MonALISA : A Distributed Service System for Monitoring, Control and Global Optimization

Iosif Legrand[1]
California Institute of Technology, Pasadena, California, USA
E-mail: Iosif.Legrand@cern.ch

Harvey Newman
California Institute of Technology, Pasadena, California, USA
E-mail: Harvey.Newman@cern.ch

Ramiro Voicu
California Institute of Technology, Pasadena, California, USA
E-mail: Ramiro.Voicu@cern.ch

Costin Grigoras
CERN, Geneva, Switzerland
E-mail: Costin.Grigoras@cern.ch

Catalin Cirstoiu
University Politehnica of Bucharest, Romania
E-mail: Catalin.Cirstoiu@cern.ch

Ciprian Dobre
University Politehnica of Bucharest, Romania
E-mail: Ciprian.Dobre@cs.pub.ro

The MonALISA (Monitoring Agents in A Large Integrated Services Architecture) framework provides a set of distributed services for monitoring, control, management and global optimization for large scale distributed systems. It is based on an ensemble of autonomous, multi-threaded, agent-based subsystems which are registered as dynamic services. They can be automatically discovered and used by other services or clients. The distributed agents can collaborate and cooperate in performing a wide range of management, control and global optimization tasks using real time monitoring information.

XII Advanced Computing and Analysis Techniques in Physics Research
Erice, Italy
3-7 November, 2008

[1] Speaker

1. Introduction

An essential part of managing global-scale systems is a monitoring system that is able to monitor and track in real time many site facilities, networks, and tasks in progress. The monitoring information gathered is essential for developing the required higher level services, the components that provide decision support and some degree of automated decisions and for maintaining and optimizing workflow in large scale distributed systems. These management and global optimization functions are performed by higher level agent-based services. Current applications of MonALISA's higher level services include optimized dynamic routing, control and optimization for large scale data transfers on dedicated circuits, data transfers scheduling, distributed job scheduling and automated management of remote services among a large set of grid facilities. MonALISA is currently used around the clock in several major projects and has proven to be both highly scalable and reliable. More than 350 services are running at sites around the world, collecting information about computing facilities, local and wide area network traffic, and the state and progress of the many thousands of concurrently running jobs.
In this paper we present the system architecture and several examples of how this system is used to monitor and control large scale distributed systems.

2. The MonALISA System Design

The MonALISA system [1] is designed as an ensemble of autonomous self-describing agent-based subsystems which are registered as dynamic services. These services are able to collaborate and cooperate in performing a wide range of distributed information-gathering and processing tasks.

An agent-based architecture [2] of this kind is well-adapted to the operation and management of large scale grids, by providing global optimization services capable of orchestrating computing, storage and network resources to support complex workflows. By monitoring the state of the grid-sites and their network connections end-to-end in real time, the MonALISA services are able to rapidly detect, help diagnose and in many cases mitigate problem conditions, thereby increasing the overall reliability and manageability of the grid.

The MonALISA architecture, presented in Figure 1, is based on four layers of global services. The entire system is developed based on the Java [3] technology. The network of Lookup Discovery Services (LUS) provides dynamic registration and discovery for all other services and agents. Each MonALISA service executes many monitoring tasks in parallel through the use of a multithreaded execution engine, and uses a variety of loosely coupled agents to analyze the collected information in real time.

The secure layer of Proxy services, shown in the figure, provides an intelligent multiplexing of the information requested by clients or other services. It can also be used as an Access Control Enforcement layer. As has been demonstrated in round-the-clock operation over the last six years, the system integrates easily with a wide variety of existing monitoring tools and procedures, and is able to provide this information in a customized, self-describing way to any other set of services or clients.

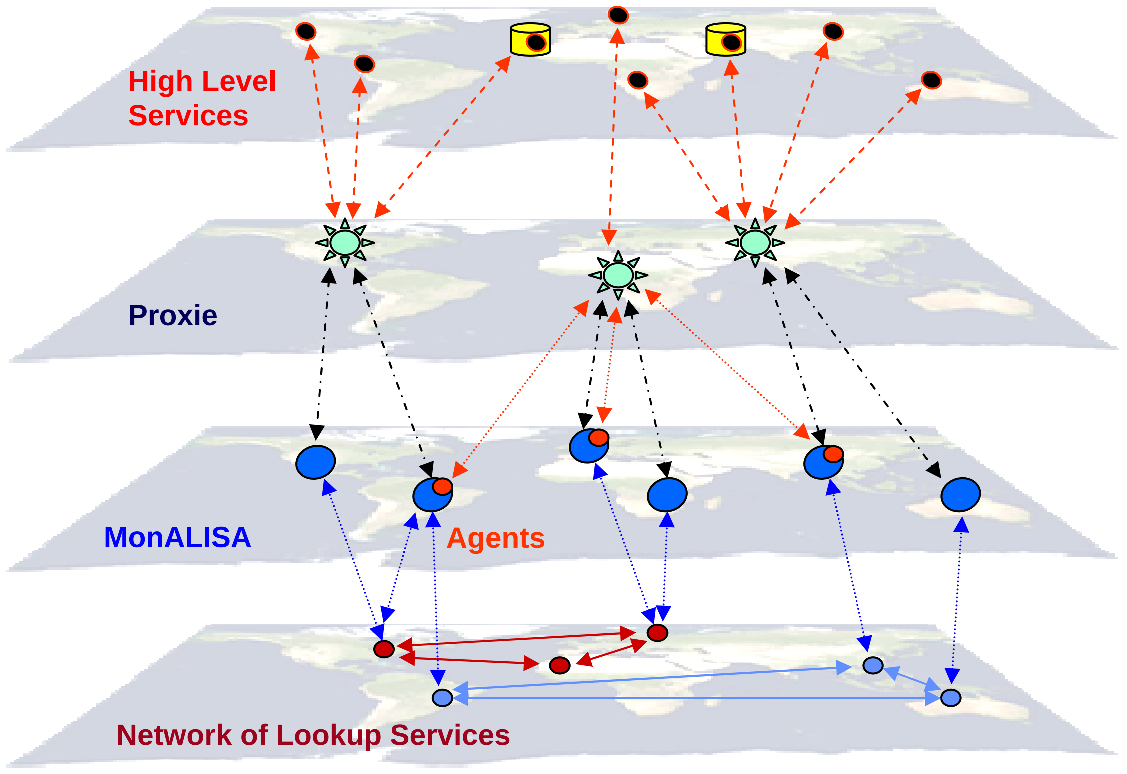


Figure 1: The four layers of main services and components in the MonALISA framework

2.1 The Distributed Service System

The basic component of the monitoring framework is the MonALISA service. This is an ensemble of multi-threaded subsystems, which carry out several functions in an independent fashion, being able to operate autonomously. Here are some of the most important functions it can perform:

- *monitoring* of a large number of entities (hosts, services, applications, network) using simultaneously several modules which interact actively with the entities or passively listen for information, or interface with other existing monitoring tools;
- *filtering and aggregation* of the monitoring data, providing additional information based on several data series;
- *storage* for short period of time of the monitoring information, in memory or in a local database. This way, clients can get also history for the monitoring information they are interested in, not only the real-time monitored values;
- *web services* for direct access to the monitored data, from the monitoring service. This way, local clients can get the information directly, without following the whole chain;
- *triggers, alerts and actions* based on the monitored information, being able to react immediately when abnormal behavior is detected;
- *controlling* using dedicated modules allow performing more complicated actions, that cannot be taken only to the local flow of information. This way, the service can act based on

monitoring information from several services at the command of a controlling client, or direct or indirect users request.

The communication between clients and the selected services is performed through a set of proxy services. The proxy services also register their existence in the lookup services and are dynamically discovered. The usage of proxies for the communication has multiple advantages. First, the data from a service can be sent only once and is multiplexed to all the interested clients connected to the same proxy. This reduces the load on the service and on the network, in case of many clients. It also provides a natural way of scaling the system, to support a large number of clients. Second, this has an extremely valuable practical advantage: the services don't require incoming network connectivity when clients running outside service's domain connect to the service. This distributed architecture is fault tolerant because it does not have a single point of failure. All the core services (lookup and proxy services) are replicated in different locations worldwide.

2.2 Monitoring Clients and Optimization Services

The clients of this monitoring framework use the Lookup Discovery Services (LUSs) to find all the active MonALISA services running as part of one or several group "communities". The lookup services also provide a notification mechanism which is used when new services are started or when services are no longer available.

Once the connection is established through one of the proxies, clients can get any real-time or historical data by using a predicate mechanism for selecting or subscribing to selected monitored values. Predicates are based on regular expressions to match the attribute description of the measured values a client is interested in. They may also be used to impose additional conditions or constrains for the interested values. For the historical data the predicates are used to generate SQL queries. The subscription requests will create a dedicated thread in the service, to serve each client. This thread performs the matching test for the client predicates with the measured values in the data flow and is responsible to send them to the client as compressed serialized objects. Having an independent thread per client allows sending the information they need, fast, in a reliable way and it is not affected by communication errors which may occur with other clients. There are three types of clients that can use the MonALISA framework to display the gathered monitoring information from all the distributed services:

- *Graphical clients*, which provide dedicated panels for some of the common monitored parameters. They also allow the user to interactively choose what parameters he wants to observe and in what format (real time chart/histogram, history plot, pie charts, stacked bars etc.) and get immediately the result from one or several services. This provides great flexibility in investigating the monitored entities (detailed, correlated, multi-site plots);
- *Repository clients*, that offer long history for a few, preconfigured parameters. The time series for these parameters are stored in a database for long-term viewing purposes. Due to the large amount of data (from all the services in community), the typical usage is for

storing aggregated and summary values. For those, it offers a set of predefined charts in a web application format that allow selecting among the set of time series and the time interval to plot. The repository client has been extended to support actions and alerts. The two clients complement each other in the support they offer to the community administrator;

- *Stub clients*, which represent the shared core among the first two clients and is the platform that can be used to build custom clients capable to take local and global action in the MonALISA framework.

Clients can also send back to the monitoring services not only predicates to select the monitoring information they are interested in, but also commands that allow changing the configuration of the service or controlling the available modules. Users can issue these commands directly from the graphical client or as a result of the configured actions in the repository or other dedicated clients.

The *stub client* represents the platform for developing high level *Optimizer* agents that consider the optimization of a large distributed system that they can monitor and control through the help of the distributed services.

3. Network Monitoring and Management

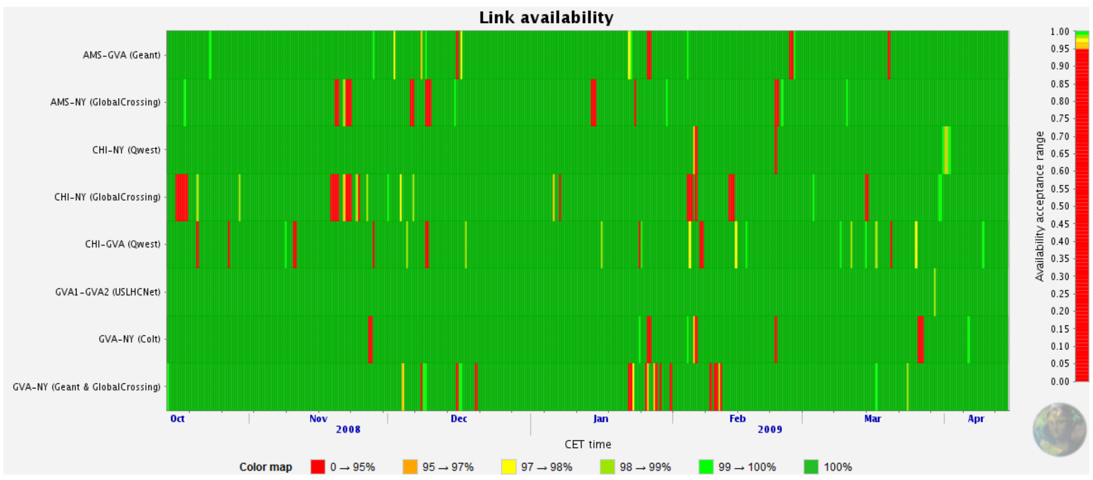
Network monitoring is vital to ensure proper network operation over time, and is tightly integrated with all the data intensive processing tasks used by the LHC experiments. Besides the central role in the performance of pro-active interventions in the case of network failure, network monitoring helps establish long term trends in terms of network utilization, and provides global quality figures about the network health as well as a database of historical events that should help speed up incident resolution in the future.

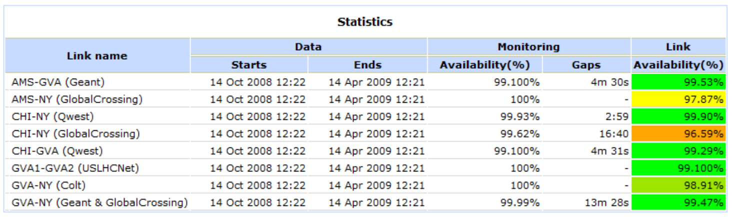
In order to build a coherent set of network management services it is very important to collect in near real-time information about the network traffic volume and its quality, and analyze the major flows and the topology of connectivity. Access to both real-time and historical data, as provided by MonALISA, also is important for developing services able to predict the usage pattern, to aid in efficiently allocating resources “globally” across a set of network links. A large set of MonALISA monitoring modules has been developed to collect specific network information (SNMP, TL1, ICMP) or to interface it with existing monitoring tools (NetFlow, sFlow, RRD, MRTG…). These modules have been field-proven to function with a very high level of reliability over the last few years.

3.1 The Network Monitoring the USLHCnet

MonALISA is used to provide reliable, real-time monitoring of the USLHCNet [4] infrastructure. In each point of presence (GVA, AMS, CHI, NYC) we run a MonALISA service to monitor the links, the network equipment and the peering with other networks. Each major link is monitored at both ends from two independent MonALISA services (the local one and one from a remote site). MonALISA services keep locally the history of all the measurements and a global aggregation, for long term history, is kept in a MonALISA repository. Dedicated TL1 modules for the Ciena CD/CI [5] were developed to collect specific information on topology, dynamic circuits and operational status.

We monitor the status for all WAN links and peering connections. For the Force10 switches we use SNMP and for the Ciena CD/CI the TL1 interface. The repository analyzes the status information from all the distributed measurements, for each segment, to generate reliable status information. Measurements are done every ~30s and the full history is kept in the repository database. The system allows one to transparently change the way a WAN is operated (via Force10 or Ciena CD/CI) and keeps consistent history. Figure 2 shows the panel that allows analyzing the links availability for any time interval.





Statistics

Link name	Data		Monitoring		Link
	Starts	Ends	Availability(%)	Gaps	Availability(%)
AMS-GVA (Geant)	14 Oct 2008 12:22	14 Apr 2009 12:21	99.100%	4m 30s	99.53%
AMS-NY (GlobalCrossing)	14 Oct 2008 12:22	14 Apr 2009 12:21	100%	-	97.87%
CHI-NY (Qwest)	14 Oct 2008 12:22	14 Apr 2009 12:21	99.93%	2:59	99.90%
CHI-NY (GlobalCrossing)	14 Oct 2008 12:22	14 Apr 2009 12:21	99.62%	16:40	96.59%
CHI-GVA (Qwest)	14 Oct 2008 12:22	14 Apr 2009 12:21	99.100%	4m 31s	99.29%
GVA1-GVA2 (USLHCNet)	14 Oct 2008 12:22	14 Apr 2009 12:21	100%	-	99.100%
GVA-NY (Colt)	14 Oct 2008 12:22	14 Apr 2009 12:21	100%	-	98.91%
GVA-NY (Geant & GlobalCrossing)	14 Oct 2008 12:22	14 Apr 2009 12:21	99.99%	13m 28s	99.47%

Figure 2 Monitoring the status of the major trans-Atlantic links.

The MonALISA framework is used to monitor the total traffic on all the Force 10 ports and on the Ethernet ports on the CIENA CD/CIs. Different aggregated views are presented: total traffic (an example is shown in Figure 3), total trans-Atlantic traffic, peering at each point of presence as well as integrated traffic over any time interval (Figure 4).

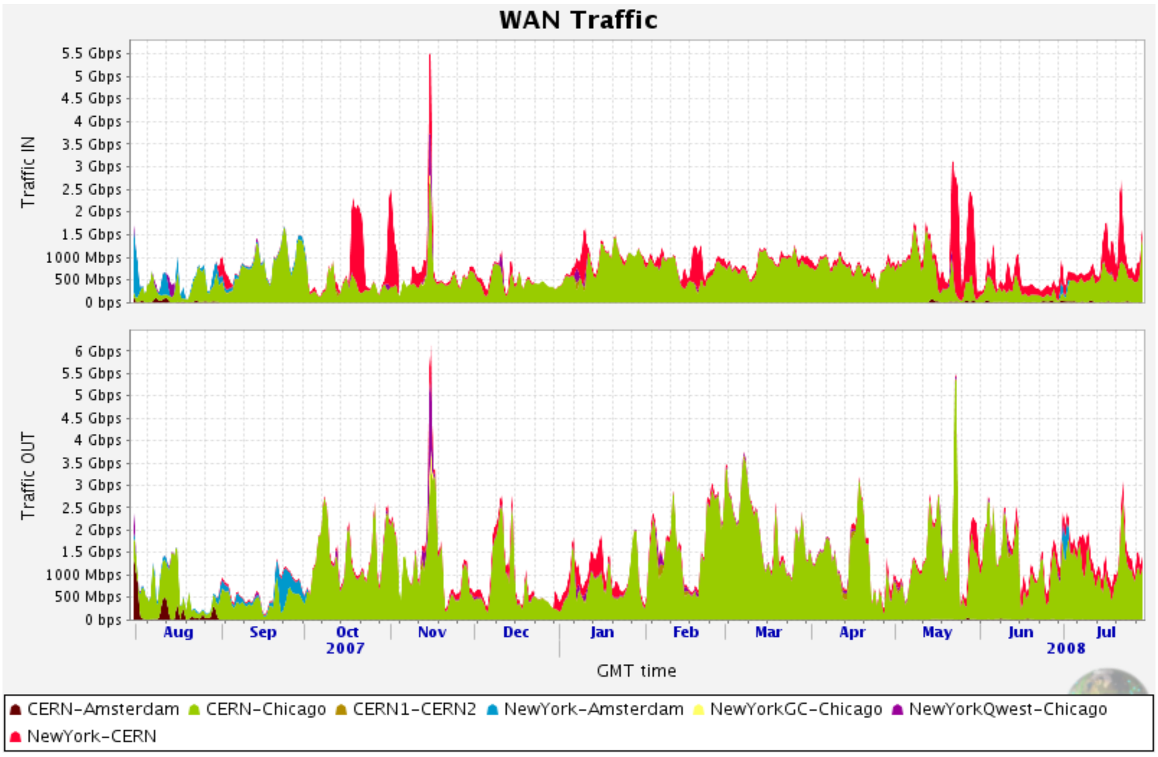


Figure 3: Annual (August 2007 – Present) traffic history for the major links in USLHCNet. On shorter timescales, peaks of 7-9 Gbps also are seen.

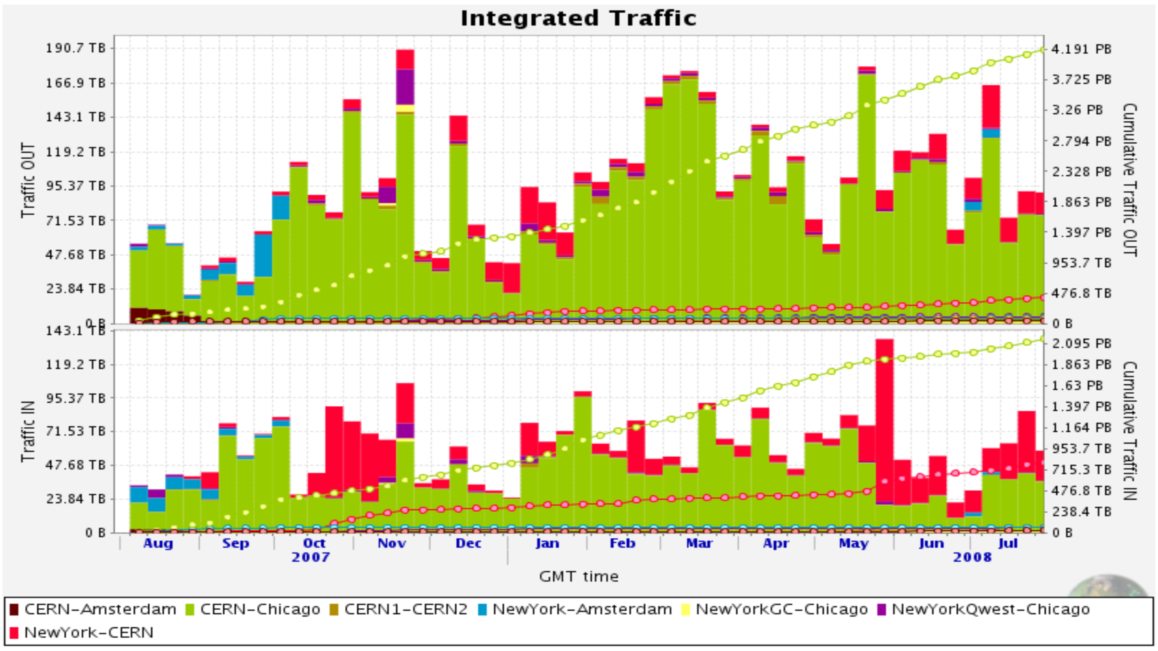


Figure 4: Integrated traffic on the USLHCNet links

The operational status for the Force10 ports and all the Cieana CD/CI alarms are recorded by the MonALISA services. They are analyzed and email notification is generated based on different error conditions. We also “monitor” the services used to collect monitoring information. A global repository for all these alarms is available on the MonALISA servers, which allows one to select and sort the alarms based on different conditions.

3.2 Network Topology

The MonALISA framework is used to construct in near real time the network topology for different types of technologies currently used in the High Energy Physics infrastructure. For the routed networks, MonALISA is able to construct the overall topology of a complex wide area network, based on the delay on each network segment determined by tracepath-like measurements from each site to all other sites, as it is illustrated in Figure 5. The combined information from all the sites allows one to detect asymmetric routing or links with performance problems. For global applications, such as distributing large data files to many grid sites, this information is used to define the set of optimized replication paths.

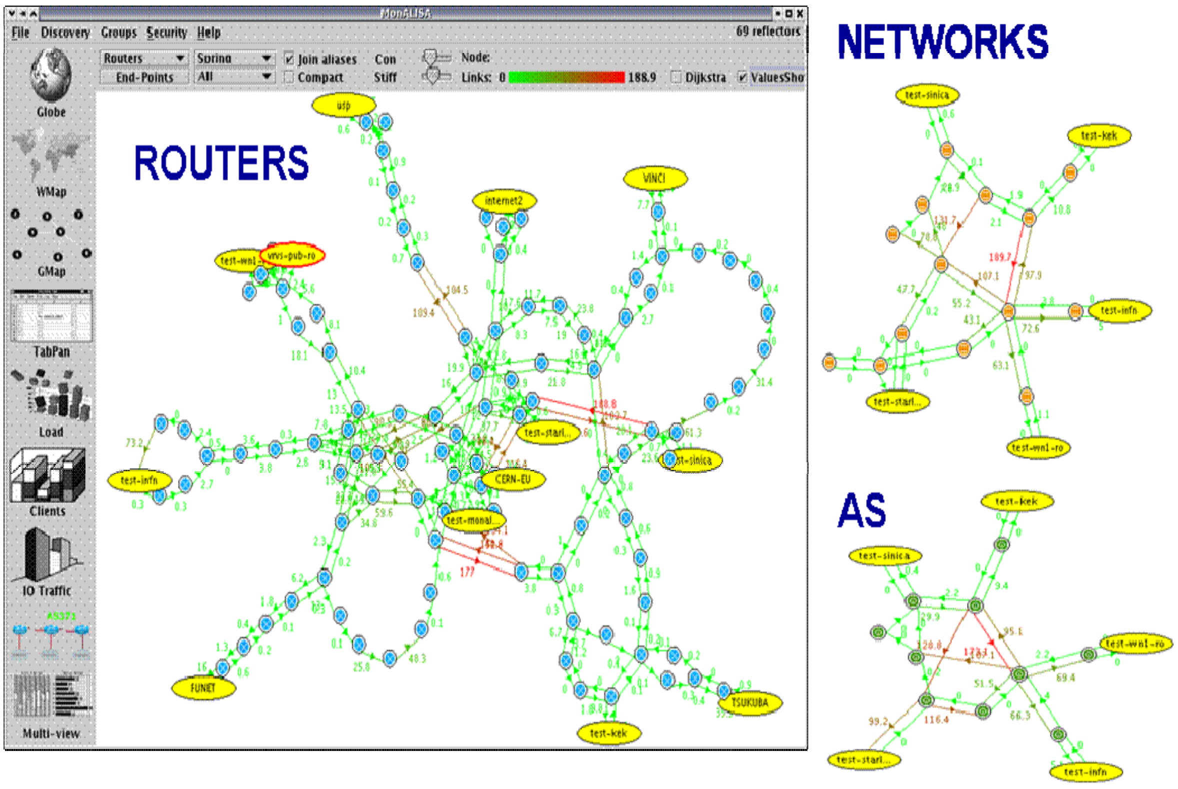


Figure 5 MonALISA real time view of the topology of WANs used by several High Energy Physics centers. It is possible to present all the routers, or the networks or just the “autonomous system” (AS). The real-time diagrams are useful to detect asymmetric routing between sites and segments with relative large Round Trip Time (presented as red lines)

Specialized TL1 modules are used to monitor the power on Optical Switches (Glimmerglass [6] and Calient [7]) and to present the topology. The MonALISA framework allows one to securely configure many such devices from a single GUI, to see the state of each link in real time, and to have historical plots for the state and activity on each link. It is also easy to manually create an end to end optical path using the MonALISA GUI.

For the Ciena CD/CI nodes we provide real-time information for the OSRP connections with all the attributes for the SONET links. The topology of all the circuits created in the entire network is presented in real time in the MonALISA interactive client. This panel allows one to select any set of circuits, and it presents how they are mapped onto the physical network, with all their attributes.

4. On Demand, End To End Optical Circuits

The MonALISA framework has been applied to develop an integrated Optical Control Plane system (OCPS) that controls and creates end-to-end optical paths on demand, using optical switches. As part of the development of end-to-end circuit-oriented network management services, we developed dedicated modules and agents to monitor, administer and control Optical Switches; specifically the purely photonic switches from Glimmerglass [6] and Calient [7]. The modules use TL1 commands to monitor the connectivity matrix of each switch, as well as the optical power on each port. Any change in the state of any link is reported to dedicated agents. If a switch is connected to the network, or if it ceases to operate, or if a port's light level changes, these state changes are detected immediately and are reflected in the topology presented by the MonALISA Graphical User Interface (GUI). By using the GUI, an authorized administrator also can manually construct any light path, and monitor the optical power on each new link as it is created.

The distributed set of MonALISA agents was used to control the optical switches, and to create an optical path on demand. The agents use MonALISA's discovery layer to "discover" each other, and then communicate among themselves autonomously, using the Proxy services. Each proxy service can handle more than 1,000 messages per second, and several such services are typically used in parallel. This ensures that the communications among the agents is highly reliable, even at very high message-passing rates.

The set of agents also is used to create a global path or tree, as it knows the state and performance of each local area and wide area network link, and the state of the cross connections in each switch. The routing algorithm provides global optimization by considering the "cost" of each link or cross-connect. This makes the optimization algorithm capable of being adapted to handle various policies on priorities, and pre-reservation schemes. The time to determine and construct an optical path (or a multicast tree) end-to-end is typically less than one second, independent of the number of links along the path and overall the length of the path. A schematic view of how the MonALISA agents are used to create an optical path for an (authorized, authenticated) end-user application is presented in Figure 6.

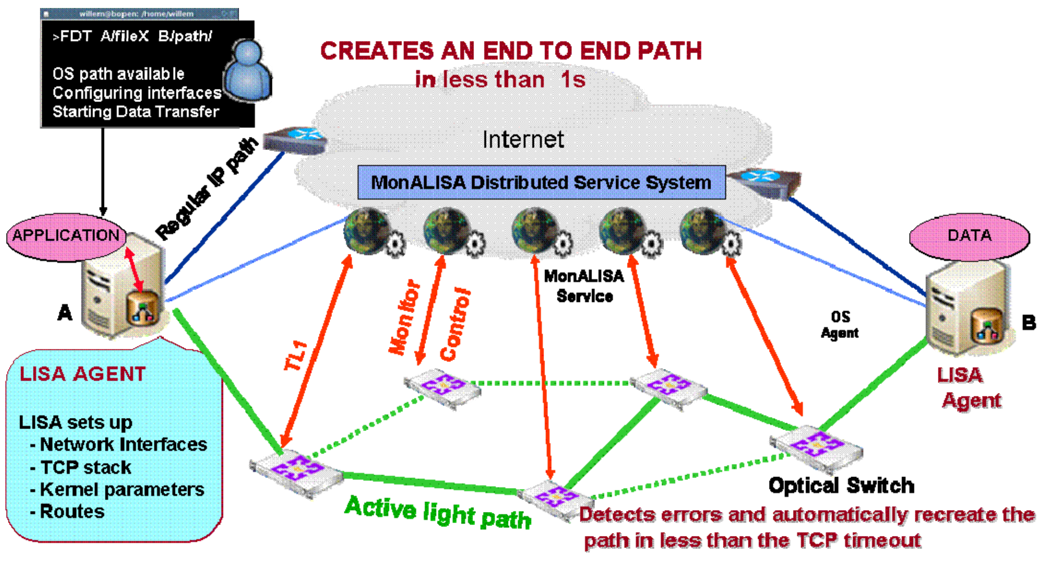


Figure 6: MonALISA agents are used to monitor and control optical switches. The agents interact with end-user applications to provision an optical path on demand.

If network errors are detected, an alternative path is set up rapidly enough to avoid a TCP timeout, so that data transfers will continue uninterrupted. This functionality is important in the construction of the virtual circuit-oriented network services.

Figure 7 shows an example of how MonALISA is used to create dynamically, on demand, a path between two end-systems at CERN and Caltech. The topology, the cross- connections, the ports and the segments where light is detected and the end-to-end path created by the system all are displayed in real-time. The end to end path is created in approximately 0.5 seconds, and then disk-to-disk data transfer using Fast Data Transfer (FDT) [8] is started. The agents controlling the optical switches detect the optical power lost and they created another complete path in less than 1 second. The data transfer was started using the Geneva – Starlight link, than we simulated a power cut on this circuit, and an other end to end optical path was created using the CERN –Manlan link. The alternative path was set up rapidly enough to avoid a TCP timeout, so that data transfers continue uninterrupted (Figure 7, right). We simulated four consecutive “fiber cuts” in the circuits over the Atlantic and the data set transfer between CERN and Caltech was switched between the two links four times uninterrupted. The TCP frames and the speed of the transfer recover in less than 20 s after the path was changed. As soon as the transfer initiated by the end-user application was completed, the path and the resources used were released.

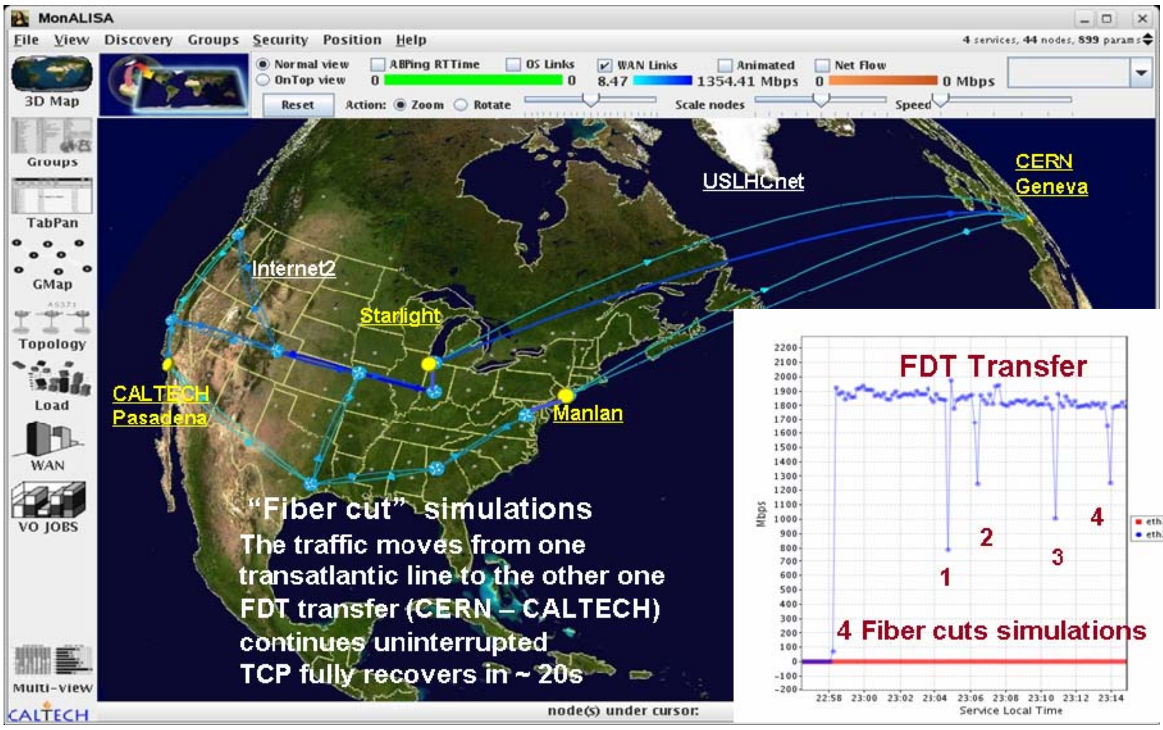


Figure 7: MonALISA agents used to create an end to end path between CERN and Caltech and to efficiently transfer data using FDT. Four "fiber cut" simulations were done for the two transatlantic circuits connecting CERN (Geneva) with Starlight (Chicago) and Manlan (New York). The "alternative path" was created rapidly enough to avoid TCP timeout and the FDT traffic continued uninterrupted.

5. Monitoring ALICE distributed computing environment

According to the ALICE Computing Model, each site member of the collaboration provides a machine dedicated to running VO-specific services, frequently called *VoBox*. The distributed computing infrastructure is controlled by AliEn [9] (ALIce Environment), a software package that has two main sets of services: central services that deal with job submission, file catalogue, scheduled transfers, users, authentication and so on, and site services that manage local site resources (jobs, storage and software packages).

5.1 Gathering monitoring information

A MonALISA service instance is installed on each VoBox (Figure 7) and on each of the machines running central services. These services are used to collect monitoring information from each active job, transfer or service and make available to any client the information in raw or aggregated form. For example the full information about each job is available on demand, but values like number of jobs in each state or total CPU power consumed by all the jobs running on that site are also available.

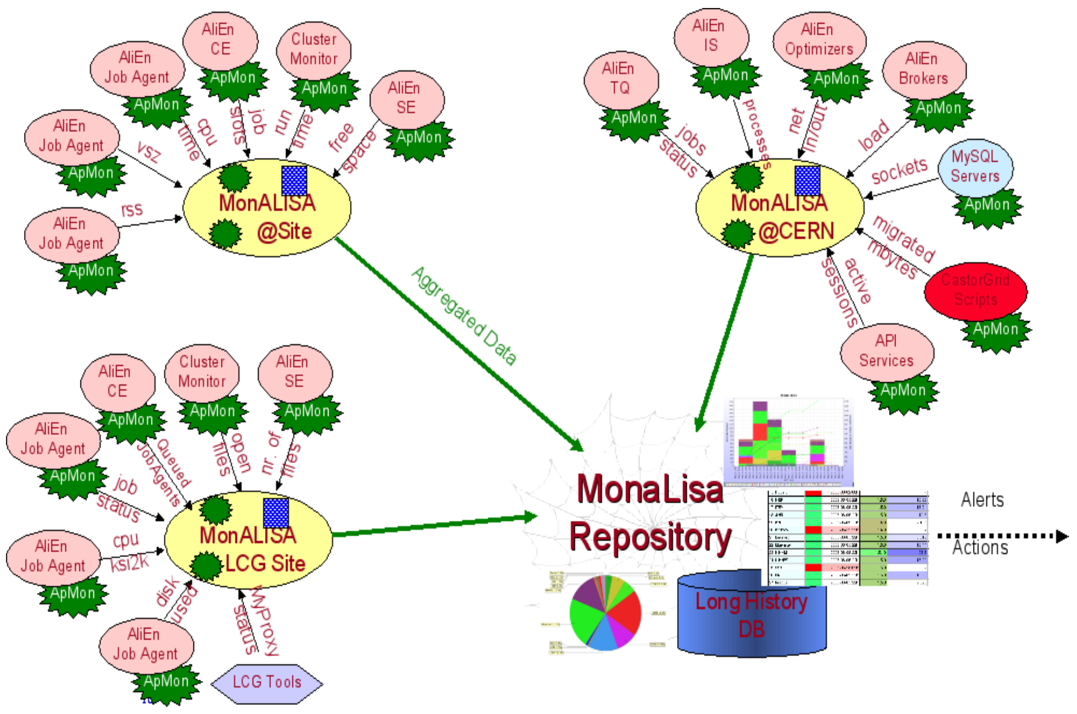


Figure 8 The ALICE monitoring architecture. Each site runs a MonALISA services that collects data from the local AliEn services (SE – Storage Element, CE – Computing Element), performs complete monitoring of the local computing cluster and receives monitoring information from all the jobs running. The Global MonALISA repository collects aggregated information from all the local services.

Most of the data is gathered in a *push* model, with services listening for events and each entity sending the monitoring parameters through the ApMon [1] library to it. This library is available in all the main programming languages so monitoring individual applications is easy no matter the development language that is used (for example jobs mostly use the C or C++ library, services are written in Perl while bandwidth testing applications use the Java version). The library, no matter the language, provides the option to send basic host monitoring data along with the application-specific data, so a lot of monitoring parameters are available for every component in this distributed environment.

Other modules are of *poll* type, periodically querying the status of resources that are not instrumented to automatically report monitoring information or in more complex cases. For example each AliEn service's status is polled periodically, but the service is not only checked to be still active in memory but also a specific functional test is executed for each of the services to determine if it is actually capable of servicing clients.

Third types of modules are *agents* that are running inside each MonALISA service, execute autonomous or collaborative tasks are report results through the same interface. An example is the tracepath module that identifies the network path and latencies between all pairs of MonALISA services in the group. Presently in ALICE there are 85 site MonALISA services and 20 central machines, publishing between 350K and 1M parameters (depending mostly on how many jobs are running at a given time), with a overall update frequency of 4 to 10KHz.

5.2 Collecting and storing monitoring data

Each service has an internal memory buffer that holds the recent history and thus is capable of answering history requests from MonALISA clients or Web Services-based clients. However, for long-term data archival a dedicated MonALISA *Repository* is collecting the most relevant portions of this data (usually aggregated values) and stores it in a relational database. The data is then available for analysis through a web interface [10] or through Web Services queries. The web interface is mostly presenting historical evolution of similar parameters (like the number of concurrently running jobs in each site at each point in time) and current status of a matrix of parameters (such as the status of each service on each VoBox). Particular views include the geographical overview of the sites, reports of consumed resources in the last day/week/month (for accounting purposes) and various dashboards (site-centric, system overview for the ALICE Offline Shifter, data production statistics and many others). The Figure 9 shows the evolution in time of the number of concurrently running jobs in site.

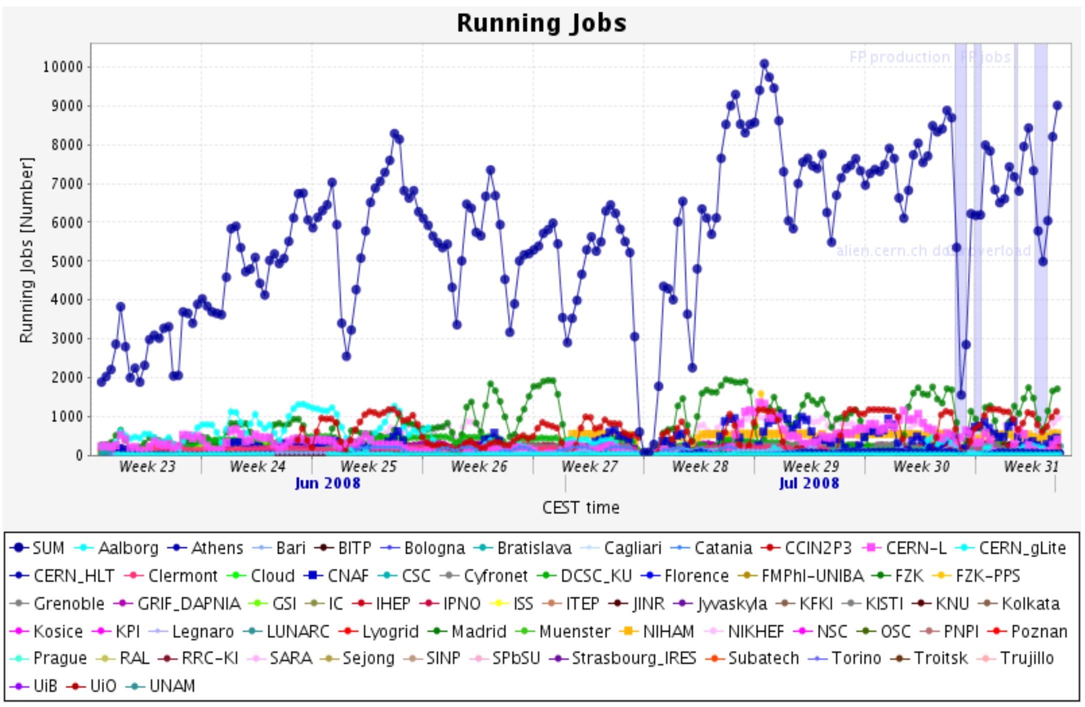
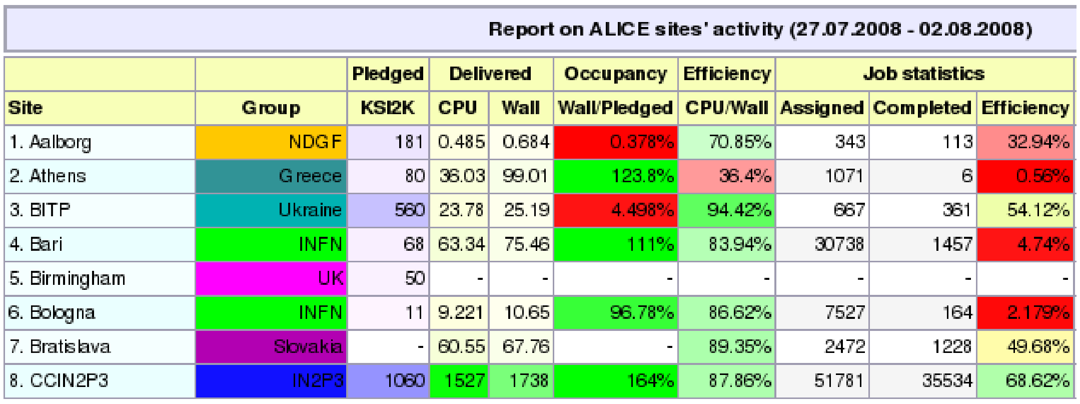


Figure 9 Overview of concurrent running jobs in each ALICE site

Synthetic views like in the Figure 10 show weekly report on sites' performance, highlighting utilized vs. pledged resources, average job efficiency and job completion ratio help spotting systematic problems such as very low performance of a file server, problems with local workload management services, incorrect configurations and so on. All the monitoring information is publicly available, without restrictions. However, a separate section of the site is dedicated to administrators, where access is restricted based on SSL certificates and AliEn user accounts and permissions. This interface is used to control remote site services (described in more detail in the next section), steer the MonteCarlo data production activity or define new software packages.



Report on ALICE sites' activity (27.07.2008 - 02.08.2008)

		Pledged	Delivered		Occupancy	Efficiency	Job statistics		
Site	**Group**	**KSI2K**	**CPU**	**Wall**	**Wall/Pledged**	**CPU/Wall**	**Assigned**	**Completed**	**Efficiency**
1. Aalborg	NDGF	181	0.485	0.684	0.378%	70.85%	343	113	32.94%
2. Athens	Greece	80	36.03	99.01	123.8%	36.4%	1071	6	0.56%
3. BITP	Ukraine	560	23.78	25.19	4.498%	94.42%	667	361	54.12%
4. Bari	INFN	68	63.34	75.46	111%	83.94%	30738	1457	4.74%
5. Birmingham	UK	50	-	-	-	-	-	-	-
6. Bologna	INFN	11	9.221	10.65	96.78%	86.62%	7527	164	2.179%
7. Bratislava	Slovakia	-	60.55	67.76	-	89.35%	2472	1228	49.68%
8. CCIN2P3	IN2P3	1060	1527	1738	164%	87.86%	51781	35534	68.62%

Figure 10 Weekly reports on sites' performance

Monitoring data is stored in a PostgreSQL relational database. For long term archival of monitoring data we use the following reduction schema:

- last 2 months at 2 minutes resolution;
- last year with 30 minutes resolution;
- forever with 2.5 hours resolution

This allows inspection in high detail for recent history while saving space for long-term archival. Currently we have reached 150GB of data for a history of 2.5 years. Each point in time is represented in ~100bytes, yielding about $1.5*10^9$ entries in the database. The ALICE repository is serving ~12 000 dynamic page requests per day and is receiving ~ 10 million new measurements per day.

The repository also runs filters that generate system-wide aggregated values, correlating values received from each site (like number of running jobs, resources or network traffic between them). The original values can be kept in the database along with the aggregated values or simply discarded if the particular details are not interesting for the end users.

This component is also responsible for running system-wide functional tests, such as remote storage testing by actually adding/removing files to each storage element, or requesting execution of network bandwidth tests between pairs of sites.

5.3 Control and automatic actions

Based on the monitoring information received from sites, the repository can take various automated actions to cope with the events. The simplest action is sending an alert (by email or instant messenger) when something has stopped working (an AliEn site or central service, storage element, critical machines and so on). From the web interface anybody can subscribe to receive notifications in each category and (in some cases) for each particular site or for all sites.

In addition to sending emails to the subscribed email addresses, an event is also made available as RSS/Atom feed and history charts are automatically annotated with these events (where it makes sense), so correlations between peaks or lows of activity and events for example can be easily made.

Another kind of automated action is maitenance of central services load balancing. The load balancing is DNS-based, with the repository updating the CERN DNS servers with the list of IP addresses where various central services are available to service remote site services/jobs/users etc. The lists are dynamically generated, so newly added services that pass the functional tests are automatically added to the load balancing, while non-responding or overloaded services are imediatelly removed. So this function not only maintains the load-balancing but also protects the central services from a flood of requests.

The repository is also responsible for the MonteCarlo data production. By continuously monitoring the number of queued jobs in the system and reacting when the number goes below a threshold, new production jobs are automatically submitted in AliEn and jobs finishing in various error states are resubmitted for execution (Figure 11). The jobs to be executed are taken from a list of production requests made by physics working groups, according to the relative priority between the requests and keeping track of the number of events that were produced for a given job type.

From this central point the site services are also controlled. By an SSL channel established between the repository and each remote service, failing AliEn services are automatically restarted, but with additional constraints like making sure that all the central services are functional. This feature, combined with email alerts sent to site administrators and Grid operators when problems cannot be automatically solved, has improved a lot the overall availability of the system.

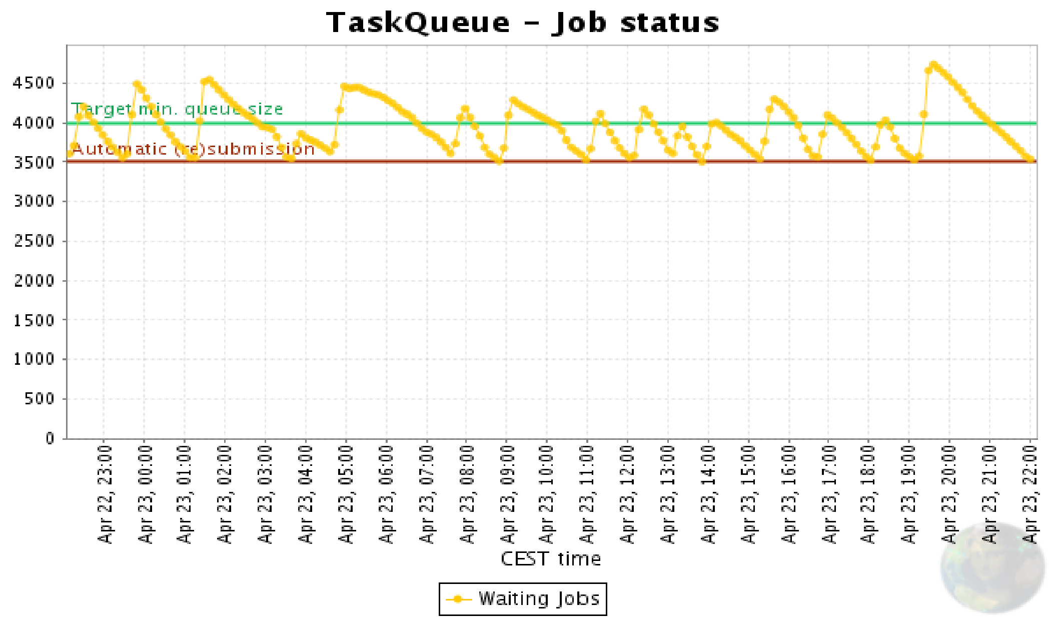


Figure 11 The MonALISA control agents automatically submit new Monte Carlo production jobs when a low threshold (red line) in the number of queued jobs is detected.

6. Summary

The MonALISA services currently deployed are used by the HEP community to monitor computing resources, running jobs and applications, different Grid services and network traffic. The system is used to monitors detailed information on how the jobs are submitted to different systems, the resources consumed, and how the execution is progressing in real-time. It also records errors or component failures during this entire process. MonALISA and its APIs are also used for the CMS dashboard, and by all the job submission tools for analysis and production jobs. MonALISA also is used to monitor, control and administer all of the Enabling Virtual Organizations (EVO) [11] reflectors, and to help manage and optimize their interconnections.

As of this writing, more than 350 MonALISA services are running throughout the world. These services monitor more than 20,000 compute servers, and thousands of concurrent jobs. More than 1.5 million parameters are currently monitored in near-real time with an aggregate update rate of approximately 15,000 parameters per second.

This information also is used in a variety of higher-level services that provide optimized grid job-scheduling services, dynamically optimized connectivity among the EVO reflectors, and the best available end-to-end network path for large file transfers. Global MonALISA repositories are used by many communities to aggregate information from many sites, to properly organize them for the users and to keep long term histories. During the last year, the repository system served more than 10 million user's requests.

References

[1] MonALISA web site http://monalisa.caltech.edu

[2] I.C. Legrand, H.B. Newman, R. Voicu, C. Cirstoiu, C. Grigoras, M. Toarta, C. DobreMonALISA: An Agent based, Dynamic Service System to Monitor, Control and Optimize Grid based Applications, CHEP04, Interlaken, Switzerland, 2004

[3] Java http://java.sun.com and JINI : http://www.jini.org

[4] USLHCnet web page : http://www.uslhcnet.org/

[5] http://www.ciena.com

[6] http://www.glimmerglass.com

[7] http://www.calient.com

[8] Fast Data Transfer : http://monalisa.cern.ch/FDT/

[9] ALICE – ALIEN web site : http://alien.cern.ch/

[10] The MonALISA repository for ALICE : http://pcalimonitor.cern.ch/

[11] EVO web Page : http://evo.caltech.edu