

Grid Applications for High Energy Physics Experiments

T. Adye², D. Andreotti³, R. Barlow¹, B. Bense⁴, C.Bozzi³, C.A.J. Brew², R. D. Cowles⁴, E. Feltres⁶, A. Forti¹, G. Grosdidier⁷, M.P. Kelly¹, A. Khan^{5,4}, H. Lacker⁶, E. Luppi³, R.K. Mommsen⁸, A. Petzold⁶, D. Smith⁴, J.E. Sundermann⁶, P. Veronesi³, J.C. Werner¹ and F. Wilson²

Abstract— This paper discusses the use of e-Science Grid in providing computational resources for modern international High Energy Physics (HEP) experiments. We investigate the suitability of the current generation of Grid software to provide the necessary resources to perform large-scale simulation of the experiment and analysis of data in the context of multinational collaboration.

Index Terms—Grid Computing, Distributed computing, Monte Carlo Simulation

Manuscript received June 3, 2005.

- (1) High Energy Physics, University of Manchester, Schuster Laboratory, Brunswick Street, Manchester M13 9PL, UK
- (2) CCLRC Chilton, Didcot, OX11 0QX, UK
- (3) Universita' di Ferrara, Dipartimento di Fisica and INFN I-44100 Ferrara, Italy
- (4) Stanford Linear Accelerator Center, Menlo Park, CA 94025, USA
- (5) Brunel University, School of Engineering and Design, Uxbridge, UB8 3PH, UK
- (6) Technische Universität Dresden, D-01062 Dresden, Germany
- (7) LAL, BP 34 F-91898 Orsay Cedex, France.
- (8) University of California at Irvine, Irvine, CA 92697, USA

I. INTRODUCTION

The BaBar High Energy Physics (HEP) detector [1] is based at the Stanford Linear Accelerator Center (SLAC), Stanford, US. The experiment investigates the subtle differences between matter and anti-matter by continuously colliding bunches of high-energy electrons and positrons 250 million times per second and searching for the creation of rare B-meson and anti-B-meson particles. High speed electronics and online processing rejects unwanted events resulting in a final raw event rate to tape of approximately 100Hz with each event requiring ~30kB. The raw data is sent to Padova (Italy), where the events are reconstructed. The reconstructed events are then separated (“skimmed”) into approximately 180 data streams based on physics properties using compute farms at Padova, Karlsruhe (Germany) and SLAC (US). These data streams are made available as datasets for analysis and used by 600 researchers based at 74 institutes in 10 countries. Since starting in May 1999, the experiment has recorded 3.5 billion events.

In addition to analysing the data, a major task is the simulation of the experiment. At least 3 times as many simulated events are needed as data events. Each simulated event takes ~10 seconds on a modern processor and results in 20kB of storage. Jobs of 2000 events are allocated to a distributed system based on 25 computer farms located in 5 countries with a total of ~1000 processors.

On each farm a complete BaBar software release needs to be installed locally or made available to the batch nodes via AFS [2]. A number of servers must be maintained locally: the location of the experimental and simulated data is provided by a MySQL or Oracle database; the experimental and simulated events are stored and accessed using the ROOT I/O protocol [3] either directly through NFS or, for heavily used farms, with a load-balanced, fault-tolerant file server called Xrootd [4]; an Objectivity [5] object-orientated database is required to

store detector conditions, alignment and calibration constants which are distributed to the jobs via either an Advanced Multithreaded Server (AMS) [6] or a Network File System (NFS). The system requires a great deal of local customisation with each farm maintained by a local manager. Load balancing between independent farms has to be done manually. Monitoring of jobs during production is difficult and there is no way to automatically ensure that all resources are available before commencing a production run.

Despite these drawbacks, the system has run with greater than 98% efficiency and produced 2.5 billion simulated events in 2004 alone. However, the coming years will see a tripling in the experimental data rate and a less manpower intensive system must be found to provide three times the resources. A possible approach is based on the tools and middleware provided by Grid systems such as Globus [7] or the LHC Computing Grid Project, LCG [8]. The goal is a production system with a single production manager submitting jobs to a worldwide Grid of remote sites with automatic resource allocation, job monitoring, output retrieval and cataloguing. Grid middleware, working over the Internet, provides the necessary infrastructure. Many Worker Nodes (WN) managed by Compute Elements (CE) with jobs directed by Resource Brokers (RB), metadata management by the Replica Location Service (RLS) and high volume of data kept on Storage Elements (SE). We have investigated two possible solutions to this task. In the first, the minimum amount of Grid software was installed on top of legacy BaBar farms with data and databases accessed locally. Only the controlling job description was sent to the Grid using the Globus system. In the second approach, no assumption was made about local resources. The required BaBar software was pre-packaged and installed through a Grid account; data and databases were accessed remotely over the Wide Area Network (WAN) using the AMS and Xrootd protocols. The first approach offered the benefits of only minimal changes to pre-existing resources; the second offered the possibility of much reduced local customisation and support load.

II. BABAR SIMULATION PRODUCTION ON THE GRID

In the UK, the Globus 2.4 gateway was installed on BaBar farms located at the Universities of Royal Holloway, Bristol and the Rutherford Appleton Laboratory (RAL). RAL acted as the controlling site. The Ganglia [9] monitoring program was installed locally to provide graphical status of the batch workers. Each identical farm comprised of 40 dual CPU Linux machines, a Solaris machine used as an Objectivity server, two interactive Linux machines, one of which also acted as the OpenPBS [10] server and some NFS mounted storage. The standard production scripts were modified to submit remotely to the gateway; the jobs were passed onto the batch system and then retrieved at the end. Initially the Globus file staging functions were used to transport these files to and from the remote site, but this proved unreliable and it was replaced with GridFTP [11].

The production was automated by creating two daemons at the central site. The gateway was unable to deal with a large number of jobs so a control script was created to monitor the number of jobs running and queued at each remote site and submit more if the number fell below a certain threshold. It also interfaced to the standard BaBar tools to rebuild failed jobs and choose the next jobs for submission. The second daemon took the finished jobs, recovered the data, validated the output and updated the catalogue.

During 2004, this system was deployed at four sites across the UK. 31 million events were produced in nine months on this system with peak production rates in excess of 2.5 million events per week and more than 100 concurrent jobs at a given remote site. The failure rate in stable running was below five percent.

This system was limited to running on existing BaBar resources as the full BaBar software and servers were required. The sites were still separate farms and load balancing was not possible. No reduction in manpower was achieved as a complete local installation of the BaBar environment was required with constant updates to the local databases. A remote control programme was needed to limit the number of jobs submitted as the standard Globus gatekeeper spawned processes for every running and queued job which overwhelmed the gatekeeper and resulted in an unresponsive system. Job submission failures were also traced to poor interaction between certain firewalls and the TCP timeout behaviour.

The second approach used the full LCG infrastructure and prototyping was performed on the INFNGrid [12] farms directly involved in simulation production for BaBar (Fig. 1). LCG offers a number of possible advantages as the middleware was developed with HEP application use cases in mind. It provides an integrated data management system and a Resource Broker (RB) that can direct jobs to compatible resources without involvement of the submitter.

The LCG model also adds a number of challenges. The BaBar software is not available at the remote site. Therefore the complete BaBar software was reduced to a core subset that was packaged as a tar archive of about 37 MB (130 MB uncompressed). The package was then distributed on each Grid Worker Node by using procedures based on the LCG middleware to install and uninstall new software and to publish new tags. A site manager, mapped as a special user, was able to use these procedures by simply submitting specific jobs on the Grid choosing which sites must be involved.

Three sites (Ferrara, Padova, Naples) were configured to provide Objectivity and Xrootd services that could be accessed from the Grid Worker Nodes over the WAN using AMS and the Xrootd protocol. The typical amount of conditions data read by a single simulation job was about 2% of the total and the use of the WAN reduced the CPU efficiency by about 5%, which was acceptable.



Fig. 1: Resources available for BaBar SP on the Italian Grid.

Standard BaBar software tools for simulation production were installed on the LCG User Interface (UI) and properly configured. These tools were needed to merge several collections of homogeneous events into a single collection, transfer events to SLAC, and to update the bookkeeping database at SLAC.

A JDL [13] script sets up all data needed and specifies the parameters for the simulation including type of events, run number, background triggers to use, number of events and detector conditions. Jobs were then submitted through an UI to a Resource Broker specifically installed for BaBar, located in Ferrara. The RB was able to manage Italian and European resources directly involved in BaBar. The RB performed matchmaking between resources available on each site of the Grid, published by Computer Elements (CE), and jobs requirements (memory, specific software release and type of the batch queue). Jobs were sent to CEs satisfying the requirements, and distributed to WNs, where the execution started. The simulated events were saved as ROOT I/O files and transferred from WNs to a Storage Element (SE) located in Ferrara. A conceptual schema of the flow described above is shown in Fig. 2.

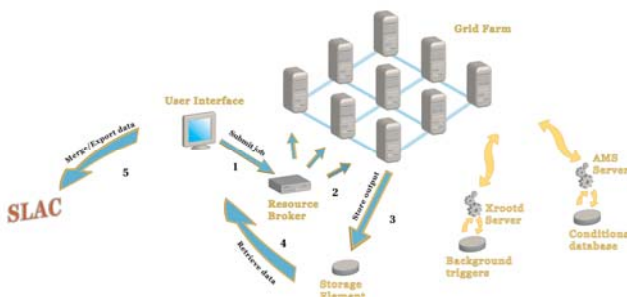


Fig. 2: BaBar simulation production model implemented in Italy

During initial tests a single condition database installed in Ferrara was used as AMS server while jobs were submitted on Ferrara, Napoli, Trieste, Catania, Bari and Padova. A modified version of AMS, written at SLAC [14], was designed to handle more connections. It was installed in

Ferrara and its performance was compared to the standard AMS. Both AMS servers used 32 threads to handle client requests. Stress tests, performed by submitting from 100 to 250 jobs of 2000 events each, showed that standard AMS supported about 30 concurrent simulation jobs before failing to respond. The number of failed submissions increased with increasing load, reaching almost 50% when 80 clients contacted the server simultaneously. Performances improved using the SLAC AMS server. The server was able to handle about 75 clients before jobs failure occurred.

The maximum number of AMS connections a client can generate could be configured in the job script. Limiting the number of connections increased the execution time for some jobs while allowing more jobs to contact the AMS in parallel. A good balance between efficiency and stability was achieved, setting the maximum number of connections to 20 for each client.

After these initial tests, 2 tests were performed accessing the AMS server at different sites. For every test, 200 jobs were submitted, each with 2000 events in order to simulate a real production flow. For both tests, the time spent to complete jobs on each site was measured. The maximum number of concurrent clients connected during production was taken into account. The link speeds to Bari, Ferrara, Napoli and Padova were 15, 16, 32 and 160 Mb/s respectively.

Data were read using the AMS server installed in Ferrara, while production jobs have been submitted on Worker Nodes located in Ferrara, Padova and Bari. Most of the jobs run on Padova due to the high number of WNs provided. About 99% jobs successful completed (Table 1), a success rate comparable to the non-GRID production.

Site	Jobs completed	Mean elapsed time
Bari	29	10:01:04
Ferrara	29	08:02:04
Padova	141	07:28:44
<i>Total</i>	<i>199</i>	<i>07:55:47</i>

Table 1: Data read from AMS located in Ferrara

In the second test, data were read remotely, using the AMS server installed in Naples from production jobs submitted on Worker Nodes located in Ferrara, Padova and Bari (Table 2).

Site	Jobs completed	Mean elapsed time
Bari	43	09:41:43
Ferrara	12	09:00:46
Padova	130	07:38:32
<i>Total</i>	<i>185</i>	<i>08:12:30</i>

Table 2: Data read from AMS located in Naples

Due to the fast network link, jobs at Padova completed more quickly than jobs submitted on other sites. The quality of network access and data location played a basic role in turn around times and is an important parameter in choosing the optimal running conditions. The whole system was tested producing real allocations data for Monte-Carlo (more than 3,000,000 events), successfully completing the whole production cycle for about 98% of the total jobs.

III. ANALYSIS ON THE GRID

The simulation of data is a much more tractable problem than the analysis of data. Data analysis requires 100% reliability if the results are to have any meaning and needs to access huge amount of data, which has to be located and accessed remotely. The output differs in form and size depending on the type of analysis. To catalogue and make this non-standardized output available needs a flexible system. To test an analysis environment, a prototype farm based on LCG middleware software was created to test simple stand-alone analysis programs that could handle data location and job management. Initial testing with specialized Grid-analysis infrastructures using AliEn [15] showed interesting possibilities, but also the need for greater extensibility and stability. We therefore created a specific BaBar job submission script called EasyGrid.

IV. EASYGRID: JOB SUBMISSION SYSTEM FOR ANALYSIS

The EasyGrid Job Submission software [16] is an intermediate layer between the Grid, where the resources can be found, and the user, who has a pre-developed software analysis (written in C++), which they run on a chosen dataset. EasyGrid's first task is to find what event files are in the dataset. For each dataset there is a metadata file containing the names of the event files. These physical files are registered with the RLS, with several logical file names assigned to them as aliases, identifying which CEs contain copies of that dataset. If a CE holds any of the files in a dataset it holds all of them. Searching all the aliases for a dataset name provides a list of CEs to which jobs can be submitted.

The next stage is generation of all necessary information to submit the jobs on the Grid. This is done by the GGenerator of Resources Available (Gera) which produces the Job Description Language (JDL) files, the script with all necessary tasks to run the analysis remotely at a WN, and some Grid dependent analysis parameters. The JDL files define the input sandbox with all necessary files to be transferred, and a WN load-balance algorithm allocates the task to machines that provide the most effective resources. When the task is delivered in the WN, scripts start running to initialize the specific BaBar environment, and the analysis software is downloaded. The analysis executable is allocated in the SE and its logical file name (LFN) is also catalogued in the RLS so any WN need download it only once.

Users can follow up the process by querying job status. If the job is done, a task recovering results in the user's directory is performed automatically. If the job was aborted in the process, the diagnostic listing is stored in the history file for further analysis.

V. CONCLUSIONS

The simulation of HEP events is now feasible using Grid architecture. Reliability is beginning to approach that of local self-contained compute farms. The Grid can be interfaced to legacy software and procedures. As most non-trivial HEP programs require access to centralized or distributed resources, our prototyping shows that the major architectural problems concern the installation of these servers and their efficient access over the WAN. EasyGrid opens the possibility to use the Grid for analysis using the data accumulated by BaBar over five years. Future work will concentrate on simplifying the identification of data, providing a common environment on all Grid resources and providing servers to provide regional access over the WAN to centralized databases.

References

- [1] B. Aubert et al., "The BaBar detector," Nucl. Instruments. & Methods, Vol. A 479 (2002), pp. 1-116
- [2] J. H. Howard, Carnegie Mellon University, "An Overview of the Andrew File System," USENIX Conference Proceedings, Winter 1988
- [3] Rene Brun and Fons Rademakers, "ROOT - An Object Oriented Data Analysis Framework," Nucl. Instruments & Methods, Vol. A 389 (1997), pp. 81-86. See also <http://root.cern.ch/>.
- [4] A. Hanushevsky, "The Next Generation Root File Server," in CHEP proceedings, Interlaken, Switzerland, September 2004.
- [5] Objectivity Inc. [online]. Available: <http://www.objectivity.com>
- [6] AMS: J. Becla, "Setting up AMS", 2002, [online]. Available: <http://www.slac.stanford.edu/BFROOT/www/Public/Computing/Databases/experts/setupAMS.shtml>
- [7] The Globus alliance, [online]. Available: <http://www.globus.org/>
- [8] CERN, "LHC Computing Grid Project", [online]. Available: <http://lcg.web.cern.ch/LCG/>
- [9] Ganglia, [online]. Available: <http://ganglia.sourceforge.net/>
- [10] OpenPBS, [online]. Available: <http://www-unix.mcs.anl.gov/openpbs/>
- [11] GridFTP, [online]. Available: <http://www-fp.globus.org/datagrid/gridftp.html>
- [12] INFNGrid: GridIt, INFN Production Grid, [online]. Available: <http://www.grid.it/>
- [13] Job Description Language: DataGrid-01-TEN-0102-0_2, 2001 Datamat
- [14] SLAC AMS: A. Hanushevsky, "Running Multiple AMS Servers per Host", 2002, [online]. Available: http://www.slac.stanford.edu/~abh/Objectivity/Misc/mult_ams.html
- [15] AliEn, [online]. Available: <http://alien.cern.ch>
- [16] J.C. Werner, "HEP analysis, Grid and Job Submission", [online] Available: <http://www.hep.man.ac.uk/u/jamwer/>