

Understanding WLCG Monitoring

An introduction to WLCG monitoring for grid service developers

James Casey <james.casey@cern.ch>

Ian Neilson <Ian.Neilson@cern.ch>

Overview

WLCG is currently trying to improve the reliability of the overall WLCG service. Part of this effort is by improving the monitoring of the underlying grid services deployed at the site. The aim of this document is to introduce the WLCG monitoring framework to grid service developers, who, up to this point, may not have considered the monitoring of their services, and its implications.

After reading this document, a Grid Service developer should

- understand the terms used to describe monitoring in the WLCG context
- understand how to document their service in terms of metrics that should be gathered

Site and Remote Monitoring – a birds eye view

The entire effort is predicated upon the fact that site administrators will use a local fabric monitoring system to find and diagnose problems with services running on their site.

Examples of fabric monitoring systems that are used currently with WLCG are LEMON¹, Nagios² and Ganglia³.

As well as the local fabric, another aspect of monitoring is remote monitoring – this is where the site is “probed” from outside by an external entity, e.g. the WLCG Service Availability Monitoring (SAM)⁴ system. The WLCG approach to remote monitoring is that information gathered by remote monitoring systems should be made available to site administrators via a programmatic interface (e.g. XML/HTTP) for integration into their local fabric monitoring system.

¹ <http://lemon.web.cern.ch/lemon/index.shtml>

² <http://www.nagios.org/>

³ <http://ganglia.sourceforge.net/>

⁴ <https://lcg-sam.cern.ch:8443/sam/sam.py>

Service monitoring consists of gathering time stamped **samples** of various **metrics** from the service. We consider two types of metrics: Availability and Performance. The reason for this split is that different fabric monitoring systems may handle one or the other category (e.g. Nagios for Service availability and Ganglia for performance monitoring).

Service Availability

Monitoring of availability consists of checking if the service is running, available to accept user interaction in a normal method. An availability metric takes on one of a small set of enumerated values signifying the state the service is currently in e.g. "Available", "Degraded", "Unavailable".

Performance Monitoring

Monitoring of performance consists of gathering metrics that could be useful in diagnosing the behavior of a service, e.g. Load, #threads in use, #jobs/transactions processed per minute.

Metric Categorization

We have identified three categories of metrics that can be gathered. The categorization is based on the type of information that the metric represents. Each category has some differences also in how it is collected, which has implications for the service developer. Since it is hard to assign names to the categories, we simply call them **Category A, B, C**, with a descriptive name associated with each.

Category A ("Traces in the fabric")

There is a lot of information about the service that can be gathered by directly examining the supporting fabric using standard operating system level tools. It is possible to build both availability and performance metrics for a service from this information. Also, note that these metrics usually need a local **agent** running on the node in order to be collected.

Examples of such information include:

1. Is a given process running
2. What processes are attached to what network port
3. Availability of space in a file system
4. Existence and permissions on a given file (e.g. a log file)
5. Appearance of a string/regular expression in a given file
6. load on the machine

From this information, relevant metrics could, for example, be:

1. # of processes called 'tomcat4' running under the user id 'tomcat'
2. the process 'tomcat4' should be listening on port 8443
3. /var should have at least 5% free
4. The directory /var/log/tomcat5 should exist, and be writable by the user 'tomcat' and group 'tomcat'
5. The number of occurrences of the string 'ORA-' in the last 5 minutes in the file /var/log/tomcat5/catalina.out
6. Average load on the node in the last five minutes

Category B (“Interpretation of a user-level operation”)

This category of metrics is gathered by running an operation a user might want to do, and parsing the output to interrogate the status of the service. Examples are:

1. “ping”-type operations which check if the service is listening on a port correctly
2. 'lfc-ls /grid' on an LFC Catalog to check results are being returned from a database correctly
3. Submitting an FTS transfer job, and monitoring its progress and termination state.

It is important to note that often such metrics can be gathered either remotely, or locally on the node (or both!)

Category C (“Internal metrics presented by the service”)

Some services provide a mechanism to obtain more detailed information about the internal state of the service. This could be via an API call in its public API, an appropriate HTTP request to a web service or via an accounting file stored locally on the file system.

Examples of possible metrics in this category are:

1. An LFC service can report, how many of it's threads are currently serving requests
2. An R-GMA MonBox provides a HTTP request an XML document, which contains the status of all producers and consumers currently connected to it
3. A CE provides, in a file, statistics on how many jobs have terminated successfully in the last 5 minutes.