Oracle10g [11], ALCHEMI [12], Condor-G [13], Sun Grid Engine [14] and other. These toolkits were used for building variety of projects including Enabling Grids for E-sciencE (EGEE) [15], Open Middleware Infrastructure Institute Europe (OMII-Europe) [16], Nordic Data Grid Facility (NDGF) [17], World Community Grid [18] as some of international projects and D-Grid [19] in Germany, National Grid Service (NGS) [20] in the United Kingdom and Open Science Grid (OSG) [21] in the USA as some of national grids.

2.3 Summary

In this section I tried to outline the general architecture and components of the Computing Grid and the purpose and function of some of the grid services. In the following sections I will shortly describe the LHC Computing Grid (LCG) and the middleware of the ALICE Grid project - AliEn.

3 LHC Computing Grid - LCG

LCG project [2] was approved on 20 September 2001 to develop, build and maintain a distributed computing infrastructure for the storage and analysis of data from the four LHC experiments. It was divided into 2 phases. The first one, from 2002 to 2005, was to build the necessary hardware infrastructure (testbead), to develop and prototype necessary software and services and perform regular tests of the system in construction. The second phase, foreseen from 2006 to the first beams in the LHC expected in 2008, was supposed to prepare the status of LCG services so that everything be in place and ready for the first data taking.

The project should provide access to experimental data acquired by LHC experiments to approximately 5000 scientists in 500 institutes and universities worldwide. The requirements for the year 2008 were estimated to a CPU capacity of 140 million of the SPECint2000 units (a common processor of the Pentium 4 family can deliver up to 1800 SPECint2000 [23], 60 PetaBytes of disk storage and 50 PetaBytes of mass (tape) storage.

The LCG architecture has been built utilizing middleware components developed by several other projects: Globus [6], Condor [24], Virtual Data Toolkit (VDT) [25] and the EGEE gLite toolkit [26]. The majority of computing resources available for LCG is operated as part of the EGEE project [15], others are part of OSG [21] and NDGF [17].

One of the LCG Project tasks is the development of the software for all experiments. This includes e.g. tools and frameworks for data management and event simulation and the support of analysis and database management.

For the data management, processing and storage, for the job submission and monitoring, for the authentication and authorization, all the LHC experiments are using a mixture of LCG-provided middleware components and services and experiment-specific components developed by the individual experiments grid groups. At each computing centre providing services for a given LHC experiment, a dedicated server is configured where all the experiment-specific grid tools are installed, so called VO-box [27]. This machine is an entry point for the particular experiment into the LCG system.

All the LHC experiments have been extensively testing their computing models and hardware and software infrastructure built for the data taking, processing, storage and access during the regular Data challenges (DC), when massive Monte Carlo event simulations were running at all the computing centers, providing large volumes of data available for further analysis. DCs have been going on ever since 2003. Lately, the LCG team has been running regular Service Challenges (SC) to probe and evaluate the current status of LCG services in environments approaching the requirements of LHC data taking.

3.1 Tiers

The LCG operates on the basis of a "four-tiered" model: all the computing centers involved in the LCG are ranked as a Tier-something level centers.

Tier-0 centre is the most powerful with respect to the hardware resources and is located at CERN. It will hold original raw data coming from the data acquisition systems and also the first-pass reconstruction will take place there.

Tier-1 centers, one level lower in the capacity of hardware resources, provide the raw data are replication and storage on tapes. Also, the reconstruction of raw data is processed here as well as analysis tasks that require access to large collections of data. The disk storage at Tier-1s holds replicas of the Event Summary Data (ESD) created by the reconstruction passes. There are altogether eleven Tier-1 sites, located in France, Germany, Italy, the Netherlands, the Nordic countries, Spain, the UK, Taipei, Canada and two sites are in the USA. These centers, same as Tier-0, should provide 24/7 support for the LCG users.

Tier-2 centers, again one more level lower in the capacity of hardware resources, should provide computational capacity and appropriate storage services for collective bulk Monte Carlo event simulations and for end-user analysis. For the analysis, Tier-2 centers obtain data from distributed Storage Elements, mostly from Tier-1s, where also the generated data are mostly stored.

Tier-3 Tier-3 level sites, which are smaller, are local computing facilities in universities and laboratories. These centers do not enter directly the Grid infrastructure and are envisaged for local analysis tasks. The access to the Grid data and computing facilities is provided by a simple user interface installed at the local machines.

Extremely important for the operation of the LHC Computing Grid is the high level networking. The national and regional networking organizations are collaborating closely with the Tier-1 regional centers and CERN in a working group to establish an appropriate network infrastructure to match LHC requirements: 10 Gb/s connectivity for Tier-0 ↔ Tier-1 traffic and at least 1Gb/s for Tier-1 ↔ Tier-2 traffic.

3.2 Tier-0 Architecture

The general Tier-0 architecture is based on three units providing processing resources, disk storage and tape storage. Each is build of many nodes connected on physical layer with hierarchical tree-structured Ethernet network. The logical connection of all nodes and functional units in the system provide three major software packages:

- a batch system (LSF) to manage the load on CPU resources
- a medium-size, distributed, global, shared file system (AFS) providing transparent access to a variety of repositories such as calibration data, programs, user space, etc.
- a disk pool manager emulating a distributed, global, shared file system for bulk data and an associated large tape storage

The heart of Ethernet infrastructure is based on set of highly redundant high throughput routers inter-connected with a mesh of multiple 10Gb/s connections (see Figure 2).

3.2.1 Resources

Resources accessible at Tier-0 centre in 2008 and planned resources for the following years are in Table 1. These values refer to the status on 17.11.2008. More thorough information are at [28].

CERN Tier-0	2008	2009	2010	2011	2012	2013
CPU (kSI2K)	11,170	23,908	35,310	37,410	40,910	45,470
Disk (TBytes)	2,423	5,656	5,542	5,942	6,142	7,200
Tape (TBytes)	10,780	22,432	36,090	51,120	66,760	83,500

Table 1: Resources available in Tier-0 centre in 2008 and prognosis for the following years

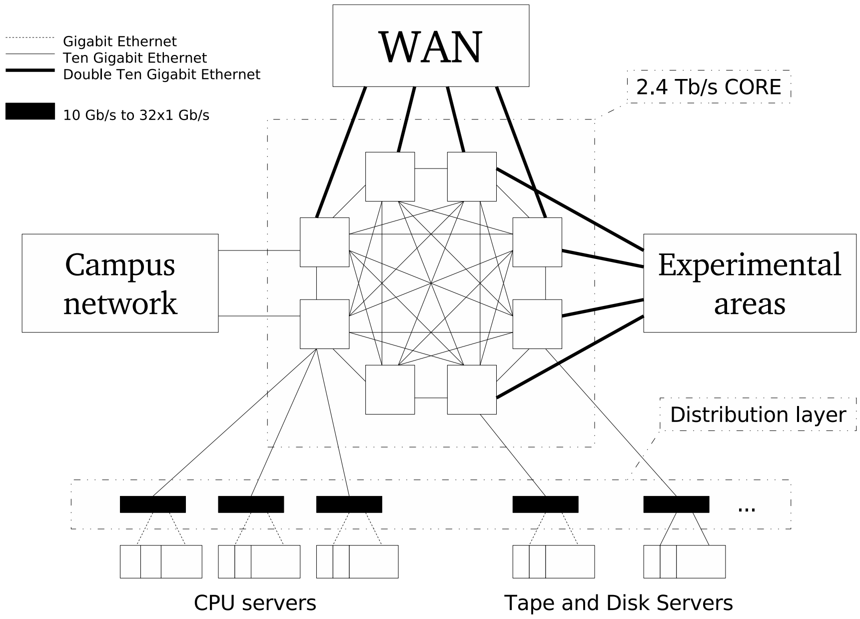

Figure 2: Layout of the Tier-0 network

3.3 Tier-1 Architecture

Tier-1 architecture is based on a few basic units: disks (providing online storage), tapes (archival storage), processing farms (delivering computing resources) and database storage (holding structured information). These - supported by physical infrastructure and using middleware and software packages - are presented as Grid services meeting agreed interface definitions. Many services are common for different VOs.

3.3.1 Archival Storage

Archival Storage systems in general stand for an automated tape library with front-end disk cache running Hierarchical Storage Manager (HMS). Within the LCG Grid common HSMs are CASTOR (CERN Advanced Storage Manager) [29], HPSS (High Performance Storage System) [30] and Enstore [31]. A Tier-1 is responsible for storing of archived data through life of experiments it supports. For archival storage the primary performance issue is long-term throughput rather than latency or peak throughput. The level of sustained throughput is determined by the speed and number of tape drives, the size and speed of the disk cache and the number and speed of the server machines to which these peripherals are connected.

Tier-1 centers are required to archivally store, reprocess and serve, as necessary, a fraction of experiment's raw data, Monte Carlo data and derived datasets.

3.3.2 Online Storage

Two ways of implementing online storage are possible. The first is use of relatively expensive robust centralized commercial systems - like FibreChannel connected RAID5 systems behind NFS file servers or systems provided by e.g. BlueArc. The second way offers less expensive, more distributed systems based on commodity hardware and public domain or community supported software - systems like dCache which run on Linux nodes using inexpensive commodity disks.

In the case of online storage record level access is required so a POSIX or POSIX-like interface is required. Also latency and peak transfer rate are much more important than in the archival storage. Online storage appears on Grid as Storage Element and supply the following services.

Reconstructed Data Service A reconstruction pass typically produces multiple levels of output. Large, very inclusive, sets called the Event Summary Data (ESD), more concise but still relatively comprehensive data sets for analysis purposes called the Analysis Object Data (AOD) and very compact highly structured sets, the TAG. AOD and TAG data sets are small enough to be stored at Tier-1 centers at multiple locations but ESD sets are in general very large and so their online storage is a specific responsibility of Tier-1 centers though some experiments may require that certain ESD sets also be maintained online. ESD sets are necessary for regeneration of derived data sets including AOD or derivation of custom data sets. For their size sustained high-bandwidth access is very important as is reasonably low latency.

Analysis Data Service AOD, TAG and other as concise datasets are typically accessed by individual physicists in frame of chaotic analysis. For this access performance, the peak bandwidth and latency are basic.

3.3.3 Computing

At Tier-1 centers, a Linux processor farm with resource management system, typically a batch scheduler like LSF [32], PBS [33] or Condor [24], is used. Grid access to computing resources is via a Computing Element (CE) which serves as Grid interface to batch queues of the centre. There is a number of services supplied by Tier-1 centers depending on the computing models of experiments they support.

esubparagraphReconstruction is a CPU-intensive activity, that requires pre-staged data from tapes and access to condition and calibration information.

Programmatic Analysis refers to passes made through more inclusive datasets of reconstruction to select data subsets. This does not take up a lot of computing resources but needs excellent connectivity for the CPUs.

Chaotic Analysis is individual user analysis. It can be CPU-intensive or/and I/O-intensive and thus creates very spiky loads in CPU/IO utilization.The amount of chaotic analysis done at Tier-1 centers as compared to that done at Tier-2 centers is experiment-dependent.

Calibration Calculation can vary grately in CPU/IO required and latency tolerated.

Simulation is in general very CPU-intensive activity that requires low I/O. The amount of simulation done at Tier-1 centers as compared to that done at Tier-2 centers is experiment-dependent.

3.3.4 Information Service.

A major information service which Tier-1 must support is serving the metadata which describes the data of experiments it supports. Other services make available information about conditions and calibrations required to analyze and process an experiment's data, information regarding the local site such as resource availability, performance monitoring, accounting and security auditing. The most commonly used database management system is MySQL.

3.3.5 Resources

Resources available at Tier-1 centers are summed in Table 2. Requested resources for individual experiments for year 2009 are in Table 3 - in parentheses is percentage of requested resources that will be available to the experiment - and number of Tier-1 centers dedicated to the experiment.

These values refer to the status on 17.11.2008. More information is available in [28].

Tier-1s Summary	2008	2009	2010	2011	2012	2013
CPU (kSI2K)	37,568	61,450	101,541	129,667	153,262	194,106
Disk (TBytes)	20,221	34,890	60,336	79,682	94,377	130,276
Tape (TBytes)	21,303	40,189	65,938	89,805	115,423	142,731

Table 2: Resources available in Tier-1 centers in 2008 and prognosis for the following years

Split 2009	ALICE	ATLAS	CMS	LHCb	SUM 2009
CPU (kSI2K)	19,900 (51%)	28,430 (106%)	16,300 (98%)	4,970 (102%)	69,600 (88%)
Disk (TBytes)	6,800 (57%)	20,920 (95%)	9,700 (87%)	2,759 (98%)	40,179 (87%)
Tape (TBytes)	12,400 (50%)	15,790 (93%)	15,000 (107%)	3,070 (106%)	46,260 (87%)
Number of T1s	6	10	7	6	n/a

Table 3: Requested resources from Tier-1 for individual experiments in 2009, in parentheses is percentage of requested amount that will be available to the experiment, and number of Tier-1 centers dedicated to the experiment

3.4 Tier-2 resources

Resources available at Tier-2 centers are summed in Table 4[1]. Requested resources for individual experiments for year 2009 are in Table 5 - in parentheses is percentage of requested resources that will be available to the experiment - and number of Tier-2 centers dedicated to the experiment. These values refer to status on 17.11.2008. More information is available [28].

[1]Note: The proposed estimates in the Tables 4-5 might not be close to the real numbers. For instance, the number of Tier-2 centers for the ALICE experiment is estimated to 16 in Table 5, while in the end of 2008 this number was over 70 (altogether 84 computing centers supporting ALICE in the Worldwide LCG).

Tier-2s Summary	2008	2009	2010	2011	2012	2013
CPU (kSI2K)	46,782	74,071	116,938	151,274	187,504	219,162
Disk (TBytes)	12,878	22,784	35,221	45,196	58,271	69,277
Tapes (TByte)	0	0	1500	2000	3000	3000

Table 4: Resources available in Tier-2 centers in 2008 and prognosis for the following years

Split 2009	ALICE	ATLAS	CMS	LHCb	SUM 2009
CPU (kSI2K)	14,300 (59%)	26,970 (100%)	28,100 (97%)	11,380 (67%)	80,750 (92%)
Disk (TBytes)	4,000 (58%)	13,300 (81%)	5,700 (62%)	23 (830%)	23,023 (99%)
Number of T2s	16	38	31	14	n/a

Table 5: Requested resources from Tier-2 for individual experiments in 2009, in parentheses is percentage of requested amount that will be available to the experiment, and number of Tier-2 centers dedicated to the experiment

3.5 Worldwide LCG

The LCG system is nowadays interfaced into other grid systems like OSG [21] or NDGF [17], thus forming a huge system of the Worldwide LHC Computing Grid (WLCG) [28].

4 AliEn

AliEn [34] is a lightweight Grid framework which is built around Open Source components using the combination of Web Service, standard protocols and distributed agent models.

It has been initially developed by the ALICE collaboration as a production environment for the simulation, reconstruction, and analysis of physics data in a distributed way. The system has been put into production for ALICE users at the end of 2001. Since then, ALICE has been using AliEn mainly for distributed production of Monte-Carlo data, detector simulation and reconstruction during regular Data Challenges. Since December 2007, when the ALICE detector started operation taking cosmic data and performing testing runs of different kind, AliEn has been used also for registration and reconstruction of the RAW data taken during the detector runs. Ever since 2007, also the single user analysis jobs have been managed by AliEn in the ALICE distributed framework. During the year 2008, almost 4,5 million of jobs have been successfully processed at the computing centers involved in the ALICE Grid system.

The architecture of AliEn provided the basics for the development of the gLite architecture. In gLite prototype, the AliEn components are used to provide an initial implementation of components and services like a file and metadata catalogue, task queue, package manager and various user interfaces, an application API and corresponding Grid Access Service (GAS).

4.1 Grid Access Service

The Grid Access Service [35] is a Web Service which provides the users entry point to the set of core services: the File and Metadata catalogue, Workload Management, Data Management etc. This collection is available for the user via the user interface and for applications by means of an API. The GAS Interface is not affected by changes to the underlying services and their implementaion. GAS follows the Web Service Resource Framework - WSRF [36] and runs under the gContainer, a Grid Service Container implementation using Perl, which supports WSRF and is a secure HTTP server. The gContainer instance consists of a WSRF::Lite Container and a stateful management service called gController. Other services are created via the stateless factory service gFactory.

4.2 Authentication and Authorization

During the authentication, the GAS is functioning as an ad-hoc user portal. Before connecting to GAS, a user creates a proxy certificate and stores it in the myproxy server. Then the user connects to the gFactory service running under an arbitrary gContainer and submits a session request. The gFactory queries the gController for the best possible locations to create an instance of GAS, retrieves the proxy from the myproxy server, creates the GAS and finally returns the address to the user. Afterwards the user talks directly to the newly created service. In addition to myproxy service, the Virtual Organization Membership Service [37] is also used. The system is able to take into account user roles when creating the GAS interface.

4.3 Workload Management System

AliEns workload management system (WMS) is based on the pull architecture. It's a set of central components (Task Queue, Job Optimizer) and site components (Computing Element, Cluster Monitor). The pull architecture has an advantage over the push one: WMS does not have to know actual state of all resources, which is crucial for large flexible grids. When the push architecture is used, WM must get, keep and analyze a huge amount of status data just to assign a job, which becomes difficult in the expanding grid environment. In the pull architecture, local agents running at individual sites ask for jobs after having checked the local hardware and software conditions and found them appropriate for the processing of the job. Thus the WM only deals with the requests of the local Job Agents to assign appropriate jobs. The descriptions of jobs in the form of ClassAds are managed by the central Task Queue.

At each site, there are running AliEn services CE, SE, ClusterMonitor and File Transfer Daemon (FTD). The AliEn CE takes care of the local Job Agents. The ClusterMonitor manages the connections from the site to central services, so there is only one connection from each site. The AliEn WM can be integrated with LCG or other Grid systems' WMS and the job management is organized by a collaboration of both systems. Sites using AliEn are running some services of the gLite, ARC or OSG, like e.g. gLite CE. The AliEn CE is in this case combined with the gLite CE in the job submission machinery.

4.3.1 Jobs

When a job is submitted by a user, its description in the form of a ClassAd is kept in the central Task Queue, where it awaits a suitable Job Agent for execution. The job description file is written in the Job Description Language (JDL).

Job description must hold:

- name of the executable to be run
- arguments to pass to the executable
- requirements on worker node
- input data list
- output data list
- software packages needed by the job
- other possible requirements

The only mandatory field for the user is the name of the executable. The central Job Optimizer checks requirements of all jobs that are waiting for the execution in the central Task Queue. It can change some requirements, or suggest data transfer so it would be more likely that some Job Agent picks up the job.

4.3.2 Job status

After it has been submitted, a job gets through several stages. Its existence begins with the status SUBMITTED and then INSERTED. While it is waiting in the Task Queue, its status is WAITING. When a suitable Job Agent is found, the status becomes ASSIGNED. If the assignment fails, the job ends with the status ERROR_A. If the site where the job is to be executed has a batch system then the job is first sent to the local batch queue and the status changes to QUEUED. After the the local processing of the job has started, the status becomes RUNNING. If there was an error during the execution and the job did not finish properly, its status is ERROR_E. Further, if the job needs validation its status at the end can be VALIDATED or ERROR_V depending on the result of the validation process. If the validation is OK, the job becomes SAVING, trying to upload the output files into a chosen Storage Element. If there is a problem during the saving of the output files, the status of the job is ERROR_SV. If the saving process finishes properly, the status is SAVED and then, since all the steps ended up properly, the job is successfully finished and gets the status DONE. Otherwise, the job gets FAILED (see Figure 3).

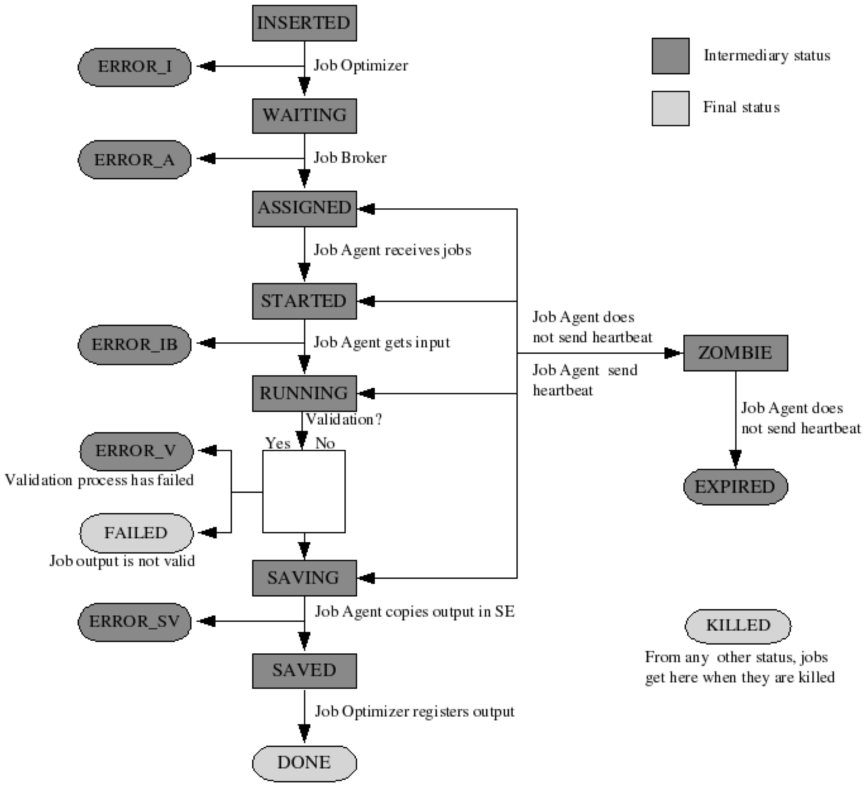

Figure 3: Possible status for AliEn jobs [38]

4.3.3 Computing element

The site CE is usually associated with the local batch system, but it could be also associated with the Worker Node. The AliEn CE has interfaces to LSF (Load Sharing Facility), PBS (Portable Batch System), DQS (Distributed Queuing System), CONDOR and SGE (Sun Grid Engine).

The AliEn CE is periodically submitting testing pilot jobs (Job Agents) to a chosen external Resource Broker or WMS. The role of these Job Agents (pilot jobs) is to verify the local hardware and software capacities at the site and only after the positive checkup to request a real job from the central Task Queue, or to die otherwise. The pilot jobs JDL contains the requirement, that the job is to be sent to its mother site CE. After the usual matchmaking procedure the pilot job is sent through the site CE into the local batch queue and then to a local Worker Node. After its startup, the pilot job (Job Agent) asks for a job from the central Task Queue, more precisely for a job JDL description, with the help of the local AliEn ClusterMonitor.

The JDL file of an AliEn Job Agent contains:

- name of the site Computing Element
- hostname of the Worker Node
- grid partitions to which the CE belongs
- the site close Storage Elements
- software packages installed in the ALICE software area in the site's NFS

4.4 File Transfer

This service provides the scheduled file transfer functionality and is running at the Storage Elements contained in the ALICE distributed storage cluster. The File Transfer Daemons (FTD) perform file transfer on user's behalf using a suitable external transfer protocol (like e.g. bbFTP or GridFTP). File transfers are requested and scheduled in exactly the same way as jobs, this time under the control of the File Transfer Broker.

4.5 File and Metadata Cataloque

The AliEn File Catalogue exhibits a structure similar to the familiar NFS and is designed to allow each directory node in the hierarchy to be supported by different database engines, running on different hosts and having different internal table structures optimized for a particular directory branch. It assures scalability of the system and allow growth of the cataloque as the files accumulate over the years (see Figure 4).

The File Catalogue aside from regular files also contains information about running processes in the system. Each job is given a unique id and a corresponding /proc/id directory where it can register temporary files, standard input and output as well as all job products. The directories and

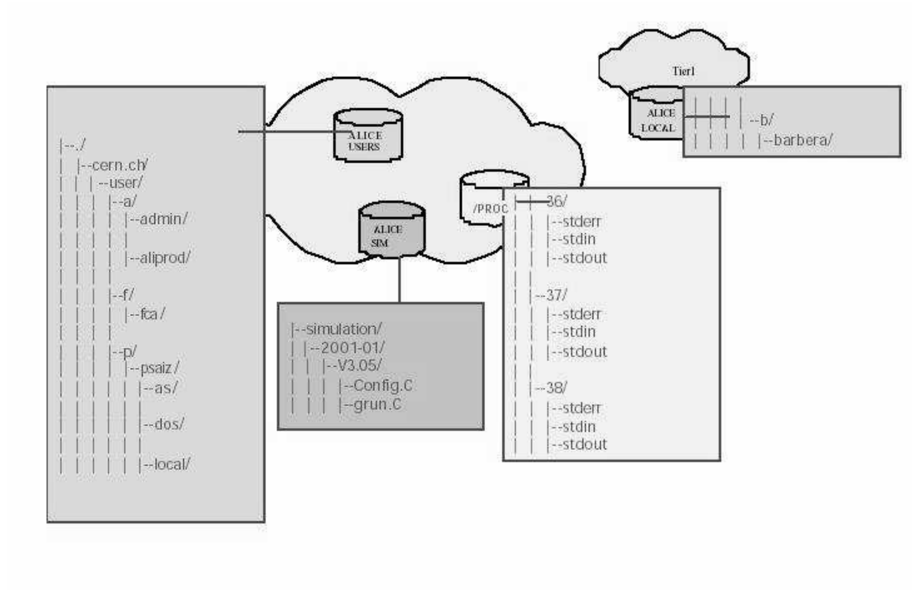

Figure 4: The hierarchy of the AliEn File Catalogue [39]

files in the File Catalogue have privileges according to Access Control List (ACL) model.

Unlike real file systems, the File Catalogue does not own the files; it only keeps an association between the Logical File Name (LFN) and (possibly multiple) Physical File Names (PFN) on a real file or mass storage system. PFNs describe the physical location of the files and include the access protocol (rfio, rootd), the name of the AliEn Storage Element and the path to the local file. The system supports file replication and caching. This information is used during the scheduling jobs for execution. The directories and files in the File Catalogue have privileges for owner, group and the rest of the world. This means that every user can have exclusive read and write privileges for his portion of the logical file namespace (home directory).

The FC keeps also an association between the LFN and a globally unique identifier (GUID). The labeling of each file with the GUID allows for the asynchronous caching. The write-once strategy combined with GUID labeling guarantees the identity of files with the same GUID label in different caches.

The hierarchy of files and directories in the AliEn FC is reflected in the structure of the underlying database tables. It allows for building of the AliEn Metadata Catalogue since it is possible to attach to a given directory an arbitrary number of additional database tables containing metadata information that further describes the content of the files in a given directory.

4.6 Package Manager

Package Manager (PackMan) automates the process of installing, upgrading, configuring, and removing software packages from the shared software area on a Grid site. It manages software common to all users of a VO. The packages are installed on demand, when requested by a Job Agent running on a Worker Node or upon an explicit request by the VO Software Manager. If a package is not already installed the PackMan would install it along with its dependencies and return a string with commands that client has to execute to configure the package and all its dependencies. The PackMan manages the local disk cache and clears it, when it needs more space to install newer packages.

While installing the package in a shared package repository, the PackMan will resolve the dependencies on other packages and, taking into account package versions, install them as well. This means that old versions of packages can be safely removed from the shared repository and, if needed again later, they will be re-installed automatically by the system. This provides an automated way to distribute the experiment specific software across the Grid.

4.7 Monitoring

Since the Workload Management does not depend directly on sophisticated monitoring, no special monitoring tools were developed in AliEn. Instead, MONitoring Agents using a Large Integrated Services Architecture (MonALISA) [40] framework was deployed as a part of the AliEn Monitor Module. The MonALISA system is designed as an ensemble of autonomous multi-threaded, self-describing agent-based subsystems which are registered as dynamic services, and together can collect and process large amounts of information.

The collected monitoring information is published via Web Service for use by AliEn Optimizers or for visualization purposes (see Figure 5). An extension of the network simulation code which is a part of MonALISA can provide a tool for optimization and understanding of the performance of the AliEn Grid system.

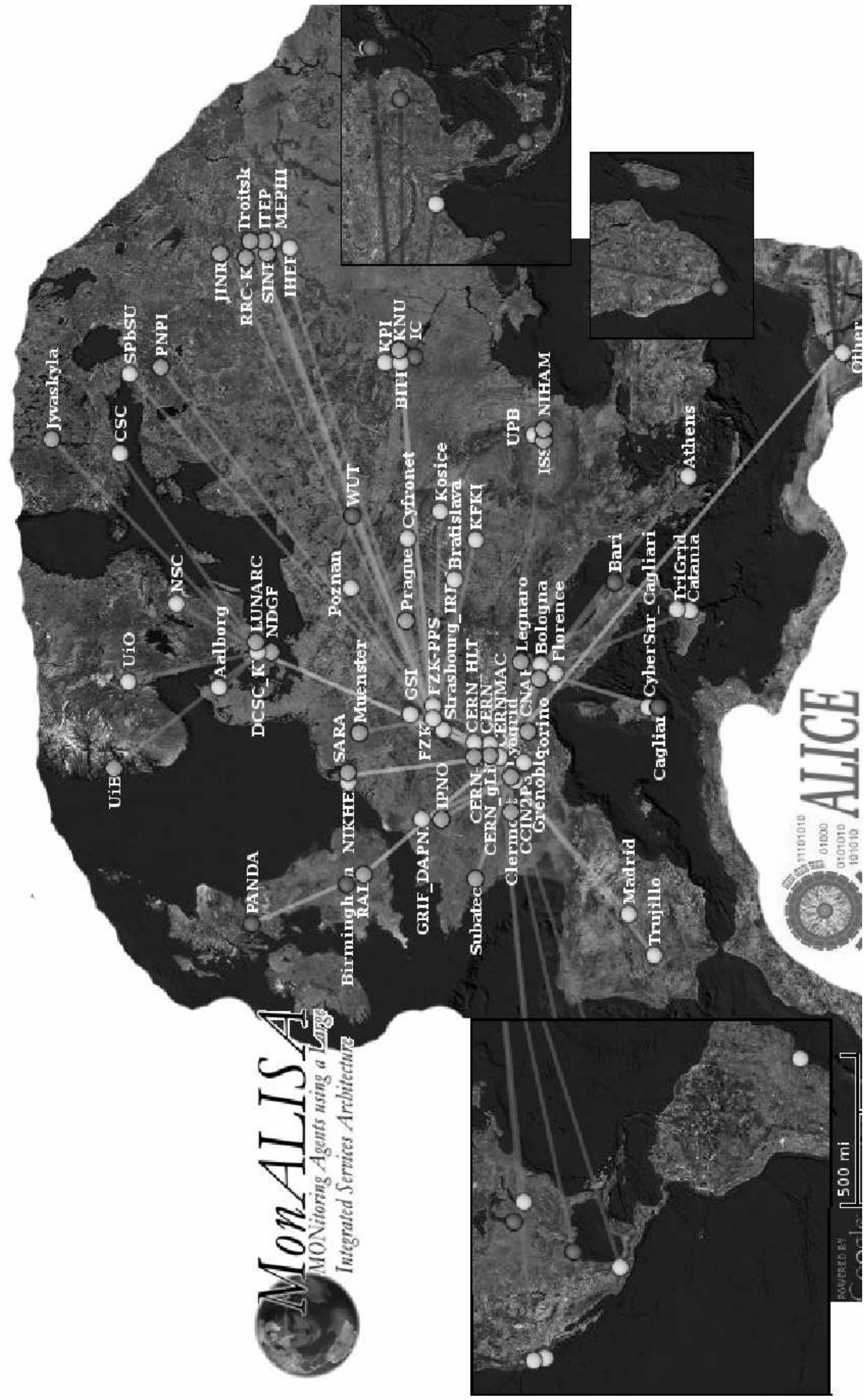

Figure 5: Map of centers using AliEn and their relations shown by MonALISA

5 Testing AliEn

To get some experience as a user of AliEn, I installed AliEn on a local computer and tested two basic tasks: getting files and submitting simple jobs.

To use alien one must be registered with one of the Virtual Organizations recognized by AliEn. There exists a quite extensive AliEn documentation: the basic instructions for the users of AliEn are described in [41], a more detailed version of the AliEn User's Guide is available in [42].

5.1 Installing AliEn

The installation of AliEn is possible on any Linux distribution using platform i686, x86_64 or ia64 and on the MacOS on platform i686-apple. The installation is described in [43] and is quite straightforward and in my case without any problems. It is executed by running the alien-installer script and provides a user-friendly GUI.

5.2 Getting files

Getting files from the AliEn database can be done in several ways. I chose the simplest one described in what follows. After the installation of AliEn was completed and the appropriate environment variables were set, I issued the command 'alien'. This way, the alien session was started and I was working in the alien shell. This shell looks similar to a UNIX shell and running it, the user gets into the AliEn catalogue. All the entries one can see in the session are the Logical File Names (LFN).

To get the files from the catalogue, one can use the command 'get'. To 'get' - download from the Grid - a specific file, e.g. one containing some simulation output, one must know the file's LFN. I was asked to retrieve some files produced during the Data Challenge in 2008. The production files from Data Challenges are reported on the ALICE web with a variety of details including also the files LFN in the AliEn catalogue. I found that the files I needed to download are in the directory /alice/sim/PDC_08/LHC08s/. The directory contains a system of subdirectories with the output files from the indivodual alien jobs running during the production cycle. Also, each of these sub-dirctories contains an archive of the data files intended for further analysis, and these archive files I wanted to retrieve.

```
[zach@gemma ~]$ alien
Warning! not able to use  /scratch/ALICE/cache as cache dir
info   Hello ALICE::Prague::Disk2
info   We will talk to the PackMan
/alice/cern.ch/user/d/dadamova/ > cd /alice/sim/PDC_08/LHC08s/200001/001/
/alice/sim/PDC_08/LHC08s/200001/001/ > get root_archive.zip
info   The file PDC_08/LHC08s/200001/001/root_archive.zip is in
SE => ALICE::Catania::DPM  pfn =>root://aliserv1.ct.infn.it:1094//dpm/ct.infn.it/home/-
xrootd/02/43131/bacbb22e-0fe6-11dd-9339-003048294c26

SE => Alice::Torino::DPM  pfn =>srm://grid008.to.infn.it:8443/dpm/to.infn.it/home/xroo-
td//02/43131/BACBB22E-0FE6-11DD-9339-003048294C26

info   Getting a security envelope..
info   In getfile, with ALICE::Catania::DPM
info   Getting the file root://aliserv1.ct.infn.it:1094///dpm/ct.infn.it/home/xrootd/0-
2/43131/bacbb22e-0fe6-11dd-9339-003048294c26
info   Error getting the file
info   Getting the copy didn't work :(. Does anybody else have the file?
info   Getting a security envelope..
info   In getfile, with Alice::Torino::DPM
info   Getting the file root://grid008.to.infn.it:1094//dpm/to.infn.it/home/xrootd//0-
2/43131/BACBB22E-0FE6-11DD-9339-003048294C26
info   Everything worked and got /home/zach/.alien/cache/BACBB22E-0FE6-11DD-9339-0030-
48294C26.98421231272949
And the file is /home/zach/.alien/cache/BACBB22E-0FE6-11DD-9339-003048294C26.98421231-
272949
[aliendb06c.cern.ch:3307] /alice/sim/PDC_08/LHC08s/200001/001/ > exit
bye now!
```

(The listing is trimmed by cutting off the beginnings of the lines to get the log messages fitted on the page. They contain just date and time.)

In this example, the retrieval of the file was successful. However, I was asked to retrieve 2000 files and Alien supports neither wildcards nor loops. But, you can run the 'alien' command from your Linux/Mac box with parameters/options that specify what command should be run using wildcards, while the loop itself is completed outside the alien shell. The command syntax is as follows:

```
[zach@gemma ~]$ alien -exec get /alice/sim/PDC_08/LHC08s/200003/001/root_archive.zip
Warning! not able to use  /scratch/ALICE/cache as cache dir
Jan  6 21:23:12  info   The file PDC_08/LHC08s/200003/001/root_archive.zip is in
                 SE => ALICE::Catania::DPM  pfn =>root://aliserv1.ct.infn.it:1094//dpm/-
 ct.infn.it/home/xrootd/13/48234/f8ae7b22-1012-11dd-8680-003048294c26

Jan  6 21:23:12  info   Getting a security envelope..
Jan  6 21:23:12  info   In getfile, with ALICE::Catania::DPM
Jan  6 21:23:16  info   Error getting the file
Jan  6 21:23:16  info   Getting the copy didn't work :(. Does anybody else have the file?
Jan  6 21:23:16  info   Getting a security envelope..
Jan  6 21:23:16  info   There is no envelope (error creating the envelope)!!
[zach@gemma ~]$
```

In this case, the 'get' command was not successful. Altogether, I have encountered 3 different errors while getting files:

- *There is no envelope (error creating the envelope)!!* - the file is at SE, but retrieving failed
- *Entry 'path-to-the-file' does not exist in the catalogue* - some files I asked for were not in the catalogue (this was due to my mistake)
- *Error contacting LDAP in alice-ldap.cern.ch:8389* - followed by one of those:
  - *IO::Socket::INET: connect: No route to host*
  - *IO::Socket::INET: Bad hostname 'alice-ldap.cern.ch:8389'*

According to the documentation, the 'error creating the envelope' is usually connected with the malfunction of the storage manager running at the given SE. The 'Error contacting LDAP' is usually due to network or host failures.

I was asked to retrieve 2132 files from the grid. Out of these, 538 download requests failed with the first described error message and 3 failed with the third one. I was able to retrieve 1590 files ($\sim$75%) of total size 93GB.

5.3 Submitting Jobs

To submit a job, one must run alien with a special parametr "login". Then, a job is submitted with the command "submit file_name", where file_name is the path in the AliEn catalogue to the JDL file with the description of the job to be submitted. When I tried to submit a simple job, however, already the command "alien login" returned a few confusing messages:

```
Jan  9 22:54:18  info   Doing 'vobox-proxy --vo alice query'
Can't exec "vobox-proxy": No such file or directory at /home/zach/AliEn/alien/li
b/perl5/site_perl/5.8.8/AliEn/LQ/LCG.pm line 1003.
Jan  9 22:54:18  info   Error: Error doing 'vobox-proxy --vo alice query'!!
No such file or directory at /home/zach/AliEn/alien/lib/perl5/site_perl/5.8.8/Al
iEn/LQ/LCG.pm line 1003.

Jan  9 22:54:18  error  No valid proxy found.
WARNING: Jan  9 22:54:18  error No valid proxy found.
```

In spite of the warning messages, the alien session started normally, but neither submission nor simple display of a file's content commands worked normally:

```
[aliendb06c.cern.ch:3307] /alice/cern.ch/user/d/dadamova/ > cat /jdl/SaveFile.jdl
Jan  9 23:33:54  info   Error getting the file: No valid proxy found.
Use of uninitialized value in pattern match (m//) at /home/zach/AliEn/alien/lib/perl5-
/site_perl/5.8.8/AliEn/UI/Catalogue/LCM.pm line 767.
[aliendb06c.cern.ch:3307] /alice/cern.ch/user/d/dadamova/ > submit /jdl/SaveFile.jdl
Error getting the file /jdl/SaveFile.jdl from the catalogue
[aliendb06c.cern.ch:3307] /alice/cern.ch/user/d/dadamova/ >
```

I guess the cause of the problems was a local inconsistency in the installation combined with momentarily failures in proxy registration services. To be completely sure, I would have to repeat the submission session another time and in case of repeated problems to consult the experts at CERN. Unfortunately, there was not enough time to complete this task.

6 Summary

To achieve the ultimate goal of sharing computing resources over large distances between dynamic user communities then the infrastructure of a Computing Grid is a very powerful technology. Tools like the Globus Toolkit [7] have been developed to help to simplify the grid development and are now assuring basic interoperability between newly emerging grids.

One of the existing grids is the LHC Computing Grid [2]. It was created using components developed by several other projects including e.g. the Globus Alliance [6] and the EGEE gLite toolkit [26] to store and analyze data coming from four LHC experiments. Majority of resources used by the LCG comes from the EGEE project [15]. The LCG system is nowadays interfaced into other grid systems like OSG [21] or NDGF [17], thus forming a huge system of the Worldwide LHC Computing Grid (WLCG) [28].

AliEn is a lightweight Grid framework, written in Perl, developed and used by the ALICE project. Only about 5% of the AliEn code was developed by the ALICE group, the rest are external modules. Although it was originally meant as an independent grid implementation and a few years worked that way, now only some of its components (like e.g. the File Catalogue, Task Quee, Computing Element, Storage Element, Package Manager) are used and deployed over the ALICE Grid production system and the rest of the grid services is provided by the LCG or LCG-interfaced middleware as a result of the ALiEn integration with the WLCG.

AliEn services have been successfully used by the ALICE Collaboration for the distributed production of simulated detector data, for the registration and reconstruction of the RAW data taken during the ALICE detector running in 2008 and for the management of the end user analysis on the grid.

7 Conclusion

Computing Grids, though now widely used, are still in early development. The vision of reaching the same level of deployment and wide use as the electric power grid and the water supply systems remains still a dream, but this world is approaching a wake-up and it won't take long.

References

[1] LHC: The Large Hadron Collider. http://lhc.web.cern.ch/lhc/.

[2] LHC Computing Grid. http://lcg.web.cern.ch/LCG/.

[3] Ian Foster and Carl Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1999.

[4] Ian Foster, Carl Kesselman, and Steven Tuecke. *The Anatomy of the Grid: Enabling Scalable Virtual Organizations*. International J. Supercomputer Applications, 2001.

[5] The Legion Project. http://legion.virginia.edu/.

[6] The Globus Alliance. http://www-unix.globus.org/alliance/.

[7] The Globus Toolkit. http://www-unix.globus.org/toolkit/.

[8] Global Grid User Support. https://gus.fzk.de/pages/home.php.

[9] A Grid Computing Dictionary. http://grid-it.cnaf.infn.it/fileadmin/users/dictionary/dictionary.html.

[10] DARPA. http://www.darpa.mil/.

[11] Oracle10g. http://www.oracle.com/database/index.html.

[12] ALCHEMI. http://www.alchemi.net/.

[13] Condor-G. http://www.cs.wisc.edu/condor/condorg/.

[14] Sun Grid Engine. http://www.sun.com/software/gridware/.

[15] Enabling Grids for E-sciencE. http://public.eu-egee.org/.

[16] Open Middleware Infrastructure Institute Europe. http://www.omii-europe.org/.

[17] Nordic Data Grid Facility. http://www.ndgf.org/ndgfweb/home.html.

[18] World Community Grid. http://www.worldcommunitygrid.org/.

[19] D-Grid. http://www.d-grid.de/index.php?id=1&L=1.

[20] National Grid Service. http://www.grid-support.ac.uk/.

[21] Open Science Grid. http://www.opensciencegrid.org/.

[22] LHC Computing Grid. *Technical Design Report*, June 2005. http://cern.ch/lcg/tdr.

[23] SPEC CPU2000 Results. http://www.spec.org/cpu2000/results/cpu2000.html.

[24] Condor. http://www.cs.wisc.edu/condor/.

[25] Virtual Data Toolkit. http://vdt.cs.wisc.edu/.

[26] EGEE gLite. http://glite.web.cern.ch/glite/.

[27] VO box HowTo. http://goc.grid.sinica.edu.tw/gocwiki/VO-box_HowTo.

[28] WLCG Resources Reporting. http://lcg.web.cern.ch/LCG/resources.htm.

[29] CASTOR: CERN Advanced Storage Manager. http://castor.web.cern.ch/castor/.

[30] HPSS: High Performance Storage System. http://www.hpss-collaboration.org/hpss/index.jsp.

[31] Enstore. http://www-ccf.fnal.gov/enstore/design.html.

[32] LSF: Load Sharing Facility. http://www.platform.com/Products/platform-lsf.

[33] PBS: Portable Batch System. http://www.pbsgridworks.com/.

[34] AliEn Homepage. http://alien.cern.ch/twiki/bin/view/AliEn/Home.

[35] A. J. Peters P. Buncic, J. F. Grosse-Oetringhaus and P. Saiz. The architecture of the alien system. Technical report, CERN, 2003.

[36] The WS-Resource Framework. http://www.globus.org/wsrf/.

[37] Virtual Organization Membership Service. http://hep-project-grid-scg.web.cern.ch/hep-project-grid-scg/voms.html.

[38] AliEn Job Status.
http://pcalimonitor.cern.ch/show?page=jobStatus.html.

[39] AliEn File Catalogue. http://aliceinfo.cern.ch/static/AliEn/About_AliEn/ch04s02.html.

[40] MonALISA: Monitoring Agents using a Large Integrated Services Architecture. http://monalisa.cern.ch/monalisa.html.

[41] How to use AliEn.
http://alien.cern.ch/twiki/bin/view/AliEn/HowToUseAliEn.

[42] ALICE Analysis User Guide. http://project-arda-dev.web.cern.ch/project-arda-dev/alice/apiservice//AA-UserGuide-0.0m.pdf.

[43] AliEn Downloading and Instalation.
http://alien.cern.ch/twiki/bin/view/AliEn/Download.