



Grid Benchmarking – A review of current efforts and approaches

Executive Summary

The grid marketplace is inflated with a variety of open source and commercial offerings at different levels of maturity. Each promises essentially the same functionality and It is difficult to choose between them. The traditional approach is to run a proof of concept which compares various options. The question is what to compare, and how to compare it. These questions are particularly difficult to answer in the context of a grid environment.

This whitepaper reviews current research and results in grid benchmarking, and draws out the following key points:

- **There is no set of standard grid benchmarks** - grids can only be compared and characterised in specific application contexts.
- **Traditional HPC benchmarking is not appropriate for Grid** - although grid benchmarking has clearly benefited from knowledge built up in this area.
- **Benchmarks should, where possible, be based upon full, realistic applications** - So called Micro kernel benchmarks are useful, but it is only possible to know how a grid system will run under a production workload by running it under a production workload.
- **Grid benchmarking and comparison is possible through the adoption of a suitable framework** - in particular this paper recommends a framework proposed by Huedo (2005) which defines appropriate metrics for useful comparison.

Introduction

Grid may be defined as a collection of resources glued together by a set of middleware that provides connections and services for shared use. The fundamentally heterogeneous possibilities offered by grid address many issues connected to enterprise computing and HPC, whilst at the same time creating new challenges. One of these challenges is in Quality of Service (QoS)- the question is whether it is possible to guarantee QoS at all in an heterogeneous, pervasive, dynamic grid environment; one in which virtual organisations (Foster 2001) can join and leave, or provide various resources at their own whim and as their own business dictates.

Attempts to answer this question lead down two paths. The first relates to the ability to provision and dynamically allocate grid resources based upon profile, policy and priority. This is a fundamental and challenging aspect of grid functionality and in one standard Service Oriented Architecture (SOA) it is handled through implementations at the *Collective* layer (Foster, Kesselman 2004). The second, which underlies the ability to allocate appropriately and is fundamental in an understanding of the very nature of a grid, is an appreciation of how the grid functions and performs in a variety of scenarios and under various loads. This characterisation of a grid is through benchmarking.

The nature of grid makes benchmarking difficult, and although drawing from established HPC work grid benchmarking is emerging as an important independent field. This whitepaper reviews and summarises the current status of grid benchmarking, drawing on published research to provide a platform for best practice for grid comparison, evaluation and troubleshooting.

The nature of benchmarking

Benchmarking is commonly employed for the measurement of processors, single machines and of HPC systems, and there are a number of well-known and well understood benchmarks that are often used to understand the various performance levels. In the HPC frame such benchmarks fall into two main camps (Snively et al 2002):(Snively, Chun et al. 2003)

- **Low level probes** - these are used for determining the rate at which machines, singularly or collectively, can perform computations. Probes often take the form of Micro-benchmarks which focus on an isolated aspect of performance. Examples would be STREAM (McCalpin 2006) for the measurement of the costs of memory allocation and HPL (top500.org 2006) for the measurement of floating point operations per second. Linpack benchmark results have traditionally been the measurement used to rank the top 500 supercomputers. Results are measured in floating point operations per second (FLOPS), with IBM BlueGene/L being at the top currently with a score of 280 TFLOPS. Recently there has been a movement to reduce the reliance on HPL as the only measurement of performance, and the direction is now towards a suite of 7 benchmarks (which include STREAM and HPL) known as the HPCC challenge Benchmarks, that offer a better all round indicator of supercomputer performance For now however, since the HPCC benchmarks cannot offer a single consolidated score, it is difficult to see how this will replace HPL in generating a “top 500” list.
- **Application Specific benchmarks** - These may be synthetic, but are designed to be representative applications that attempt to match the calculation characteristics and workloads of classes of grid applications. There are two ends of the application specific benchmarking spectrum; at one end real applications are used as benchmarks. The idea of realistic application benchmarking is central to the SPEC philosophy, and the SPEC HPG application benchmarks for parallel computation (Eigenmann, Gaertner et al. 2001) consist of large scale application simulations that exercise inter-processor communications. The NAS parallel benchmarks are at the other end of the spectrum, these benchmarks (Bailey, Barszcz et al. 1991), which may be characterised as micro-kernels, model the workflow and characteristic of four generic application types.

Understanding how these benchmarking approaches translate to GRID represents a significant challenge to the GRID community (Dikaiakos 2005). Efforts so far have centered around evolutions of the NAS parallel benchmark suite. At this time the NGB (Nas Grid Benchmarks) form the basis of the GGF grid benchmarking initiative (Van der Wijngaart, Rob 2006, F, Van der Wijngaart, Rob., F., Frumkin 2004), but as yet there has been no effort to construct or port full application benchmarks to a Grid infrastructure.

Benchmarking is a difficult task, not so much because of the technical issues around building and running benchmark codes (although this should not be underestimated) but more because it is very its very nature a social, consensual activity. Benchmarks are

used for different purposes. They may be used for system and/or component comparison and in system design (Eigenmann 2001). Manufacturers go to great lengths to ensure that their system or component is designed to perform well under a given benchmark. Of course, this benchmark may well have no bearing on the actual use to which a given system or component may be put. Another use of benchmarking is in evaluation. In this case it will be the customers, rather than the manufacturers who choose and perform the benchmarks. Of course no benchmark will give accurate predictions of behaviour of a system or component under production stress, other than perhaps a full production run of a full application that the system is intended for. But that does not mean that benchmarks are inappropriate. On the contrary they give valuable information. However there are some guidelines which should be observed when building benchmarks that increase the value of a given benchmark and that are as relevant to Grid benchmarking as any other. The following, non exhaustive list, are taken from Dikaiakos (2005).

- ❑ **Benchmarks should be fair** - they should not favour a particular system or implementation. To this end SPEC have gone to great lengths to define run rules to the level of allowed compiler flags. This so called *baseline debate* centres around the question of whether excessive optimisations should be permitted (Darema 2001).
- ❑ **Benchmarks should be easy to use** - The more a benchmark is run and reported the more valuable it is, making the benchmark available, portable and easy to use increases its value to the community and its chances of wider adoption.
- ❑ **The specification should be openly obtainable** - This is particularly important if a benchmark provides a single number (Such as LINPACK and SPEC HPG benchmarks) as an output. It is crucial to know what it is that the benchmark is measuring, and what the nature of the sample workload is (for example: is it FPU or ALU intensive). This helps ensure relevance of the benchmark to the system under test.
- ❑ **Management should be affordable** - The management cost of all aspects of the benchmark should not outweigh the value of the benchmark itself. Performance metrics must be simple to collect, store, analyse and understand and execution, installation and compilation of the benchmark should be as simple as possible.
- ❑ **Metrics should be closely aligned to the benchmark** - Metrics should be associated with the structure of the underlying benchmark and the characteristics of the system under test in order that useful conclusions may be drawn from the tests.
- ❑ **Benchmarks should be realistic** - As far as possible, they should be a representation of the realistic conditions under which the system under study would be used.

However benchmarks are designed and run, no benchmark will satisfy all users. Despite efforts to create parity through run rules and baselining, the fairness of benchmarks will always be questioned, as will the general applicability of benchmarks in predicting the behaviour of systems under production workloads.

“the TAP philosophy is that realistic applications are the most appropriate way of benchmarking complex High Performance Computer systems”

Eigenmann (2001) has been at the forefront of a push towards realistic benchmarks, and has created the TAP (Top Application Performers) list for benchmarking computers based upon realistic applications. Although not in direct competition with the top 500 list, it does represent a very different philosophy. With only 3 systems benchmarked so far, the TAP project has a way to go against stiff opposition and the emergent HPC benchmarking effort. TAP is aligned closely with the SPEC HPG (it uses their most recent benchmark) which has taken as a basic underlying philosophy that full, realistic applications are the most appropriate way of benchmarking complex High Performance Computer systems.

“Evaluation of High Performance Computer Systems must be based on realistic, full application. Kernel. Algorithm and small application benchmarks are important for measuring and discussing performance of a computer system. However the complex performance behaviour of these components in concert and in the context of a large computational problem can only be realistically understood if we observe the problem as a whole”.(Weicker 2001) p.40.

The criteria for a SPEC HPG benchmark are that it must be:

- Widely used to solve realistic problems
- SPEC would be allowed to distribute the application (this is difficult to achieve)
- It is available in both serial and parallel variants
- It has a sponsor who is able to support the maintenance and development of the code.

The resulting set of applications is drawn from seismic, chemical and climate simulations. They are available in MPI (message passing) and Open MP (Parallel directive, shared memory) versions. The single metric published on the SPEC web-page is essentially a measure of throughput - the number of times that an application could be run in a day, and is based upon measurement of the elapsed (wall clock) time which reflects the overall performance of the code.

Grid technology has benefited from lessons learnt in HPC (Kennedy 2004), and this is particularly the case in benchmarking. Most benchmarking efforts take as a starting point the HPC benchmarking initiatives. It is important then not to lose sight of the difficulties encountered in previous benchmarking efforts.

Loosely coupled distributed systems are valuable ways of harnessing compute power to solve large computational problems. However, performance comes at a price as there is a risk of component or service failure. Reliability, availability, maintainability and fault tolerance are essential for the maintenance of a given service level, and so are important characteristics of such systems. It is important then to establish ways of measuring the ability of a system to deliver specified services within a given SLA - the *dependability* of a system. Dependability benchmarking is well established in the field of

databases, networks and operating systems (Koopman, Madeira 2002), and is emerging as important in the grid space.

Benchmarking the Grid - challenges and efforts

The Grid ‘problem’ has been defined as “flexible, secure, co-ordinated resource sharing among dynamic collections on individuals, institutions and resources” (Foster 2001) p.200. These dynamic collections of resources are clustered into Virtual Organisations. Sharing resources in this way brings with it a unique set of challenges. Crossing organisational boundaries requires technological as well as social successes and the heterogeneous nature of the underlying hardware and infrastructure will affect dependability as well as performance. The Grid architecture is a fully featured middleware with a set of services that provide for resource allocation and discovery, broking, policy management, scheduling and prioritisation and security services. There are a number of steps in going from job submission to completion and equivalently many opportunities for waiting, latency and failure. It is hoped that there will be a Global grid which virtual organisation can join and leave offering compute services at a price and/ or sharing resources. Characterising the nature of the resources and understanding the available qualities of service will become exceedingly important. Benchmarking, and benchmark results, then may become a key aspect of the Grid for the future. At this time, however, it must be recognised that there are many grids running in single organisations that perform similar functions to HPC systems, which also require characterisation. The emergence of the ‘Blade’ server, coupled with low latency grid Systems has allowed the construction of Supercomputers that sit very near the top of the top 500 list.

“Traditional HPC benchmarks cannot be directly transferred to the Grid setting”

Traditional HPC benchmarks cannot be directly transferred to the Grid setting because there is an overhead in using Grid services (Tsouloupas, Dikaiakos 2003). This overhead must be quantified in order to understand their impact on application performance (Snively, Chun et al. 2003).

Useful metrics for Grid benchmarking are *Turn-around time* - the time between starting a job and obtaining the data; and *Throughput* - the maximum possible submission volume without effecting the turn-around time (Snively, Chun et al. 2003). A throughput benchmark is intrusive and thus undesirable because in this situation at least one of the grid resources has to be stressed to its limit. Turnaround time also poses issues due to reproducibility - it is extremely difficult to guarantee determinism on a heterogeneous environment where resource discovery and allocation works towards efficiency and not reproducibility. The only way to guarantee reproducibility in turn around time is to reserve all of the appropriate resources for the benchmarking test.

The overall performance of a grid can be affected by the performance of its individual elements: The performance of underlying grid hardware in a given VO (Virtual Organisation); the performance of the network or WAN; the overhead of the Grid services supporting job submission themselves; the reliability of the grid software; the performance of libraries and services providing higher level grid layers such as data transfer and synchronisation. Because of these factors, statistical analysis of many runs of the same benchmark test are required (Dikaiakos 2005, Tsouloupas, Dikaiakos 2003).

“Performance is measured by job Turnaround time”

A proposal for a methodology for the evaluation of that capabilities of various Grid environments (Huedo, Montero et al. 2005) suggests that the focus of evaluations and benchmarking be around three key areas

- *Functionality* - tested by running the NGB grid benchmarks
- *Reliability* - tests the ability of the environment to continue execution under the three failure scenarios of job cancellation, system crash or network disconnection
- *Performance* - measured by **Turnaround time** and **Productivity** (defined as the number of completed benchmark instances per unity of time).

Recognising the need to quantify the overheads of the Grid itself, the authors propose that additional metrics are captured:

- **Response time** - time between submitting the job and the starting of the sub job on the execution host. This provides information about the Grid middleware overhead
- **Transfer and execution time** - these metrics are averaged, and are useful to evaluate the impact of individual components on overall performance. Execution time being defined as the average of the execution times on each node; Transfer time the average time between the job leaving the manager and starting at the execution node.
- **Resource Usage** - defined as (execution time/ turnaround time) it represents the achieved level of parallelism.

To illustrate the applicability of this framework NAS grid benchmarks were run on a Globus basic service using the GridWay meta scheduler implemented on the Distributed Resource Management API (DRMAA), a high level standard interface. The measures obtained gave useful information about the performance of various parts of the grid under the NGB synthetic applications, offering possibilities for performance adjustment at each layer of the grid. The hope is that this framework, or one evolved from it will serve as the standard for the evaluation of Grid environments, The use of the NGB is not essential for the success of the methodology, but is achieving popularity because of its ready availability and due to GGF support (Van der Wijngaart, Rob., F., Frumkin 2004). Recently the NGB have been renamed the ALU intensive benchmarks because they exercise mainly the ALU (*ibid*).

The NAS Grid Benchmarks represent a class of grid benchmarking that is designed to simulate realistic workloads. Derived from fluid dynamic applications the NGB are not realistic applications, but attempt to match four identified workload profiles:

- **Embarrassingly Distributed** - loosely coupled applications that function on a “scatter-gather” paradigm, bundling up tasks to be calculated on each node independently, and then returned to a master for final processing. Examples of this are the SETI@Home project, and Monte Carlo calculations.

- Helical chain- represents chains of repeating processes, such as sets of fluid flow calculations run one after another (which is a usual practice with long running simulations)
- Visualisation Pipe - represents the scenario in which chains of compound processes are run such as those encountered when visualising flow solutions.
- Mixed Bag - involves flow computation, post processing and visualisation. Different amounts of data are transferred from each task, and several flow calculations can start simultaneously. This calculation severely stretches the capabilities of grid solutions.

“One of the key differentiators is the inclusion of a strict verification test”

One of the key differentiators of this benchmark is the inclusion of a strict verification test. A benchmark performance figure is meaningless if the result is wrong, and so checks are included to ensure that the results are reliable (Frumkin, Van der Wijngaart 2001). These computationally intensive grid benchmarks aim to be representative of tasks run on a grid and use little initialisation data. The key metric for the benchmark is the *Turnaround* time, which reflects measures of latency. (Dikaiakos 2005) argues that this metric on its own offers few insights about a Grid due to the large number of factors outside the NGB specification that determine the success of an NGB job, including the network traffic, queuing, file system etc. One may equally argue that because of the complexity of the grid, and the number of factors involved, the only meaningful benchmark is the turnaround time and its derivatives. Certainly deterministic processing, although desired is almost unachievable, and Snavelly, Chun et al. (2003) acknowledge that one issue with turnaround time as a metric is repeatability.

A number of studies have focused on the NGB benchmarks and their applicability in characterising a Grid. One of the key aspects of benchmarking is to test functionality. (Herrera, Huedo et al. 2004) use an implementation of the NGB on a Globus test bed in order to demonstrate their middleware independent implementation of a Distributed Resource Management Application API (DRMAA), a standard API for porting distributed applications among different environments, for scientific application development. With an initial reference implementation run on a 46 node heterogeneous cluster (Frumkin, Van der Wijngaart 2001) show that the NGB Turnaround time benchmark can give insight into the behaviour of a Grid. (Peng, See et al. 2004) successfully use NGB to compare Grid middleware. Metrics taken on a grid using the Sun Grid Engine (SGE) and on the same Grid using the Globus toolkit show that the SGE incurs less grid overhead. This is particularly evident for small problem sizes, with as expected, the grid overhead becoming negligible on large problem sizes. Clearly the NGB have value, although there is still work to be done in this space, in particular in the definition and measurement of metrics and synthetic applications. It will be a while before the real value of the NGB are known.

“Low level probe benchmarks are designed to test fundamental operations”

An alternative and complementary approach to benchmarking the whole grid using synthetic applications is in the use of micro-benchmarks. These low-level probe benchmarks are drawn out of the class of HPC benchmarks designed to test fundamental operations (i.e. Linpack, STREAM). They focus on basic grid operations and aim

to quantify compute times, network transfer times and middleware overhead. Chun, Dail et al. (2004) have developed a set of three probes with different data exchange patterns. The probes are designed to run independently of a grid scheduler, and run on a defined set of nodes, exchanging data between them on a defined pattern to capture metrics on grid functionality such as file transfer and remote execution. These probes are tested in a suitable test bed, and repeated runs generate valuable datasets that the authors suggest are valuable for Grid research. The results were particularly useful for diagnostic analysis of the grid configuration.

Gridbench (Tsouloupas, Dikaiakos 2003) is a tool for design, integration and configuration of grid benchmarks. It provides a portal for running benchmarks, and collecting and analysing results. The analysis and interpretation of results is as important as collecting them, and GridBench, which has been built as part of the CrossGrid Project, offers a useful framework. A common definition language (GBDL) is used to configure and specify benchmarking experiments. Experiments may be run multiple times, and simple data analysis tools are provided. So far no new benchmarks or novel metrics have been developed as part of the GridBench initiative. However the value of a tool such as GridBench is in productivity and consistency; this is shown in (Kenny, Coghlan et al. 2005) where GridBench has been successfully used as the basis to run and compute benchmark results in comparison of two OS platforms in an heterogeneous grid.

Conclusion

Benchmarking a Grid is a difficult business, not least because of the complexity of the interaction of the various services and infrastructure that together form the Grid. Lessons can be learnt from benchmarking efforts in HPC which have found accepted ways of measuring and comparing parallel distributed systems. Indeed both the NGB synthetic application benchmarking effort and the micro-kernel effort draw directly from the HPC world.

Grid benchmarking is a new, emerging field, which has practical application in characterising and understanding grid configurations in a production and development environment. At this time there are two distinct and equally valuable approaches to measuring aspects of grid performance. These approaches are complimentary and future work may see the coming together of these initiatives into a suite of valuable benchmarks.

Unlike the HPC world the suite of application benchmarks does not include realistic applications drawn from business and industrial scenarios. These are difficult benchmarks to build and maintain, but they do offer good indications of how related applications would perform on a given grid configuration. SPEC HPG offer standard application benchmarks, but have only three in their portfolio. None of these have been ported to the grid, and none are financial applications. Further work may be appropriate to build and port realistic applications into an appropriate framework for grid characterisation.

It is important to continue to evaluate and standardise evaluation methodologies such as that offered by Huedo (2005). In coming to a shared understanding in the community of what to measure, it may be easier to understand how best to measure it.

References

- BAILEY, D.H., BARSZCZ, E., BARTON, J.T., BROWNING, D.S., CARTER, R.L., DAGUM, L., FATOOHI, R.A., FREDERICKSON, P.O., LASINSKI, T.A., SCHREIBER, R.S., SIMON, H.D., VENKATAKRISHNAN, V. and WEERATUNGA, S.K., 1991. The NAS parallel benchmarks—summary and preliminary results, *Supercomputing '91: Proceedings of the 1991 ACM/IEEE conference on Supercomputing*, 1991, ACM Press pp158-165.
- CHUN, G., DAIL, H., CASANOVA, H. and SNAVELY, A., 2004. Benchmark probes for grid assessment, 2004, pp276.
- DAREMA, F., 2001. Performance evaluation with real applications. , pp. 8-18.
- DIKAIKOS, M.D., 2005. *Grid Benchmarking: Vision, Challenges and Current Status*. TR-2005-11. Department of Computer Science, University of Cyprus.
- EIGENMANN, R., ed, 2001. *Performance Evaluation and Benchmarking with Realistic Applications*. . Cambrige, Mass: MIT Press.
- EIGENMANN, R., GAERTNER, G., SAIED, F. and STRAKA, M., 2001. SPEC HPG benchmarks: performance evaluation with large-scale science and engineering applications. , pp. 40-48.
- FOSTER, I. and KESSELMAN, C., 2004. Concepts And Architecture. In: I. FOSTER and C. KESSELMAN, eds, *The Grid 2: Blueprint for a new Computing Architecture*. second edn. San Fransisco: Morgan Kauffman, pp. 37.
- FOSTER, I.T., 2001. The Anatomy of the Grid: Enabling Scalable Virtual Organizations, *Euro-Par '01: Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing*, 2001, Springer-Verlag pp1-4.
- FRUMKIN, M. and VAN DER WIJNGAART, R.F., 2001. NAS Grid Benchmarks: a tool for Grid space exploration, 2001, pp315-322.
- GGF, global grid forum. Available: <http://www.ggf.org/> [01/22, 2006].
- GLOBUS, , the globus alliance. Available: <http://www.globus.org/> [01/22, 2006].
- HERRERA, J., HUEDO, E., MONTERO, R.S. and LLORENTE, I.M., 2004. Execution of typical scientific applications on Globus-based grids, 2004, pp177-183.
- HUEDO, E., MONTERO, R. and LLORENTE, I., 2005. An Evaluation Methodology for Computational Grids, September 21-23, 2005 2005, Springer-Verlag GmbH pp499.
- HPCC, HPCChallenge. Available: <http://icl.cs.utk.edu/hpcc/> [12/22, 2006].
- KENNEDY, K., 2004. Languages, Compilers and Run-Time Systems. In: I. FOSTER and C. KAUFFMAN, eds, *The Grid 2: Blueprint for a new computing infrastructure*. second edn. San Fransisco, UAD: Morgan Kauffman, pp. 491.
- KENNY, E., COGLAN, B., TSOULOUPAS, G., DIKAIKOS, M., WALSH, J., CHILDS, S., CALLAGHAN, D.O. and QUIGLEY, G.E.-., 2005. Heterogeneous Grid Computing: Issues and Early Benchmarks.
- KISHIMOTO, H., SAVVA, A., BERRY, D., DJAOUI, D., GRIMSHAW, G., HORN, B., MACIEL, F., SIEBENLIST, F., SUBRAMANIAM, R., TREADWELL, J. and VONREICH, J., 2005. *The Open Grid Services Architecture, Version 1.5*. GWD-I(draft-ggf-ogsa-spec-1.5-004). <http://forge.gridforum.org/projects/ogsa-wg/>.
- KOOPMAN, P. and MADEIRA, H., 2002. Workshop on Dependability Benchmarking, *DSN '02: Proceedings of the 2002 International Conference on Dependable Systems and Networks*, 2002, IEEE Computer Society pp790-791.
- MCCALPIN, J. stream benchmark performance results. Available: <http://www.cs.virginia.edu/stream/> [01/21, 2006].
- PENG, L., SEE, S., SONG, J., STOELWINDER, A. and NEO, H.K., 2004. Benchmark performance on cluster grid with NGB, 2004, pp275.
- SNAVELY, A., CHUN, G., CASANOVA, H., WIJNGAART, R.F.V.D. and FRUMKIN, M.A., 2003. Benchmarks for grid computing: a review of ongoing efforts and future directions. *SIGMETRICS Perform.Eval.Rev.*, **30**(4), pp. 27-32.

TOP500.ORG, the linpack benchmark.

<http://www.top500.org/lists/linpack.php> [01/21, 2006].

TSOULOUPAS, G. and DIKAIAKOS, M., 2003. GridBench: A tool for, 17 Nov. 2003 2003, pp60-67.

VAN DER WIJNGAART, ROB, F., , grid benchmark working group. Available: <https://forge.gridforum.org/projects/gb-rg> [01/22, 2006].

VAN DER WIJNGAART, ROB., F. and FRUMKIN, M., 2004. *ALU Intensive Grid Benchmarks*. GWD-1. Globul Grid Forum (2004).

WEICKER, R., 2001. SPEC HPG benchmarks: history, use and current issues. , pp. 20-39.

Definition of terms

Term	Definition
OGSA	Open Grid services Architecture - a pencil-and paper specification of Grid architecture with the view to standard
QOS	Quality of service
HPC	High Performance Computing
SOA	Service Oriented Architecture - a design based upon gluing together services with defined interfaces. Services have defined functions.
GGF	Global Grid Forum - A body leading the global standardization effort for grid computing.
SPEC	The Standard Performance Evaluation Corporation - an organisation formed to establish, maintain and endorse a standardized set of relevant benchmarks that can be applied to the newest generation of high-performance computers.
FPU	Floating Point Unit - component of a Central Processing Unit that has responsibility for floating point calculations
ALU	Arithmetic Logic Unit - component of a Central Processing Unit that has responsibility for Integer calculations
VO	Virtual organisation - An organisational entity or administrative domain that owns resources that it may share on a grid, but that have a separate set of access and usage policies from the grid at large.
SLA	Service Level Agreement