ROC SAM Nagios Portal Requirements

Judit Novak

March 18, 2009

Abstract

This paper describes a requirements for a new web interface for the Service Availability Framework that adopts changes implied by the fact that it is being migrated to Nagios.

1 Overview

Components of the Service Availability Monitoring framework (SAM) are being migrated to the Nagios monitoring system together with a Messaging System as a transport layer these days.

Since all the underlying architectures will be replaced, currently existing SAM Portal can not be used anymore. A new web interface has to be shipped together with the new architecture to *display data* stored there. The interface is designed in principal for ROC Managers, in order to help them *debugging* attached sites.

This document describes basic requirements against this new interface.

2 Current SAM Portal

There is a web portal already existing for the SAM Framework, which is an important interface used for monitoring

- ROCs
- sites
- resources

For a better understanding about what is needed about the portal to be developed, first let's take a look at the usage of the already existing one (Figure 1).

2.1 Users

Since for all software tools requirements are mostly defined by their user groups, we are identifying requirements below based on who the users of the SAM Portal are.

2.1.1 ROC and Site Admins, GRID Operators

The current SAM Portal was strongly designed for ROC and Site Administrators, to monitor and to debug sites, and this will be also true for the future one.

Req-1: These users need views that are *easy to follow* and understand, and which allow problems to be spotted rapidly.

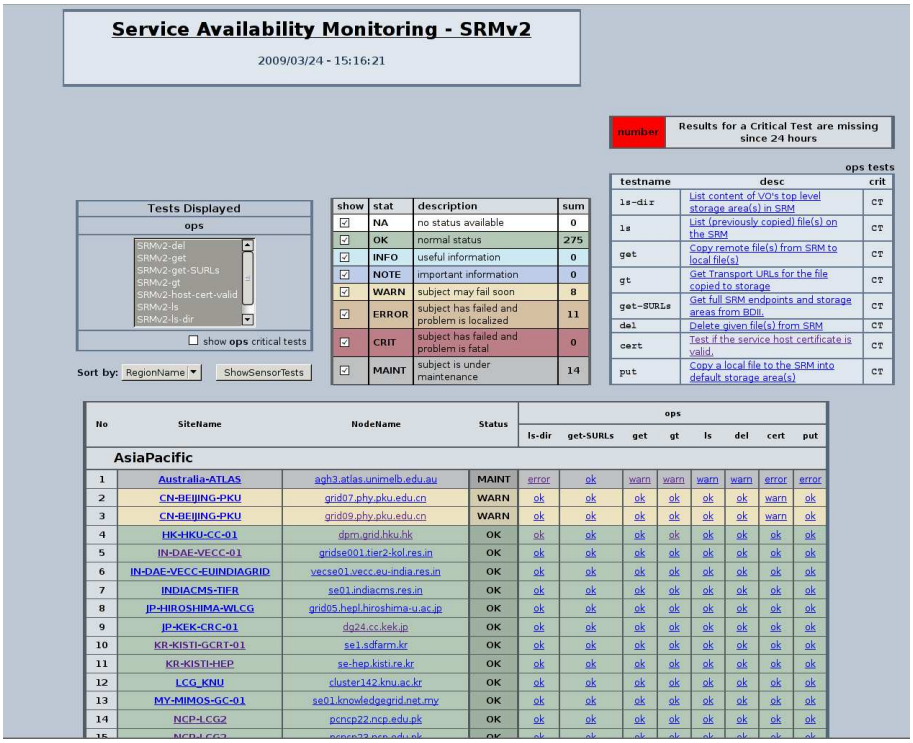

Figure 1: Latest status listed for AsiaPacific ROC's CEs

Table views of the SAM Portal contain sufficient amount of information (not too little, not too much) with colors that help to quickly recognize actual resource statuses.

Req-2: Quick and lightweight access to *recent status* of resources and the tests that indicated the overall status.

This helps to quickly identify the direction where problems could come from. Further than the overall resource status, it's useful to see the status of the individual tests. This allows to find the problematic tests immediately, the ones that caused a failure status for the site.

Req-3: Similarly lightweight views must present *detailed data* of recent tests.

This is to help debugging the sequence of commands the test consists of, to see all commands executed in the most verbose way possible, and to find the error messages.

UseCase-1: To present a use-case, take the following real-life example. WLCG Working Group Site Nagios installation is used at the site. This tool raises alarms, thus notifies Site Administrators, once a SAM Test failure for the site was detected. Site Administrator goes to the SAM Portal Latest Results page for the concerned service, and looks for the row that has information about the node. The status line color indicates the failure, and among the critical tests one appears with an 'error' status. Clicking on the 'error' box, the Site Admin is redirected to the detailed test result page, where (s)he can search for the reason for the failure.

UseCase-2: As for the ROCs, SAM Portal is used in the process of certifying new sites. Having the SAM tests submitted to the site (for instance via the SAM Admin Portal [3] tool), the ROC Managers have to go to the SAM Portal to see if the site passed the tests successfully. If it didn't, the ROC Manager might also look at the detailed results of the tests that failed, and suggest the Site Admin, how to fix it.

Req-4: It's useful for the ROC to see resources of a particular service within the region in *one list*, as on current SAM Portal.

Issue-1: On the other hand, there is no way for a Site Administrator to see all the *resources of the site on one page*. SAM Portal doesn't provide a view where overall status of sites within the ROC would be displayed.

The **Req-4** must be part of new the interface, **Issue-1** would be nice to have, but not obligatory.

2.1.2 Grid Operators

The way sites and ROCs are notified about failures happens by receiving a ticket submitted by the Grid Operators who are on duty on that particular week. Grid Operators are also main users of the SAM Portal.

UseCase-3: Doing their weekly duty, operators often want to take a closer look about the problems that raised an alarm for a resource. Sometimes it occurs, that an alarm was raised in the past, but gone by the time, the operator looks at it. Then (s)he has look at the past few hours or days, and analyze test results for the resource (is the problem coming-and-going, did the Site Admin apply a fix, etc).

When notifying a site about a failure that was discovered, the operators refer to the corresponding SAM Portal pages in the notification e-mails.

Req-5: For Grid Operators, further than what was needed by the previous users group, it's absolutely necessary to have *precisely referable links* to particular views of the portal.

2.1.3 Service Experts

Though much fewer, yet very important users are the Grid service experts. What they need is rather an overview about "what's going on" in the Grid. They want to find coherence, connections between failures within a region, or the whole Grid.

UseCase-4: There is a high number of failures for Computing Element tests. The expert looks at all CEs in the Grid on the SAM Portal's corresponding display. High number of failures is well-reflected on the SAM Pagesi by the summary counters for resource statuses. (S)he finds that most of the failures were indicated by Replica Management tests and come from one particular region. This gives the assumption that the ROC's top-level BDII was suffering of a downtime (as it has to return correct response for successful replication). (S)he can confirm the assumption by looking at the detailed test results of a few resources in the region.

Req-6: Service Experts need to see *summary numbers* of resources in a particular status.

Req-7: They need *comparative overview* of actual site/resource statuses, which includes overall view of resources and tests statuses.

Rcmd-1: For the time being it's unsure, if the future SAM Portal will also have a Grid-wide installation, or Service Experts will have to use another tool. In the software design, the possibility displaying information about the *whole Grid* should be kept open.

2.1.4 Not for Project Managers

The way information is presented for users above is very much different from what for instance Grid Project Managers want to see. Information that is crucial for Site Administrators, like error messages appearing in the test outputs are out of interest for Project Managers.

Req-8: They need *high level views* that show summaries on different levels (site, service, etc.) regardless of details behind.

UseCase-5: Project managers monthly evaluate availabilities of sites, based on summary graphs presenting overall site status for the past month.

Development and design of the currently existing SAM Portal didn't into this direction. These are entirely different from the goals defined by SAM Portal users. Having a separate tool developed for this purpose was a much better solution. The corresponding web interface, Gridview [1] was developed by colleagues from BARC, India.

Rcmd-2: Would be a nice feature if future SAM Portal could support these type of *high-level summary* views.

2.2 Advantages, disadvantages

Current SAM Portal has a couple of features that make it handy to use. On the other hand, there are also a few things to improve (or to be avoided) in the new version to come. The former group defines further requirements for the future portal, while the latter identifies issues.

2.2.1 Features of the existing portal that should be kept

Feat-1: Using the interface is rather *intuitive, straightforward*.

Navigation within the SAM pages needs no special knowledge, neither to understand information displayed on the portal. Information is presented in a simple way, so users would be able to follow up by a short glance what they were looking for.

Feat-2: While the pages are loading, information is displayed *incrementally*, as it is being retrieved from the database.

This way, users can browse part of the data, while the rest is still loading. Users feedback proved the usefulness of this feature.

Feat-3: As from the *security* point of view, the SAM Portal code proved be resistant against attacks.

Feat-4: The interface has exact, copy-paste-able, bookmarkable *direct links* to every view.

Further than the Grid Operator mails, in many other cases it's also very useful to have this feature. Site Admins for instance can just bookmark the history pages for resources on their sites, and check them from time-to-time.

Feat-5: The SAM Portal provides particularly useful and *suitable views*.

An example is the Grid-wide list of resources belonging to a service. Also, a similar list is available (**Req-4**) with the scope of a ROC. The latter is handy to use if no more information needed, as this one loads faster than the former. Status of resources in this list is indicated by color.

A row in the list contains the status of latest tests that were run on the resource.

2.2.2 Identified short-comings

Issue-2: Unfortunately, the SAM Portal is *lacking views* that would be important to have.

In particular it was many times stated that a page where a site's resources latest status would be listed (**Issue-1**) is strongly anticipated. This missing view is a design problem, and would require significant work to get it integrated in the current SAM Portal.

Issue-3: SAM Portal history pages give a *confusing output* for tests submitted individually.

The SAM Portal inherited the assumption about test submission method, that was true in the old version of the tool (Sites Functional Tests). Namely that tests for a resource are submitted in a bunch, and not individually. SAM Portal history pages gave a very misleading output, since the first tests out of the main, grouped submission appeared especially for the Computing Elements.

Having Gridview history graphs embedded recently to the SAM Portal history pages, that provide a much more suitable view, improved this situation.

Issue-4: Especially to generate history pages there are a few very *heavy database queries* in the portal's DB module.

The original idea was to rather put the load on the database, then on the application. DB queries were prepared so that they return data that can be basically displayed with not much of further logic applied. But especially having strongly growing tables over time, this proved not to be the best approach. The SAM database often suffered of high load, occasionally because of these queries.

Cnstr-1: Fixing problems listed above or adding the missing features would require a *significant time*, and *serious changes* in the portal's design. The amount of work and changes to be done are so much, that it's questionable if it worths to keep patching the existing version and not to restart development from scratch.

2.3 Changes implied by migration to Nagios

The new system that takes over the role of the SAM submission framework (and much more) brings changes to the SAM framework in many terms.

Cnstr-2: The *structure* of the SAM Database will entirely change, together with the way to *communicate* with it. Neither the approach how the SAM portal is communicating with the DB, nor the tables and columns queried by the SAM Portal could be used in the future.

Req-9: An interface that works with the *Nagios-based architecture* is needed.

Customizing the current SAM Portal so that it would adopt these changes might not be even be possible. Constructing a new interface seems to be much more reasonable in this case.

Req-10: A more *ROC and site-oriented* view would be more suitable for future needs.

Emphasis from a central SAM submission will rather move to the ROCs. Still there will be tests sent out centrally to Grid resources, but most of them will be submitted by ROCs to their sites.

3 Requirements

Below we are listing the most important requirements against the new SAM Nagios interface.

3.1 Functional requirements

REQ-1: Support for data coming from Nagios (together with the messaging system as a transport layer). (**Req-9**)

REQ-2: Displays customized for ROC and site-level usage. (**Req-10**, **Feat-5**) Up-to-date summary status views (**Req-2**) on hierarchical levels below should be supported.

- organizational units (ROCs (**Req-??**), sites (**Issue-??**, **Issue-2**))
- services
- resources

REQ-3: Depending on the scale of the installation of the portal it would be good to have Grid-wide views as well both for

- high level summaries (**Rcmd-2**)
- comparative detailed status lists (**Req-7**, **Rcmd-1**)

Both views should come with summary numbers (**Req-6**)

This item has a *low priority*, implementation is rather optional than obligatory.

REQ-4: Tests detailed data views (**Req-3**).

REQ-5: Easy navigation (less clicking possible).

REQ-6: Straightforward, clear, easily browseable displays, optimized for the user communities. (**Req-1**, **Feat-1**)

REQ-7: Historical data display that is quick to load (**Issue-3**).

REQ-8: Test documentation linked from the portal (together with hints on how to fix problems reported by the tests).

REQ-9: DN-based authentication.

REQ-10: Would be useful, but not crucial to have different views of test data (tables, graphs, etc.) both for latest results and historical data.

3.2 Technical requirements

REQ-11: Avoiding code duplication, but rather using already existing Nagios-related visualization tools.

REQ-12: Parameters passed in the URLs, so direct links could be used (references, bookmarks, etc.) (**Req-5**, **Feat-4**).

REQ-13: Pages must be quick to load (**Feat-2**).

REQ-14: Optimized database structure and queries (**Issue-4**).

In the second part of the document we propose solutions, that correspond to this requirements list.

4 Available Nagios Visualization Tools

There are many Nagios-related tools available today, much of these are actually visualizing Nagios data. We had a look at how these could be used for SAM purposes.

4.1 Native Nagios Pages

Basically all possible information is available on the Native Nagios Pages. The Nagios pages provide many different ways to look at both summary and individual test results. It's even a bit "too much".

Prob-1: There are *long pages* to scroll down, which are *constantly being refreshed* (that also takes time with such an amount of data being displayed at once).

Such views might be confusing and sometimes annoying to use for a ROC or Site Administrators. We need more clear, straightforward and simple displays.

Prob-2: Another problem is that the Nagios have a very *different structure* to organize resources, than what is present in the Grid.

In Nagios terms, for instance both a site and the group of computing elements on the site is presented as a so called servicegroup. Clearly, a ROC or Site Admin wants to see a distinction between these categories. The new SAM interface should be customized for Grid topology structures.

4.2 NagVis

Perhaps the most popular tool to generate a topological view out of Nagios data is NagVis [4]. The main target of this project is to indicate Nagios-measured status of instances over a static map.

Most recent version of the tool also supports the generation of dynamic graphs based on Nagios host-to-host relationships. But unfortunately the model to describe Grid topology in the Nagios database is very different from that, as it includes links between servicegroups (for instance, sites) and resources (hosts).

Prob-3: The complexity of the Nagios description of the Grid topology *can't be handled* by NagVis

4.3 pnp4nagios

pnp4nagios is a very useful tool to create graphs of Nagios performance data. Such graphs should be used by the future SAM portal, for tests returning a single numerical value and to show how OK/failure status (which is a binary value) was changing over time.

Prob-4: On the other hand, SAM probes are in principal *not numerical* measurements.

Prob-5: *More details* about probes must be available, than just the graphical output over time.

Concl-1: Tools described above are very useful, but none of them *can be used "as it is"*.

We either have to collect useful parts of them and integrate those together (perhaps also with components from other, Grid-specific interfaces like Gridview), or we have to come up with a different, new solution.

References

[1] Gridview
http://gridview.cern.ch/GRIDVIEW/same_index.php

[2] GOC Wiki Pages
http://goc.grid.sinica.edu.tw/gocwiki/SiteProblemsFollowUpFaq

[3] SAM Admin Portal
https://cic.gridops.org/samadmin/

[4] NagVis
http://www.nagvis.org

[5] pnp4nagios
http://www.pnp4nagios.org