MonALISA, an agent-based monitoring and control system for the LHC experiments

J Balcas, D Kcira, A Mughal, H Newman, M Spiropulu, J R Vlimant
California Institute of Technology, Pasadena, CA 91125, USA
E-mail: monalisa@hep.caltech.edu

Abstract. MonALISA, which stands for Monitoring Agents using a Large Integrated Services Architecture, has been developed over the last fifteen years by California Insitute of Technology (Caltech) and its partners with the support of the software and computing program of the CMS and ALICE experiments at the Large Hadron Collider (LHC). The framework is based on Dynamic Distributed Service Architecture and is able to provide complete system monitoring, performance metrics of applications, Jobs or services, system control and global optimization services for complex systems. A short overview and status of MonALISA is given in this paper.

1. Monitoring distributed systems
When monitoring and managing large distributed data processing facilities, it is essential to be able to have information about the facilities themselves, the computing and storage systems, the networks as well as the workflows and applications running on them. Furthermore, it is crucial to have a monitoring system which can function in near real-time.

The monitoring information gathered for all the different subsystems can be used for the design, modeling, debugging, accounting and the development of higher level services. These services can provide decision support, some degree of automated decisions as well as maintain/optimize workflows in large scale distributed systems.

2. The MonALISA architecture
The MonALISA system is designed as an ensemble of autonomous, multithreaded, self-describing agent-based subsystems, which are registered as dynamic services, and are able to collaborate and cooperate in performing a wide range of information gathering and processing tasks [1, 2, 3]. These agents can analyze and process the information in a distributed way in order to provide optimization decisions in large scale distributed applications. An agent-based architecture provides the ability to add to the system increasing degrees of intelligence, to reduce complexity and make global systems manageable in real time.

The scalability of the system derives from the use of a multithreaded execution engine to host a variety of loosely coupled, self-describing dynamic services or agents and the ability of each service to register itself and then to be discovered and used by any other services or clients that require such information. The system is designed to easily integrate existing monitoring services into the overall framework.

1 monalisa.caltech.edu
Figure 1. The four logical layers of the MonALISA architecture, from top to bottom: 1. high level services, 2. proxies, 3. distributed agents, 4. lookup services.

tools and to provide the monitoring information in a dynamic, customized, self describing way to any other services or clients.

MonALISA is written in Java and on respective technologies, like JINI [4]. A schematic of the four logical layers of the MonALISA architecture is shown in Figure 1. We will describe here shortly the layers from top to bottom.

The first (top) layer is that of the regional or global high-level services (HLS), repositories and clients. These are the consumers of the information and allow to store it, make decisions and allow to do automation. The HLS use a predicate mechanism for requesting or subscribing to monitoring information from the services. The predicate mechanism uses a syntax similar to regular expressions.

The second layer of the architecture is that of the proxies. The use of proxies allows for secure and reliable communications, dynamic load balancing, as well as scalability and replication. The use of proxy services allows for scalability by allowing a service to send the data once and multiplexing it for all the clients that subscribe to that information.

The third layer is that of the services. The services perform the monitoring tasks and use a multithreaded execution engine. In addition, loosely-coupled agents are used to analyze and collect information in real time both locally and globally in collaboration with other agents or services.

The fourth layer is that of the lookup services (LUS). It consists of a network of services that provide dynamic registration and discovery for all the other components described above. MonALISA services are able to discover each other in the distributed environment and to be discovered by the interested clients. Each MonALISA service registers itself with LUS as part of one or more groups and it publishes some attributes that describe itself. In this way any interested application can request MonALISA services based on a set of matching attributes.
The registration uses a lease mechanism. If a service fails to renew its lease, it is removed from the LUS and a notification is sent to all the services or other application that subscribed to such events. Remote event notification is used in this way to get a real overview of this dynamic system.

Lookup services have replicated information. It is important for the monitoring service to be registered in two or more distributed lookup services, because if one fails to respond, interested clients can find the MonALISA services registered in the other working LUS. Thus, the single point of failure problem can be avoided and a more reliable network for registration of services can be achieved in the distributed environment. The JINI technology used allows dynamically adding and removing Lookup Services to and from the system.

The MonALISA architecture described above is fault tolerant and has no single point of failure. All the services are replicated and the dynamic registration takes care of automatically updating the available components.

In addition to its own monitoring modules, MonALISA contains many modules that allow integration of popular monitoring tools like Ganglia, Nagios, MRTG, Spectrum, HTCondor, Panda, etc. New modules can be easily added by implementing a simple Java interface.

3. Multithreaded execution engine
The MonALISA service is the basic component of the whole framework. It consists of an ensemble of multi threaded independent subsystems. These subsystems: monitor a large number of entities; filter and aggregate the monitoring data; store monitoring information for shorter periods of time; manage web services for direct data access, provide triggers, alerts and actions based on the monitoring data; control the system using dedicated modules.

The multi threaded system is depicted schematically in Figure 2. It allows to perform many independent data collection tasks in parallel. The necessary monitoring modules are dynamically loaded and executed on independent threads. MonALISA creates a dynamic pool of threads once
and reuses the threads when the assigned tasks to them are completed. This is done in order to reduce the load on the system. Furthermore, this allows to run concurrently a large number of modules and adapt to the load and response time of the different components of the system. For tasks run periodically, a priority queue is used.

In this design, failure of a monitoring task does not delay the other tasks as they run on independent threads. This system is able to monitor a large number of heterogeneous nodes with different response times.

4. Application monitoring using ApMon
A dedicated API, ApMon, was developed to allow sending customized monitoring information into the MonALISA system. It consists of a separate library that integrates with MonALISA. ApMon uses the UDP protocol to transport the information with an additional sequence number to allow for verification of integrity of the monitoring reports. ApMon can be dynamically configured from remote services and from local configuration files.

Different implementations of the ApMon library are provided with the MonALISA package for some main programming languages: C, C++, Java, Perl, Python. ApMon is used by both the CMS and ALICE experiments at CERN to monitor tens of thousands of running jobs in their worldwide distributed computing infrastructure.

5. MonALISA today
Presently MonALISA runs 24x7 on more than 370 sites with more than 100K servers. It also monitors more than 100 links of major computer networks. MonALISA gathers and visualizes information on about tens of thousands of grid jobs running concurrently as well as 14K end-to-end network path measurements. MonALISA uses intelligent agents, as described above. It collects 6M persistent parameters in real time and deals with 100M of volatile parameters per day. It presently updates 35K parameters per second and the repository servers deal with 10M user requests per year.
MonALISA is a central component of the ALICE distributed computing environment\(^2\) [5, 6]. Furthermore, Monalisa has been used over many years to monitor and steer high-throughput data transfers at the competitions and demonstrations of the Caltech CMS group at the Supercomputing conference over the last decade\(^3\).

6. Summary
MonALISA provides a unified platform for monitoring information for local and distributed systems. It has a service-oriented architecture as well as dynamic discovery and subscription of its services and information sources. It has a powerful multi threading execution engine and a variety of monitoring modules together with interfaces to other popular monitoring packages [1].

MonALISA has been developed at Caltech in the past 15 years. With its solid architecture and design it is used in many data centers and facilities around the world for computing, storage and network connections monitoring.

Acknowledgments
The work presented here was funded mainly by the U.S. Department of Energy through several projects in the past decade. The authors would like to acknowledge the support of the software and computing program of the CMS and ALICE experiments at the Large Hadron Collider.

References
[1] I C Legrand, “Monitoring and control of large-scale distributed systems”, in Grid and Cloud Computing: Concepts and Practical, Applications, Carminati, F., Betev, L., Grigoras, A., ISBN, 978-1-61499-642-2, 2016
[2] I. Legrand, H. Newman, R. Voicu, C. Grigoras, C. Cirstoiu and C. Dobre, “MonALISA: A distributed service system for monitoring, control and global optimization,” PoS ACAT 08, 020 (2008).
[3] H. B. Newman, I. C. Legrand, P. Galvez, R. Voicu and C. Cirstoiu, “MonALISA : A Distributed monitoring service architecture,” eConf C 0303241, MOET001 (2003) [cs/0306096 [cs-dc]].
[4] B Venners, “Jini: New technology for a networked world.” JavaWorld (http://www.javaworld.com) (1999).
[5] F. Carminati, “The ALICE computing project,” CERN Courier 52N3, 31 (2012). cern-courier.com/cws/article/cern/49059
[6] M Meoni, “Monitoring of a distributed computing system the grid alien cern.” Master’s thesis, University of Florence (2005). monalisa.caltech.edu/docs_marco_thesis.pdf

\(^2\) alimonitor.cern.ch
\(^3\) supercomputing.caltech.edu