Project no. FP6-045520

EchoGRID

European and **Ch**inese C**o**operation on **Grid**

SSA Project

Advanced Grid Technologies, Systems and Services

D.2.1 – **Workshops Conclusions**

Due date of deliverable: 31 May 2007

Actual submission date: 11 July 2007

Start date of project: 1 January 2007 **Duration**: 24 months

Organisation name of lead contractor for this deliverable: STFC (was CCLRC)

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	R

Keyword List: Grid, open standards, programming, middleware
Responsible Partner: Keith Jeffery, STFC (was CCLRC)

MODIFICATION CONTROL			
Version	Date	Status	Modifications made by
0	DD-MM-YYYY	Template	Florence Pesce
1	10-06-2007	Draft	Keith Jeffery
2	11-07-2007	Review and contributions on security	Ignacio Soler (ATOS)
3	11-07-2007	Final	Keith Jeffery, STFC (was CCLRC)

Deliverable manager
Keith Jeffery, STFC (was CCLRC)

List of Contributors

- Depei Qian, Beihang University
- Bruno Le Dantec, ERCIM
- Keith Jeffery, STFC
- Thierry Priol, INRIA
- Zhiwei Xu, ICT
- Jinpeng Huai, BUAA
- Dimosthenis Kyriazis, NTUA
- Boaping Yan, CNIC
- Hai Jin, HUST
- Denis Caromel, INRIA
- Yongwei Wu, Tsinghua
- Minghui Zhou, PKU
- Kumardev Chatterjee, Thales
- Tianyu Wo, BUAA
- Ji Wang, NUDT
- Yan Zhu, BUAA
- Weiyuan Huang, Tsinghua University
- Jiangning Liu, CVICSE
- Ignacio Soler, Atos Origin
- Jacky Sheng, HUAWEI

List of Evaluators

- Ignacio Soler, Atos Origin

Summary

- The first workshop was held in Beijing in February 2007, associated with the project kick-off meeting. The major objective was to establish the state of the art in both Europe and China and to analyse similarities and differences, thus leading to first indications of a roadmap for further work. The workshop concentrated on 3 major topics: Grid Open Standards, Grid Programming and Grid middleware. In each case the issues were discussed, in particular considering the scientific and enterprise environments and their differences. It emerged that GRIDs success is hindered by multiple candidates for standards, multiple approaches to programming environments and multiple implementations of middleware. There is an urgent need for characterisation of these offerings to allow comparison of advantages and disadvantages leading hopefully to an agreed starting platform for further cooperative research and development. Key requirements are standardised metadata formats to describe services and resources and associated protocols for interaction between services, resources and services with resources.

Echogrid First Strategic Workshop

8-9 February 2007

Table of Content

1 Session 1: GRID Open Standards

1.1 Introduction

Weiyuan Huang (Tsinghua University) led this session. Three different current situations were analysed covering Orientware, the view of the company Thales and the project CROWN.

Orientware (presented by Minghui Zhou) is a Chinese middleware organization of academia and industry that uses the results of the Chinese 863 project, integrating them in the domain of middleware producing an industrial strength codebase. OW^2 is a cooperation with ObjectWeb producing middleware which could be considered a candidate for standardization and wide adoption.

Thales (presented by Kumardev Chatterjee) as a company has a vision for GRIDs involving multiple layers of virtualization. Building on the previous key strengths of the company, Thales is evolving towards a vision that is person-centric and utilizes a trusted collection of services. The NGG (Next Generation GRID) Expert Group Report 3 on SOKU (Service-Oriented Knowledge Utility) is key to the strategy.

The CROWN project (presented by Tianyu Wo) 2005-2007 provides a demonstrable job submission system. It has applications in weather forecasting, large-scale multimedia, bioinformatics (the BLAST software), in astronomical sky survey and others. It is based strongly on OGSA and so has a linkage with W3C (World Wide Web Consortium).

The key areas for discussions centred around virtualization, job submission, security and the enterprise grid versus the scientific grid.

1.2 Virtualisation

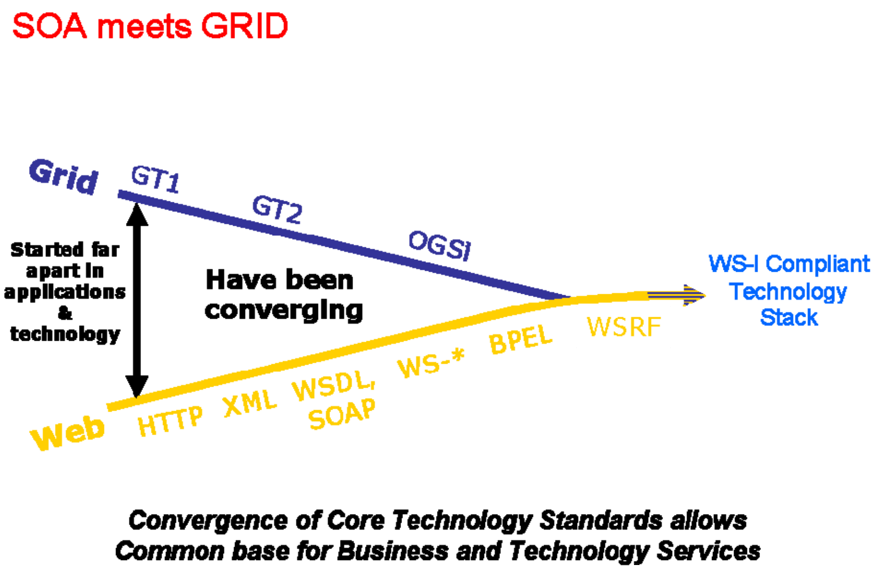
There were intensive discussions on what is virtualization and its advantages and disadvantages: it is clearly an advantage for ease of use by end-user, programmer or systems administrator but potentially at a cost of performance, flexibility and security. Then the discussion moved on to how to achieve virtualization – a complex systems engineering problem given the wide range of application domains with associated functional requirements and the expectations concerning non-functional requirements such as performance and security. It was noted that Google has an internal grid for resilience and flexibility and that this is virtualized from the end-user. Further discussion concerned benchmarking issues; which parameters are significant and sufficiently indicative to be useful as a guide to expectations of end-users? Finally portability was considered; again there are conflicts between portability (a kind of virtualization) with a general interface across multiple different platforms and efficiency / performance with targeted optimization for one platform.

1.3 Job Submission

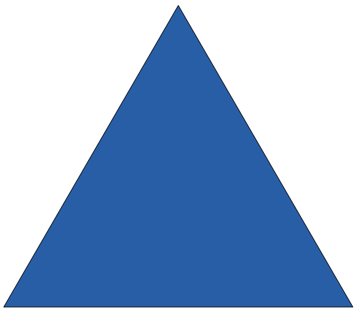
There are many offerings and no real standard. It was recommended that one (JSDL) should be chosen now from among the offerings and then used so ending the debate which appears not to be progressing. However, there are deeper issues. Given the direction of architectural development of GRIDs towards service-orientation, there are questions over the breadth of applicability of the job submission model –perhaps it is less applicable outside the scientific computation domain. What is important is the concept of remote process, allowing multiple parallel ‘submissions’; control and management of such a distributed environment is likely to require a workflow engine in order to keep the interdependencies – especially the preconditions – satisfied and the processes scheduled.

1.4 Security

As long as GRID is converging to WS-* standards there was a discussion on how the convergence is done, and if the standard should cover both of GRID and Web Services issues:



There was a wide discussion. Web services has the concept that security is trust plus access control. In the GRIDs environment security is in a triangle with trust and privacy; the problem is to satisfy appropriately the requirements of each apex of this triangle. It was recognized that B2B (Business to Business) and B2C (Business to Customer) have different characteristics with respect to this triangle (and also to transaction properties). A banking example was discussed to illustrate these aspects.



1.5 Difference between Enterprise and Scientific GRIDs

The first major difference discussed was the different infrastructure. Typically scientific GRIDs run over high speed dedicated academic/research networks supported by public funds, whereas enterprise GRIDs run over commercially supplied networking services and the performance depends on the perceived business value for the cost.

Another difference is in transactions; much present-day (but not necessarily future) scientific GRIDs computing is using batch job submission. This may well change in the near future to a much more interactive mode of use requiring access by schedulers to multiple heterogeneous resources for computation, data access and management, visualization, further data collection etc e.g. for computational steering, experiment control, analytical comparison of observed experimental results and in-silico experimentation. Meantime, the predominant form of transaction for enterprise GRIDs is the short information management transaction e.g. update a financial record. However, again, this will change in the near future with much longer, multi-level nested transactions. At one level in an enterprise GRIDs environment the transaction is complete only when the complete cycle order/invoice/goods receipt/payment is completed whereas with present systems each of these steps is treated as a separate transaction. Furthermore, management information queries in future may well involve not only statistical analysis of historical data but also simulation and prediction of future states of the business – but all within one environment and possibly one multilevel transaction.

A third difference is in software licensing and restrictions on use. The scientific GRIDs community tend to use open source software in a sprit of cooperation for advancement. The Enterprise GRIDs environment, on the other hand, requires the security of commercial guarantees of software quality and behaviour together with the commercial security of not revealing the software used to competitors.

However, despite their separate starting point characteristics there is some optimism that there will be convergence and in the near future there will be many more similarities than differences and that each environment can benefit from thinking, planning and experience related to the other.

2 Session 2: GRIDs Programming

2.1 Introduction

Thierry Priol led this session. It considered two GRIDs programming environments and then concluded with a more general discussion of the environment for GRIDs programming.

2.2 ProActive

Denis Caromel described the ProActive, object-oriented, Java-based GRIDs programming environment. This allows a component-based approach developed within CoreGRID and used in GRIDComp. There are issues concerning the programming environment, latency and protocols to control the distributed parallel execution.

2.3 iVCE

Ji Wang described the iVCE environment developed under the Chinese 973 programme. It is based on the internet programming concept and includes a metamodel (currently under construction) which covers autonomic elements, a virtual commonwealth of resources and virtual executors. The key issues concern the metamodel; how to describe the resources and how to wrap/virtualise them so they can act autonomically and be aggregated and scheduled at runtime.

2.4 GRID Programming

Thierry Priol introduced this more general topic taking an overview of the environment required. At present there are different approaches to GRID programming which are dependent on the infrastructure in each case leading to programming silos which lack interoperability and generality. There is a move towards the acceptance of the service concept (analogous to web services in the WWW environment) but the concept needs to be distinguished carefully from the concepts of objects or components which have some similarities but also some important differences. The clear specification of these differences should lead to a convergence in programming environments towards the common goal.

2.5 Discussion

This leads to some principles:

1. current programming models have insufficient abstraction; we require a greater level of abstraction;
2. GRID programming must be independent of the underlying infrastructure to allow portability and interoperability;
3. because GRID infrastructures are (or should be) dynamic with self-* properties, then there are implications for the programming environment:
   a. the infrastructure resources must not be managed by programmers but hidden below a middleware layer providing infrastructure-independence (analogous to device independence in operating systems);
   b. the applications program should not bind to a particular instance of a service but to a general service definition which discovers and binds the specific service at run time(and even can change during execution) using the middleware which itself is using metadata describing the service including functionality, performance characteristics, security/privacy/trust characteristics, any restrictions of use including charges and preconditions for execution;
4. we should be courageous and investigate new disruptive approaches; it is not certain that existing incremental approaches will provide the solution. Examples include inspiration from nature – bio-inspired computing – from behaviour of biological cells and intra-cell behaviour through to the behaviour of groups of individuals in colonies or swarms;
5. this leads to the underlying idea that computer SCIENCE is required to solve the problems, not computer ENGINEERING. The engineering challenges come once the theoretically correct background is defined.

3 Session 3: GRIDs Middleware

3.1 Introduction

This session was introduced and led by Yan Zhu from BUAA-ACT. It considered existing middleware implementations (CNGrid, Vega GRID, NextGRID and OGSP) and drew general conclusions.

3.2 CNGrid and Vega GRID

This topic was presented by Zhiwei Xu. There was discussion of the concept and implementation of GOS (GRID Operating System) with its functional and non-functional requirements. There is a project plan through 2007 to 2009. There was discussion of industry involvement and contributions leading to a discussion of trends – in which direction is GRIDs middleware going? This can be analysed from three angles: capabilities; system abstraction and the properties and laws. The capabilities need to describe what the middleware can do and how it can be used to manage the resources – mainly through some kind of metadata.

The system abstraction angle indicates the power the middleware offers to the applications programmer – the more the abstraction the less the applications programmer is concerned with managing the infrastructure and the more she can concentrate on the application itself. The properties and laws provide an interesting and formal description of the constraints associated with the capabilities and assist in raising the level of abstraction; furthermore they ensure (if constructed correctly) integrity and correctness in the implementation of the application.

3.3 NextGRID

Dimosthenis Kyriazis presented NextGRID; aimed at being the GRIDs architecture of the future. NextGRID has an advanced architecture. Despite business successes with cluster GRIDs, the idea of NextGRID is to take GRIDs outside the company firewall (i.e. to allow cooperative business to business interworking). The key elements are functional systems (provide the functional services), management systems (manage service level agreements, resources), orchestrators (to compose and allocate services), schemas (to provide the information for the other components).

3.4 OGSP

Weiyuan Huang presented this environment providing interoperability of CGSP. It is the ChinaGRID support platform. Its characteristics were presented and discussed; the abstractions were presented and case studies were presented and discussed revealing deeper insights into the environment. In particular the interactions with GridCOMP were discussed including the steps for submitting a job to CGSP from GridCOMP.

3.5 Discussion

The major discussion centred on a review of the current state of the art. The problem is that we have too many approaches to - and implementations of – middleware. We require some method to characterize them all in order to discuss their merits and disadvantages and to identify overlaps and differences. This should lead to a general agreement on which provides the best platform for further coordinated work.

However, the growth of importance in service-oriented architectures based on web services may overtake our efforts. This was foreseen by European researchers in GRIDs in the late nineteen-nineties and at GGF (Global GRID Forum) they pressed for an acceptance of the service-oriented approach leading to OGSA (Open GRID Services Architecture) and OGSI (Open GRID Services Interface) in 2002.

4 Overall Conclusions

The major conclusion is that there exists a large body of heterogeneous, uncoordinated or partly-coordinated research and development leading to multiple candidates for standardization, multiple programming environments and multiple implementations of middleware. They are usually non-interoperable and non-portable across different infrastructure environments. This depressing picture is perhaps not unexpected at this relatively early stage in the development of the technology, although it is 10 years since the first metacomputing GRID software was developed.

However, principles for the future start to emerge. In particular:
1. Architecture: there is now general acceptance of the need for integration in a service-oriented architecture, thus disqualifying environments based on specific binding of applications to underlying middleware and infrastructure;

2. Programming Environment: similarly there is growing acceptance of the level of abstraction that should be presented to the applications programmer – the middleware interface should hide underlying complexities in resources;

3. Metadata: there is agreement on the need for metadata to describe services and resources, but no general agreement on a standard for this metadata. This perhaps is the major area where a contribution could be made since it is central to the required architecture for interoperation, portability and infrastructure-independence;

4. Protocols: there is a need for standardised protocols for managing the interactions between services and resources, between services and services and between resources and resources; this links with (3) above.