Grid monitoring for ZEUS

Summer Student Program 2007, DESY

Alexander Chernyak
Moscow State University
Department of mechanics and mathematics

DESY Hamburg
14th September 2007

Abstract

This report describes the work which has been done during the DESY Summer Student Program in 2007. Introduction to Grid computing and its relevancy to ZEUS Monte Carlo production are presented. The motivations for advanced monitoring system are outlined. Implementation of ZEUS monitoring system is described.

Contents

1 Introduction

During the DESY Summer Student Program 2007 I was working for ZEUS experiment. I started from learning basic concepts from Grid Computing. Then I got familiar with the ZEUS Monte Carlo (MC) production system on the Grid.

My main task was to deploy the standard Site Availability Monitoring system (SAM) for Grid. I have extended the monitoring system by introducing ZEUS specific tests. The goal of my work was to improve the efficiency of the MC jobs running on the Grid.

2 Grid

2.1 Definitions

The original idea of Grid computing comes from Carl Kesselman and Ian Foster. In 1998 attempted a definition in the book "The Grid: Blueprint for a New Computing Infrastructure". They wrote: "A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities." Since then a more precise definition were given. At present time previous definitions can be captured in a simple checklist.
According to that the Grid is a system which:

1) *coordinates resources that are not subject to centralized control …*
The Grid integrates and coordinates resources and users that live within different control domains—for example, the user's desktop vs. central computing; different administrative units of the same company; or different companies; and addresses the issues of security, policy, payment, membership, and so forth that arise in these settings. Otherwise, we are dealing with a local management system.

2) *uses standard, open, general-purpose protocols and interfaces*
The Grid is built from multi-purpose protocols and interfaces that address such fundamental issues as authentication, authorization, resource discovery, and resource access. It is important that these protocols and interfaces are *standard* and *open*. Otherwise, we are dealing with an application specific system.

3) *delivers nontrivial qualities of service.*
The Grid allows its constituent resources to be used in a coordinated fashion to deliver various qualities of service, relating for example to response time, throughput, availability, and security, and/or co-allocation of multiple resource types to meet complex user demands, so that the utility of the combined system is significantly greater than that of the sum of its parts.

2.2 EGEE

The biggest project in Europe which deals with Grid for science is EGEE project (**E**nabling **G**rids for **E**-scienc**E**). It brings together scientists and engineers from more than 240 institutions in 45 countries world-wide to provide a seamless Grid infrastructure for e-Science that is available to scientists 24 hours-a-day. Conceived from the start as a four-year project, the second two-year phase started on 1 April 2006, and is funded by the European Commission.

Expanding from originally two scientific fields, high energy physics and life sciences, EGEE now integrates applications from many other scientific fields, ranging from geology to computational chemistry. Generally, the EGEE Grid infrastructure is ideal for any scientific research especially where the time and resources needed for running the applications are considered impractical when using traditional IT infrastructures.

The EGEE Grid consists of over 36,000 CPU available to users 24 hours a day, 7 days a week, in addition to about 5 PB disk (5 million Gigabytes) + tape MSS of storage, and maintains 30,000 concurrent jobs on average. Having such resources available changes the way scientific research takes place. The end use depends on the users' needs: large storage capacity, the bandwidth that the infrastructure provides, or the sheer computing power available.

First applications using the EGEE Grid infrastructure were from the fields of High Energy Physics (HEP) and Biomedicine, but EGEE now also supports applications from many other scientific domains, such as Astrophysics, Computational Chemistry, Earth Sciences, Finance, Fusion, Geophysics, Multimedia. In addition, there are several applications from the industrial sector running on the EGEE Grid, such as applications from geophysics and the plastics industry.

2.3 Grid resources available for ZEUS

ZEUS has access to the Grid resources through its member institutes. ZEUS Virtual Organization (VO) is registered and maintain on the Grid. This gives the access to the basic Grid services for the ZEUS collaboration.

The number of potentially available CPUs amounts to about 8000. In fig.1 you can see a location of computing sites supporting ZEUS VO.

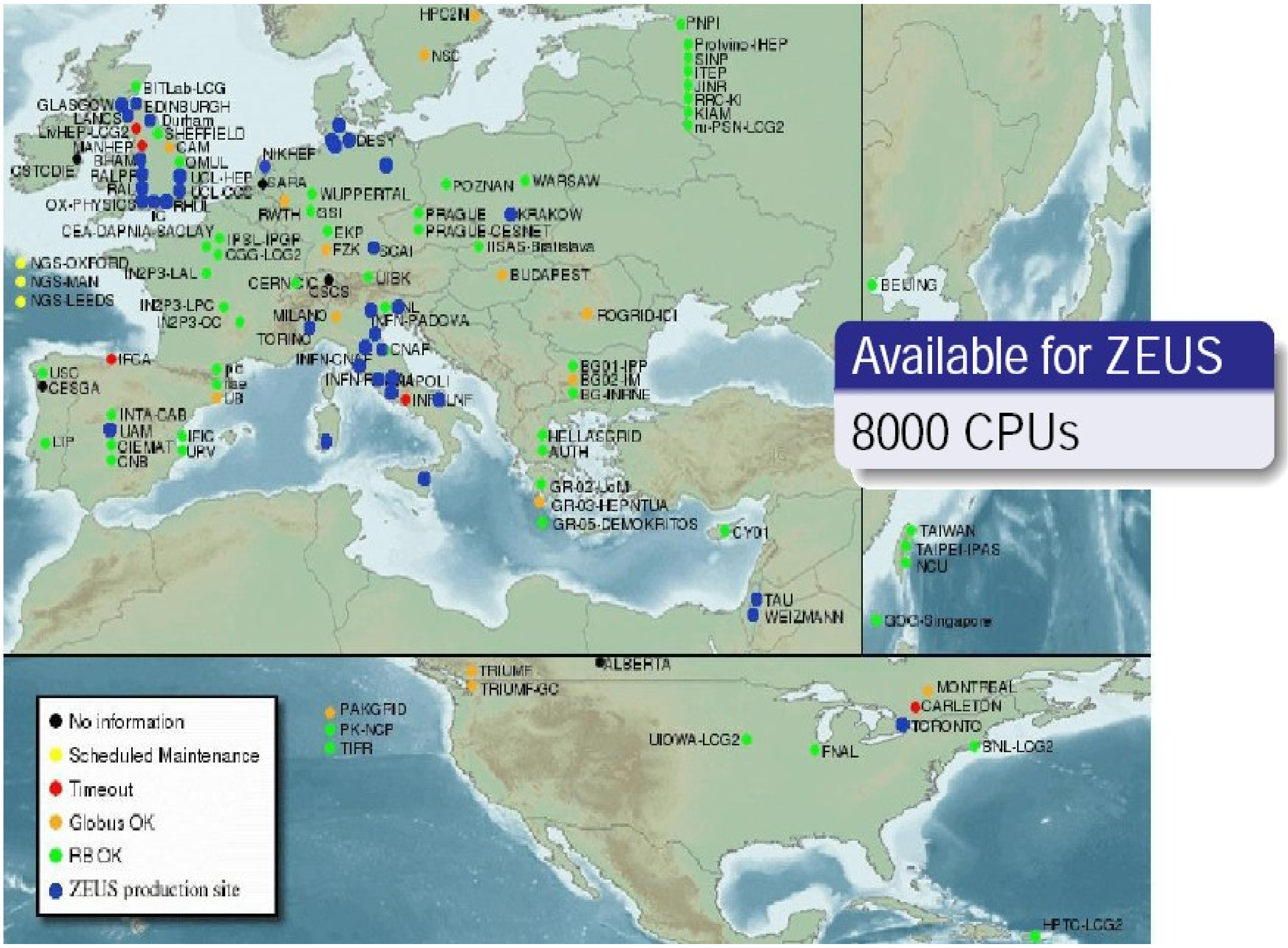


Fig. 1
Sites, accessible for ZEUS, shown as blue dots.

2.4 ZEUS-grid-toolkit

Zeus-grid-toolkit is a software package developed in ZEUS, which can be used for various grid applications. It is written in the Perl language using object oriented style. The main purpose for this software is a separation of the client interface from its actual implementation. In a highly evolving grid-world it is very important that the users' code is decoupled from the tools delivered by a particular service implementation. This package can be seen as an interface to these tools. Currently it implements basic functionality for data and job management and internally uses tools provided by different releases of LCG2. The first application built-in on top of this package is a Monte Carlo production package for ZEUS.

2.5 The Monte Carlo production on the Grid

ZEUS uses the Grid infrastructure mainly for Monte-Carlo (MC) simulation. Simulation requests from users are transformed to the Grid jobs and are processed automatically until the simulation output is stored on DESY mass storage system. The automatic processing is performed using a set of scripts developed for that purpose. These scripts are based on the zeus-grid-toolkit which provides enhanced interface to the Grid services.

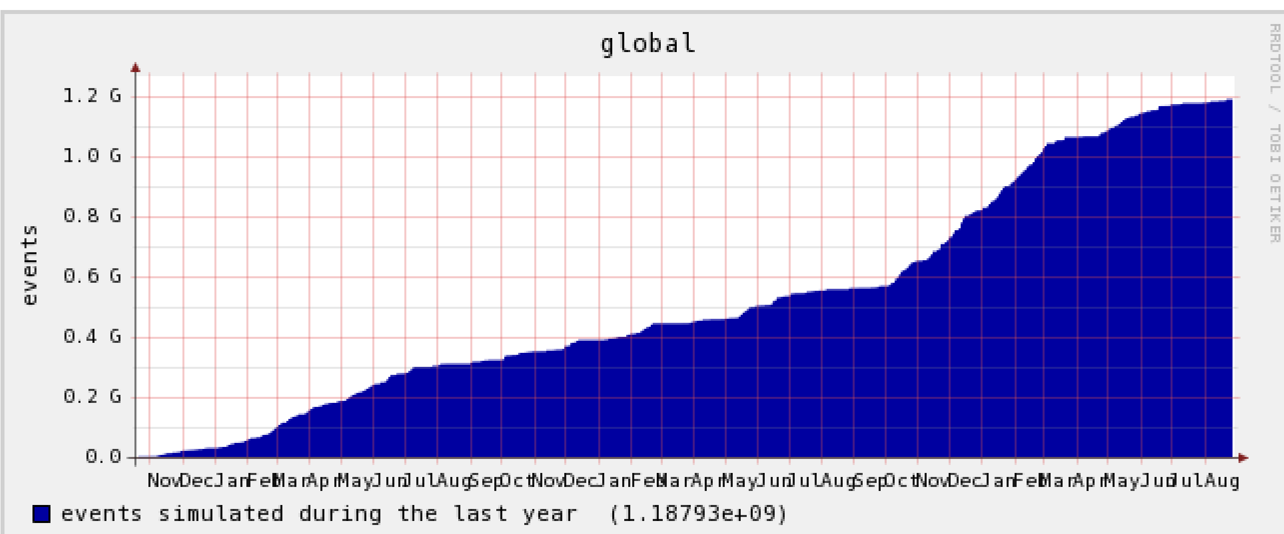


Fig.2
Number of events simulated during last 3 years

The Grid resources provide currently more than 75 percents of computing power for the whole ZEUS MC production. In fig. 2 you can see the number of events simulated over last 3 years. The total MC production on the Grid amounts to about 1.2 billion events.

2.6 Job efficiency

The system has proved to be very robust. Despite of many jobs failures caused by various Grid problems it can automatically detect errors and resubmit jobs.

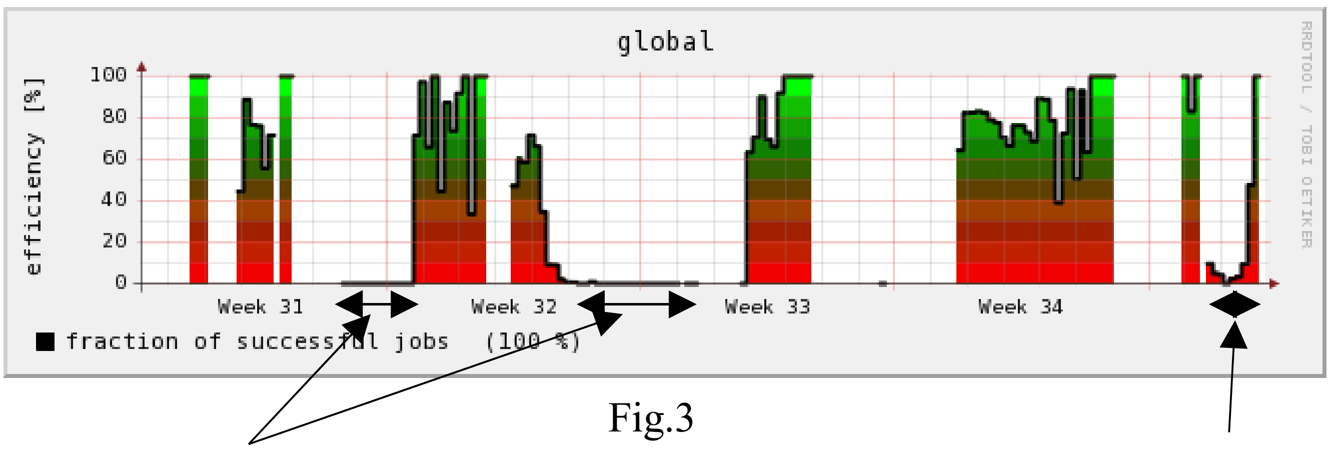


Fig.3
Statistic of completed jobs
1 – Periods with low efficiencies

On fig.3 the efficiency plot for all completed MC jobs over last 4 weeks is shown. Efficiency is defined as the number of successful jobs divided by the number of all submitted jobs. Since all failed jobs are automatically resubmitted the high efficiency is relevant for the production rates and the latency of completing user requests. At the end all users' requests are fulfilled. On the plot one can point out several periods of time when the efficiency significantly drops down, sometimes even to zero. Very often a low efficiency is caused by one faulty site or a single service not working properly. In order to improve the efficiency a more advanced monitoring system is needed. Early problem detection or even automatic repair in some cases can increase efficiency of submitted jobs.

2.7 Grid infrastructure

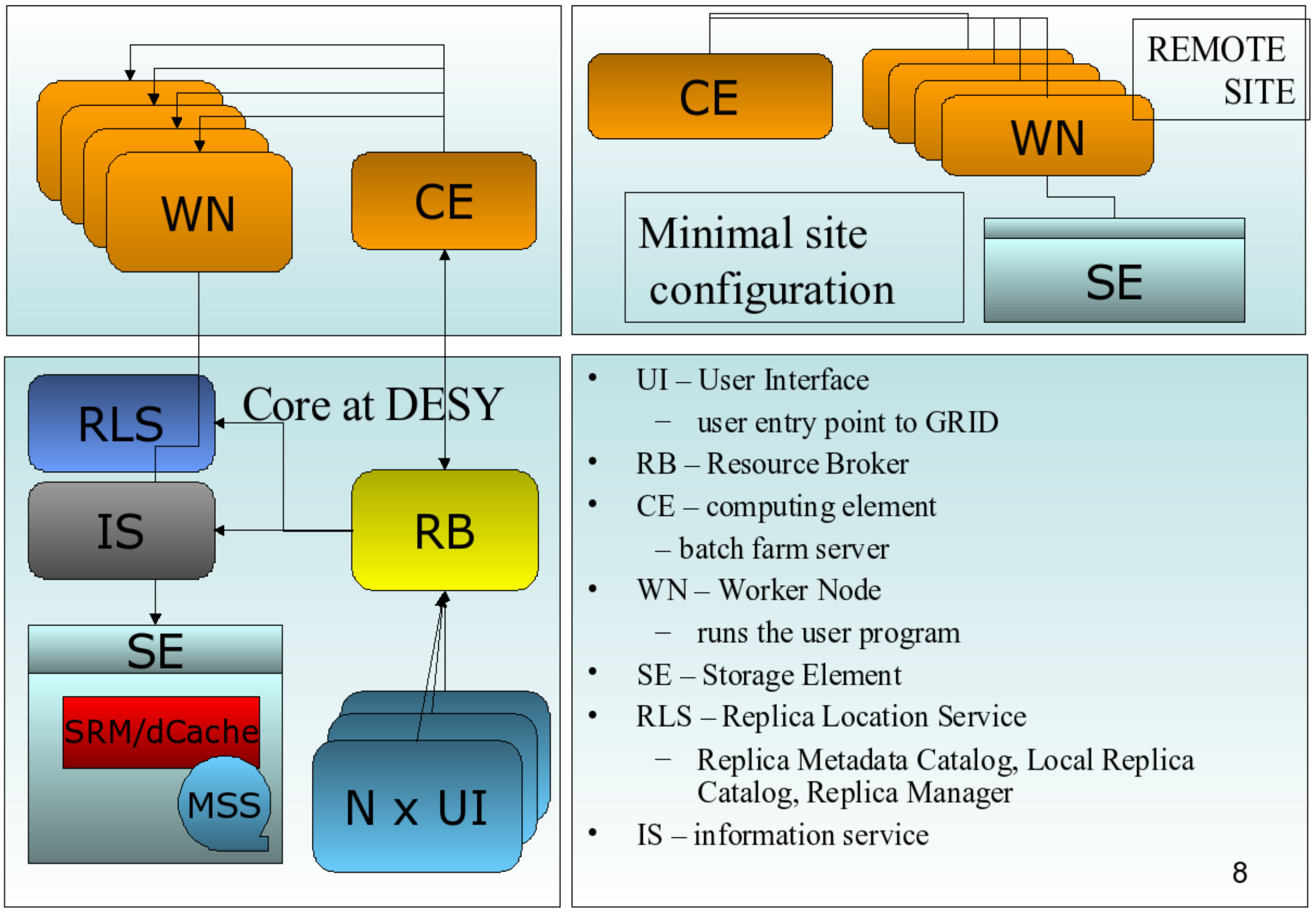


Fig.4
Grid components

The basic Grid components and services as seen from ZEUS point of view are shown on the fig.4.

On the left side one can see the core of Grid infrastructure at DESY and the services which are critical for ZEUS. The most important components are Resource Broker, main Storage Element connected to DESY Mass Storage System, Information Service and File Catalog (RLS).

Users' jobs are submitted to Resource Broker, which then decides where each job should run (job means simulation program). It can run in DESY or on one of the remote sites.

On the right side of fig.4 one can see minimal configuration of the remote site suitable for running ZEUS jobs. It consists of Computing Element, which is batch farm server, number of Worker Nodes, where the jobs are executed and a local Storage Element used for files access.

The availability of these services is very important for running ZEUS MC jobs. Failures of one of core services may cause delays in MC production. Broken remote sites may lower the job efficiencies. A monitoring system, which periodically checks the availability of these services, is needed. One of the choices is to make use of standard Service Availability Monitor for Grid.

2.8 Service Availability Monitor

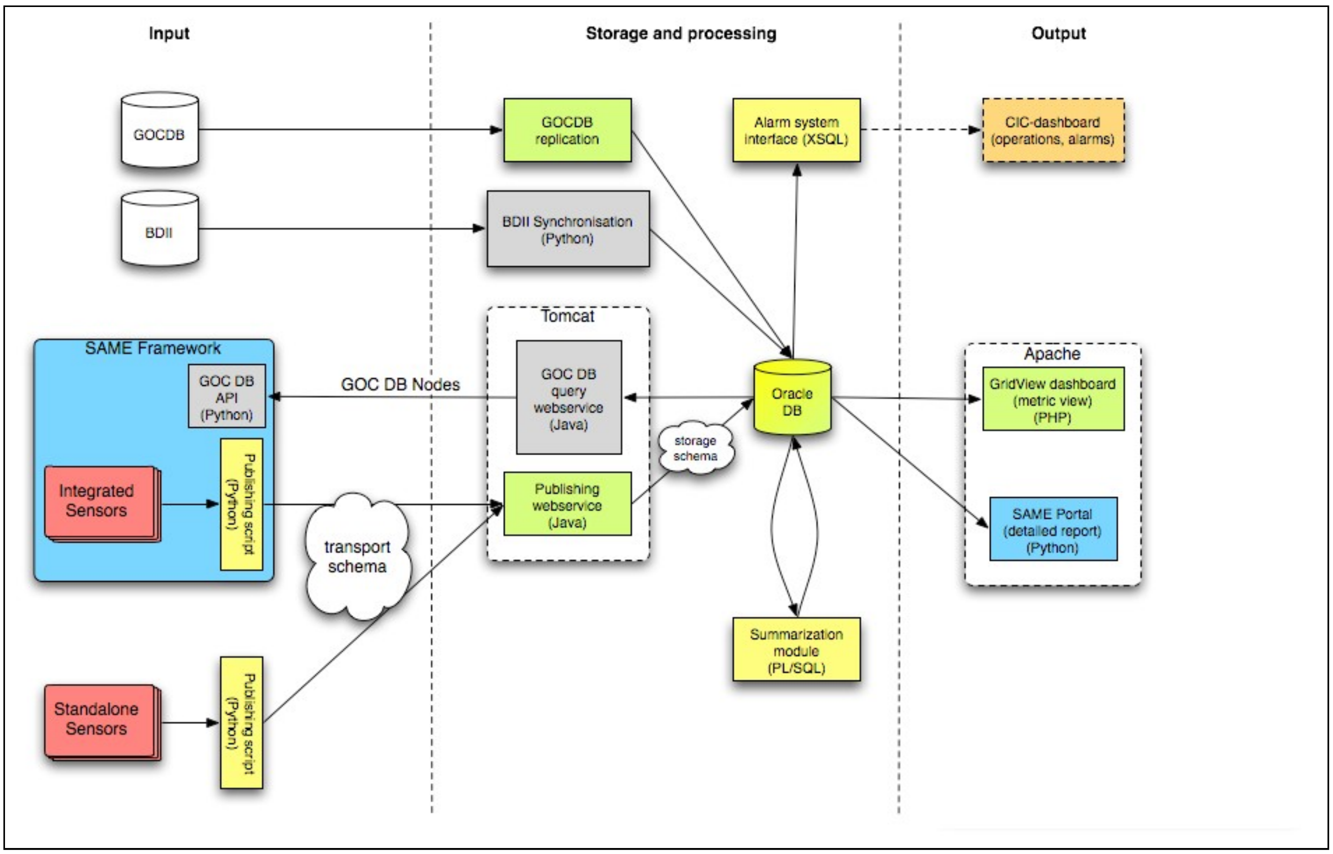


Fig.5
General architecture of the SAM framework

The Service Availability Monitoring framework (SAM) aims to provide a site independent, centralized and uniform monitoring tool for all Grid services. It is the main source of monitoring information for high-level Grid operations and is being used in the validation of sites and services with calculation of availability metrics.

SAM has conceived as an SFT (Site Functionality Test) extension. It has been written mainly in Python (submission framework, logging, error reporting, configuration files), re-using most of the SFT code, with the only exception of the web service API framework.

The heart of the testing and publishing system is an Oracle database (fig.5), which is connected to the SAM sensors through the Tomcat web services. These services include (GOC DB) query and publishing web services implemented in Java or using servlets. In addition to that, the Oracle DB interacts with the top level BDII using a Python script.

SAM testing agents are called “sensor”; a sensor is a collection of scripts which are conceived to check the functionalities of a certain service.

The monitoring procedures are carried out using sensors which regularly publish the results for all monitored service nodes and that are integrated within the SAM framework.

The services currently monitored by SAM are: SE, CE, gCE (gliteCE), LFC, SRM, FTS. Some others are still under development, namely: gRB (gliteRB), MyProxy, VOMS, RGMA.

The SAM system can be extended by additional sensors (tests) which test users or task specific functionality.

3 Description of my work

3.1 Deploying SAM for ZEUS

After I had learnt what is GRID and SAM, I have deployed SAM for ZEUS. I started from setting the standard sensors provided by SAM. The following tests were installed and are run periodically[1].

- CE-sft-wn
- CE-sft-softver
- CE-sft-caver
- CE-sft-brokerinfo
- CE-sft-csh
- CE-sft-lcg-rm
- CE-sft-vo-tag
- CE-sft-vo-swdir
- CE-sft-rgma
- CE-sft-posix
- CE-wn-sec-crl

They test various services on all remote sites supporting ZEUS VO.

In addition to that I have developed and registered new sensor specific to ZEUS MC Production System. It is called CE-sft-zeusmc-files.

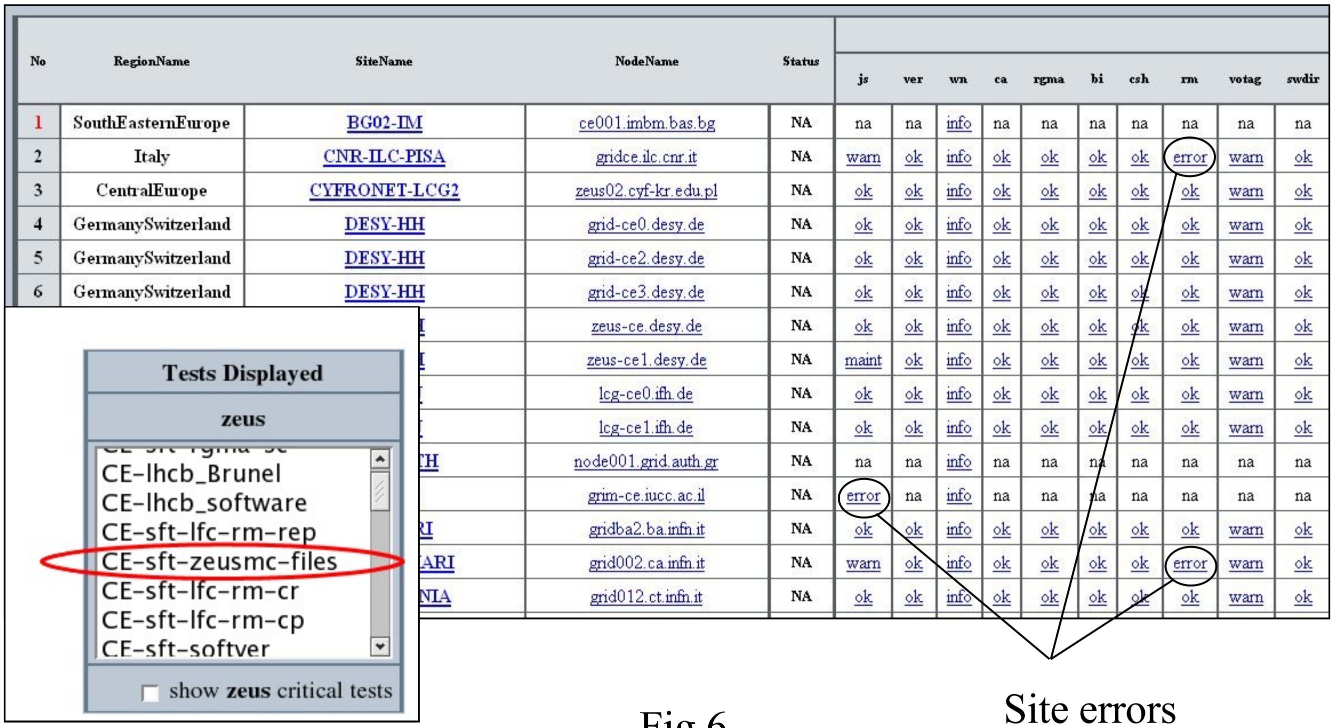
3.2 CE-sft-zeusmc-files – ZEUS specific sensor

Using zeus-grid-toolkit I have written CE-sft-zeusmc-files sensor which checks access to ZEUS files needed for successful completion of MC jobs.

The sensor performs operations on files in a similar way like a real MC job.

I have registered this sensor in a centrally maintained database at CERN. The result of my test can be inspected in the standard way on the common web page (fig.6)

[1] Description of this tests you can find in https://twiki.cern.ch/twiki/bin/view/LCG/SAMTests



No	RegionName	SiteName	NodeName	Status	js	ver	wn	ca	rgma	bi	csh	rm	votag	swdir
1	SouthEasternEurope	BG02-IM	ce001.imbm.bas.bg	NA	na	na	info	na	na	na	na	na	na	na
2	Italy	CNR-ILC-PISA	gridce.ilc.cnr.it	NA	warn	ok	info	ok	ok	ok	ok	error	warn	ok
3	CentralEurope	CYFRONET-LCG2	zeus02.cyf-kr.edu.pl	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
4	GermanySwitzerland	DESY-HH	grid-ce0.desy.de	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
5	GermanySwitzerland	DESY-HH	grid-ce2.desy.de	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
6	GermanySwitzerland	DESY-HH	grid-ce3.desy.de	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
		I	zeus-ce.desy.de	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
		I	zeus-ce1.desy.de	NA	maint	ok	info	ok	ok	ok	ok	ok	warn	ok
			lcg-ce0.ifh.de	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
			lcg-ce1.ifh.de	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
		TH	node001.grid.auth.gr	NA	na	na	info	na	na	na	na	na	na	na
			grim-ce.iucc.ac.il	NA	error	na	info	na	na	na	na	na	na	na
		RI	gridba2.ba.infn.it	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok
		ARI	grid002.ca.infn.it	NA	warn	ok	info	ok	ok	ok	ok	error	warn	ok
		NIA	grid012.ct.infn.it	NA	ok	ok	info	ok	ok	ok	ok	ok	warn	ok

Fig.6
Test results in web

In fig.6 one can see that some tests have failed on several sites. The problems can be easily seen and it is possible to find the real reason for the failure by inspecting log files. Because of the standard way of publishing test results, sites administrators and MC production operator in ZEUS use the same interface for all tests and they can react much faster.

In fig.7 you can see result page for CE-sft-zeusmc-files sensor. Detailed log information is given for all tested operations.

SAM test: *CE-sft-zeusmc-files*
Submitter VO: *zeus*
Node: *zeus-ce.desy.de*
Execution time: *11-Sep-2007 14:36:38*

Access ZEUS MC files , CE/CE-sft-zeusmc-files. Date : Tue Sep 11 16:33:15 CEST 2007

```
arguments :
HOSTNAME : zeus033.desy.de
SAME_VO=zeus
```

COPY FILES

file name	status
funnel.tgz	[OK]
gaf.tgz	[OK]
czar_tlt2006_12_zrelease2007a.1	[OK]
czar_tlt2006_22_zrelease2007a.1	[OK]
czar_v2004_2_newvcrecon	[OK]
czar_v2005b_2	[OK]
czar_v2005c	[OK]
mozart_g321_v2006a.1	[OK]
mozart_g321_v2007a	[OK]
zephyr_v2005a.1_kffit_fix	[OK]
zephyr_v2006a3	[OK]
zephyr_v2007c.1	[OK]

log file

funnel.tgz	debug: 0 cp from funnel.tgz to funnel.tgz copying file from grid: /grid/zeus/funnel.tgz to ./funnel.tgz
gaf.tgz	debug: 0 cp from gaf.tgz to gaf.tgz

Fig.7
Result of CE-sft-zeusmc-files sensor

4 Conclusion

I have studied Grid and SAM monitoring system. I have deployed SAM for ZEUS and have written additional test which focused on ZEUS specific functionality. Now we can use Grid facility more efficiently.

I will finish simple MC test job with real physics to check the right computing ability of sites. The last step I will finish the automatic procedure for disabling broken sites.

During my stay at DESY I have gain experience with working in international team of scientists and learnt about many aspects of the scientific work.

I have improved my technical skills in the computing area. In particular shell scripting, perl and tools used in HEP (ROOT).

I have also learnt about basic ideas of the object oriented software design and its implementation in the real application.

5 Acknowledgement

I want to thank my supervisor Krzysztof Wrona and to Tim-Andrew Namsoo for all the help and advise I got during those eight weeks at DESY and patience when something didn't work. Thanks to Ingrid Gregor for doing hard organization work and being such a helpful person. Thanks also to Igor Katkov (officemate) for answering many questions and for good company.

Finally thanks to all DESY organizers (Andrea Schrader, Joachim Meyer and others), lecturers and summer students for the wonderful and profit time we had here in Hamburg.