

1 Introduction

Following the success of our 2003 tutorials on Globus Toolkit 3, UK Globus Week was held during the first week of April 2005. This report details the outcome of the meeting and defines steps for future work and collaboration.

The Globus Toolkit (<http://www.globus.org/toolkit>) provides libraries and components that enable the development of service-oriented Grid applications and infrastructures. Core Globus components address basic issues relating to security, resource access and management, data movement and management, resource discovery, and so forth. A broader Globus ecosystem includes numerous tools and components that build on core Globus functionality to provide application-level functions.

Globus Toolkit version 4 (GT4), in beta at the time of the meeting and released as a final version on 30 April 2005, provides a range of new services and features. In addition, by delivering the first robust implementation of Globus Web services components, GT4 completes the first stage of the migration of Web services that began in 2003 with GT3. Thus, it was timely to convene a meeting, both to communicate the latest status of Globus software and to enable intense conversation between Globus developers and users concerning future directions for open source Grid software.

The UK Globus Week had six goals:

- Bring together people from the UK, Europe, and globally with a significant interest in the Grid; the Globus Toolkit; and stable, standard, and cost-effective high-performance computing.
- Present Globus developments and research, and demonstrate case studies of academic and commercial Globus use.
- Provide a program of tutorials, workshops, questions & answers, and demonstrations from the Globus Alliance, tailored to the needs to the UK Grid community.
- Develop a roadmap for continued Globus development, deployment, integration, and support, drawing on skills and experience from within institutions, e-Science projects, the Grid Operations Center (GOC), UK National Grid Service (NGS), and UK Open Middleware Infrastructure Institute (OMII).
- Stimulate the formation of a community that will undertake this work.
- Document the event's proceedings as a UK e-Science technical report.

Presenters for the tutorials included Globus Alliance members Carl Kesselman, Bill Allcock, Charles Bacon, Lisa Childers, Jarek Gawor, Jennifer Schopf, Amrey Krause and Kostas Tournas and members of the UK e-Science community who discussed their experiences with the software.

This report is structured as follows. Section 2 gives the abstracts of the talks presented by UK e-Science researchers, which focused on their experiences and the future needs for Globus software. Section 3 summarizes the work of the breakout sessions, which

discussed potential areas of collaboration and needed future work. Also summarized are the points made during the panel discussion on interoperability. Section 4 provides a set of useful links to associated material.

2 GT2, GT3, and GT4: Experiences and the Future

Nine presentations focused on experiences with Grid middleware and Web services and with various Grid environments.

2.1 David Wallom, The University of Bristol Campus Grid: Production Using a PreService-based Architecture

Within the University of Bristol it has been historically true that each research group has specified, purchased, managed, and used their clustered resources. This situation has led to their being about 15 clusters and 4 Condor pools within various differing departments. Hence, one of the key tasks for the e-Science centre was creation of a campus grid, to unify access and increase system usage levels. Since its creation we now have five different groups of users with up to six different applications already ported. These results show the success of the system and its possible expansion towards other universities within the region.

I will describe the tools chosen for each component of the system, with experiences of what we have found good and bad about the tools described. In particular, I will focus on tuning that was necessary for scalability.

2.2 Mark Calleja, Globus and the eMinerals Project

The eMinerals project has been operating a production minigrid for some time now. This minigrid is built from the three middleware pillars of Condor, Globus, and the SRB. In this talk I will describe how these have been integrated in order to make the middleware as transparent as possible for the end users, with emphasis on how Globus was made to fit in. In particular, some modifications that were found to be necessary will also be described.

2.3 Stephen Pickles, The National Grid Service and Globus

The National Grid Service (NGS) provides computational and data Grid infrastructure to the UK e-Science community. The NGS is run by the Grid Operations Support Centre (GOSC). The core NGS facilities – data nodes at RAL and the University of Manchester, and compute nodes at the Universities of Oxford and Leeds – are funded by the JISC and CCLRC. The UK national high-performance computing facilities CSAR and HPCx are also part of the NGS. The NGS is expanding the range of facilities it provides through a partnership programme.

In this talk, I will briefly describe the NGS, the GOSC, and the services that they offer to end users and to projects that wish to provision their own services using NGS resources. I will review the issues that arise when trying to evolve the middleware infrastructure of a production Grid, while expanding the deployment base, offering

sensible migration paths to end-users and resource providers, and preserving interoperability with peer Grids. I will discuss how these issues give rise to requirements on the Globus Toolkit – flexible deployment options, continued support for pre-WS components, balance between stability and new capabilities, good packaging, and so on. I will also discuss how the lack of middleware support for advance reservation and coallocation is retarding imaginative use of the Grid.

2.4 Conrad Hughes, Dependable Grid Services: Composing Services against Quality-of-Service Requirements

The DIGS project seeks to automate as much as possible the composition of services in pursuit of quality-of-service requirements (such as service level agreements) through choice of appropriate components (diversity) and modification of workflow (structure). This automation can be done only if the effect of differing service choices and workflow modifications can be understood. In some of our most recent work we offer a tool that predicts the aggregate behaviour of a composition in terms of its workflow structure and the known behaviour of its individual services. By using such predictions, compositions can be optimised to match service level agreements, affording a critical tool in the future development of distributed applications.

During 2003 the project was entirely based within the Globus Toolkit 3 framework (from alpha release onwards), encompassing three different development strands engaged with notification, dynamic invocation, and service proxying. In this presentation I will give an overview of the project, a taste of our current work, and a review of our experiences with GT3.

2.5 Terry Sloan, GT 2 and 3 Experiences on the INWA and HPC Europa JRA2 projects

The INWA project has used GT2- and GT3-based tools to create a Grid for analysing commercial data located at sites in the UK and Australia. A site in China has recently been added to this Grid with a view to analysing commercial data from there too. The HPC Europa JRA2 project is building a single point of access portal for visitors to European HPC centres. As part of JRA2, EPCC is integrating the GT3-based Job Scheduling Hierarchically (JOSH) tool with this portal. In this presentation I will describe the experiences of the INWA project team using GT2 with the Grid Engine Transfer-queue Over Globus (TOG) tool. I will then describe the experiences of HPC Europa JRA2 team when developing and deploying JOSH using GT3. Finally, I will provide a brief wish list for GT4.

2.6 Terry Harmer, UK Engineering Task Force GT4 Evaluation

I will provide an overview of the UK Engineering Task Force's GT4 evaluation that has been in progress since December 2004. The presentation will cover experiences with installation and configuration of GT4, security, and GRAM and experiences with migration from GT3 to GT4.

2.7 *Theirry Delaitre, Experiences with Migrating GEMMLCA from GT3 to GT4*

Relatively little attention has been paid to how end-users can survive in the rapidly changing world of Grid generations. The primary goal of our research is to construct a high-level Grid application environment where the end-users can employ and use any legacy code as Grid services and can easily and conveniently create complex Grid applications.

In an ideal Grid environment users would be able to access Grid services through a high-level, user-friendly Grid portal. More than that, users would not only be capable of using such services, they could dynamically create and deploy new services and also construct complex Grid workflows in a convenient and efficient way. All these services would be either specifically designed Grid services or legacy code programs deployed as Grid services.

The research team in the Centre for Parallel Computing at the University of Westminster has developed GEMMLCA (Grid Execution Management for Legacy Code Architecture), a system to enable legacy applications to be quickly and efficiently deployed on the Grid via a user-friendly Web interface. With GEMMLCA, existing programs written in any language (Fortran, C, Java, etc.) can be deployed very easily as a Grid service by end users having no programming knowledge. The user-friendly Web-based access to GEMMLCA has been provided by integrating the tool with the P-GRADE Grid portal and workflow system developed by SZTAKI Research Institute in Hungary. Utilising the portal, users having little or no knowledge of Grid systems can deploy old legacy codes as new Grid services, simply by selecting Grid resources and attaching the codes. With modest know-how, they can even construct and execute workflows that support complex processes.

In this talk I will explain the importance of porting legacy applications onto Grid platforms; introduce GEMMLCA, a high-level solution that supports this task; report on experiences on migrating GEMMLCA from GT3 to GT4; and demonstrate the integrated GEMMLCA P-Grade portal solution.

To demonstrate the integrated GEMMLCA P-Grade portal solution, we create a workflow for analysing urban car traffic on road. The workflow consists of three components: a Manhattan road network generator, traffic simulators called MadCity, and an analyser. The Manhattan road network generator creates MadCity-compatible network and turn input-files. The output of this component is used as input to MadCity traffic simulators that run parallel on UK e-Science OGSA Testbed sites with different numbers of cars as input parameters. After the simulations are completed, the generated trace files are input to an analyser that compares the traffic density on roads and illustrates the results on graphs.

2.8 Neil Chue Hong, OGSA-DAI: Responding to a Changing World

OGSA-DAI is a UK e-Science-funded project to provide access to data resources on Grids. The project was originally based on GT3 and has moved forward to use, amongst others, the early versions of GT4. In this talk I will look at the process of moving data services written in GT3 to GT4 and the teams' experiences with porting and supporting code, as well as discussing the issues with continuing support for an established GT3 user base.

2.9 Bill Allcock, Report on GT4 Performance Tuning Activities and External Security Review

In the development of version 4 of the Globus Toolkit, several areas have received unprecedented levels of effort. We have striven to make the process more open by holding public phone conferences, community calls for testing, and having an external review of the security architecture. Performance has also received never before seen levels of effort. In this case, we take performance in its broadest sense. We include robustness, stability, throughput, and user experience. In this talk I provide a brief overview of the results of the external security review that was performed and the performance efforts that have taken place to date.

2.10 Lisa Childers, Discussion of Process

The Globus Toolkit is an open source software toolkit used for building Grids. It is the flagship product of the Globus Alliance, which is an international collaboration that conducts research and development to create fundamental Grid technologies. The Globus Alliance welcomes community contributions to the toolkit. In this talk I provide a brief overview of the ways in which members of the community can contribute to toolkit development.

3 Breakout Session Summaries

This section summarizes the discussions of the five breakout sessions and the panel.

3.1 Data Management: Bill Allcock, chair

The purpose of this breakout session was to address three questions:

- What services and software are already available?
- What services and software could UK/U.S. research efforts add in the near future?

In what areas were there obvious gaps, and where could we have a reasonable chance for success in joint funding proposals?

3.1.1 What services and software are already available?

The Globus Toolkit provides GridFTP for secure, fast, robust, byte-oriented data movement over the wide area network. The Reliable File Transfer (RFT) service provides a Web Services Resource Framework (WSRF)-compliant service interface to data movement and provides database persisted state for increased reliability in the face of service failure. The Replica Location Service (RLS) provides robust, distributed, scalable service that provides mappings from Logical File Names (LFNs, which are arbitrary strings) to some other arbitrary string, but one that allows access to a physical instance of the file of interest. This could be a simple URL or a string that may need to be passed to some other service for translation prior to access. =The Open Grid Services Architecture – Data Access and Integration (OGSA-DAI) is a UK development that currently holds the status of tech preview within the Globus Toolkit; it provides access to databases and files through a document-based interface. These documents allow a series of activities to be pipelined together and executed at the service, thus avoiding unnecessary data movement. The framework is extensible, allowing new functionality to be introduced.

Other services and software that are currently available and are seen as filling critical pieces in the architecture include the Data Format Description Language (DFDL), which provides an XML-encoded description of a binary data stream or file (schema for non-database data), and the Storage Resource Management (SRM), which allows for space reservation, staging from mass storage, and some optimisation of storage usage.

3.1.2 What services and software could the UK/U.S. research efforts add in the near future?

Among the services needing additional features or hardening is the Data Format Description Language, which is in its early stages and will need further refinement and development.

Application-level quality-of-service features that are planned include bandwidth throttling for GridFTP, inclusion of priorities in data movement services, replica coherence, and transport modalities that optimise 1 to n data distributions (multicast-like, but without depending on IP multicast).

3.1.3 In what areas were there obvious gaps, and where could we have a reasonable chance for success in joint funding proposals?

While there exists strong work in both bulk data (GridFTP, RFT, RLS) and structured data (OGSA-DAI), ultimately there needs to be a seamless bridge between these forms of data. We envision a point when there is complete data virtualisation and the user need not know what is in a database or a file. The metadata, the structure of the data, and key characteristics such as the relative “closeness” or “cost” of the data will be what the user responds to. Achieving this objective will also require easy and effective mapping of different types of schema.

One key issue that arises as we begin to use a service-oriented architecture and multiple levels of virtualization is performance and efficiency. We believe there will be a need for an extremely rich service level agreement (SLA) negotiation mechanism. For instance, each service being considered for use could be presented – if not the entire job, at least large pieces of the job – so that it could “bid” on one or more pieces of the job. This capability could prevent extreme losses of efficiency when a single service could pipeline multiple stages of a job but, for lack of broader information, each stage is sent to the same service sequentially.

We also envision the need for more efficient communication between services in the workflow, possibly using their “locality” (i.e., is the service I am communicating with in the same process space, on the same node, in the same cluster, or half-way around the world), to negotiate transport mechanisms (for example, shared memory if in the same process space, GridFTP if over the WAN).

We note that SLA’s and the description of communication between nodes in a workflow necessarily involve the execution management/workflow community. We need to be a vocal user community providing input to the execution/workflow developers to ensure that the services and protocols developed allow us to get maximal advantage from our services.

Besides those issues related to the functionality of existing services, services will need to expose more operational state and provide properly secured interfaces for altering that operational state. These capabilities will greatly enhance our ability to account for Grid resource usage as well as troubleshoot problems that occur during the execution of Grid operations.

As Grid applications grow and mature, one of the primary distinctions between the more mature Grid services and software and earlier versions will be their ability to provide nontrivial qualities of service. One manifestation of this will be that more and more services and resources will be managed, have standard interfaces, and be able to provide reservations for resource consumption. We believe that existing protocols and service interfaces are not sufficiently general to support reservations of resources and that work will be required to define how a reservation can be passed to a service in a general way and how the service may then effectively make use of that reservation.

In an SOA, the composability of services is critical to being able to build up complex functionality. As we gain experience, “callouts” to certain service functionality will be critical for the broad acceptance of services. Security callouts for mapping of users to local accounts and fine-grained authorization for resource access are already showing their utility. Service developers will need to keep abreast of these developments and integrate such callouts into their services in a timely manner. One of them that we believe will become necessary is a semantic, or ontological, translation callout. We do not envision data management driving the development of these ontologies, but we will likely need to be able to work with them, and having a callout so that by simply writing a translation from one’s application community specific ontology to the ontology that the service supports will greatly increase the utility of the service and ease its integration into new communities. Other potential

callouts that may gain wide utility include callouts to signal the consumption of a resource reservation and the automated generation of metadata.

While the above covers a fairly broad gamut of potential research areas that apply to data management, we also provide a narrower scope here that identifies areas that we feel would be appropriate for seeking joint funding in the next 24 months:

1. Integration of bulk and structured data services. This could include expansion of the use of delivery services and necessary additions to bulk services to understand structured files via the use of something like DFDL.
2. Additions to workflow systems that support data management requirements.
3. Higher-level replication services that include support for reliable replication, and replica consistency services.
4. Exposure of operational data and management interfaces to control it: specifically, working to enable proper accounting and troubleshooting.
5. Development of efficient data transfer methods between services including negotiation of transfer method and knowledge of “locale”.

3.2 Job Submission: Peter Lister, chair

The Job Submission session started with a discussion of perceived needs. Lister presented RAL’s Web portal, which requires time-critical scheduling. Wallom presented brokering work from Bristol. Other projects have devised similar solutions, performing data staging “by hand” using home-brewed scripts within Web front ends.

Several groups brought up the issue of workflow. There was a request for better documentation of workflow solutions within GT4, especially those that worked with Condor or DagMan. Pegasus was mentioned as a way to integrate information for use by DAGman. Wallom mentioned work at Bristol that used information from Globus MDS to control DAGman, and indicated that he would be happy to pass this work back to the Globus team. A list of recommended configurations that are known to work together was requested.

There was a discussion about the functionality that GRAM was supposed to cover. In general, there was a misunderstanding about advance reservations – GRAM was not meant to control them, but one could interact with the advance reservation capability offered by PBSPro of LSF through GRAM’s resource specification language (RSL).

3.3 Monitoring and Discovery: Jennifer M. Schopf, chair

Monitoring was defined as the gathering of data to make smart decisions. The collection of information was clearly seen as a separate issue from the performing of actions (such as scheduling or administrative tasks).

Schopf gave a brief overview of the current state of MDS4, the Monitoring and Discovery System that is part of GT4. More information can be found on the Web at <http://www-unix.mcs.anl.gov/~schopf/Pubs/mds-sc05.doc>. Three main open issues were identified: the need for additional data sources, application monitoring, and background behaviour monitoring.

3.3.1 Additional Data Sources

Interfaces to MonaLisa, Nagios, Inca, the Netperf archive, and the NetLogger archive are planned. However, additional sources are also needed, including Big Brother and possibly UDDI. UDDI is needed when a group is already publishing its service data to UDDI but would like to access the data through a WS-RF standard mechanism.

3.3.2 Application Monitoring

Application monitoring was recognized as an important open question. There is a strong need to know what's going on with an application – where it is in its life cycle, how it's interacting with the services and resources – but no one has begun to address this yet (to the knowledge of the group present). Currently, the best solution is simply looking at output files.

A fair bit of discussion focused on whether application monitoring belongs in a Grid service, with the agreement that if it is application-specific in terms of a framework, it is probably not suitable. One thought was that a set of predefined files could be monitored if they had data streaming to them about application progress; the Chirp technology from Condor might be a starting point. It might be possible to do some kind of framework that could interact with PBS, Condor, and the like – but this would need funding for a collaborative proposal.

A simpler solution would be to have MDS4 interface to NetLogger archives and then to have applications instrument themselves as needed using NetLogger. This might require more work on the applications side, and there may be issues with respect to the simplicity of the NetLogger output, but it was seen as a feasible starting point.

3.3.3 Background Behaviour Monitoring

The group clearly agreed on the need for better understanding of background behaviour, specifically tests to show the performance of a wide set of systems, which are then examined over time see whether changes are happening.

This monitoring capability would require sites to have a well-defined set of performance benchmarks on all their resources in order to determine a baseline, something several groups (including Casanova at UCSD and the Inca group, part of TeraGrid) are already doing. Analysis would come in two flavours – quick checks, which look to see whether there's an immediate problem, and then longer-term variation analysis, which send warnings if values are out of some defined scope.

3.4 Virtual Organisation Management for Community Grids and Related Globus Requirements: Michael Griffiths, chair

Grid computing has been defined as the flexible, secure, coordinated sharing of resources amongst dynamic collections of individuals, institutions, and resources. Such collections are referred to as virtual organisations (VOs). Community Grids are an example of Grid computing infrastructures that enable the sharing of resources with trusted external institutions. The management of virtual organisations is a complex task, with a large number of technical and political issues. A necessary component of a Grid development toolkit is a set of tools for managing VOs; such tools would ultimately enhance the quality of service for users and relieve workload for Grid administrators. Clearly, the successful development and testing of such tools depend on a collaborative development effort.

The takeup of services provided by community Grids depends on the ease with which users may register for access to those services. For many services this registration may require a number of stages, including registration to use local services, registration for X509 credentials, and registration for the community Grid. Such a procedure is cumbersome for both users and administrators and is a barrier to the takeup of services offered by community Grids.

Within Globus Toolkit-based community Grids, a problem for administrators is the maintenance of up-to-date grid map files. This problem has been recognised by a number of organisations, and the European Data Grid in particular has developed the Virtual Organisation Management Service (VOMS) to enable effective management of resources across the VO. There exist major difference between VOMS and the Globus Toolkit Community Authorisation Service (CAS). One such difference is that CAS relies on the issuing of new proxy certificates; hence, services that are not CAS enabled are unable to determine the identity of users to whom CAS has granted a certificate. The following requirements are important for enabling interoperability and effective management for dynamic, flexible virtual organisations.

- Support interoperation with additional authentication technologies
 - Shibboleth
 - Plug-ins for Community Authorisation Service
 - Permis
- Provide tools enabling update of resource grid map files from a centralised service
- Cast certificate management services as grid/web services
- Provide pool accounts
 - Grid services enabling allocation of pool accounts to authenticated Grid users, with well-defined usage policy and service lifetime
- Enable flexible VO operation
 - Tools for supporting asymmetric VO operation
 - VO lifecycle tools

Another method for managing VOs is to use location-independent networking (LIN). LIN enables a user to visit participating organisations and to gain access to the Internet without having to register for another username and password. The

service is a currently being developed in conjunction with UKERNA and a number of other UK higher education institutions. One avenue to explore is providing tools to enable integration of Globus tools with location-independent naming services.

3.5 Deployment, Packaging, Build & Test: Steven Newhouse, chair

The focus of much of the discussion was on deployment and packaging issues. Steven Newhouse and Charles Bacon presented an overview of the NMI's Build & Test framework (<http://www.cs.wisc.edu/condor/nmi>).

Several requirements emerged from the discussions that should form the basis of future work within Grid middleware distributions:

- The client and server installations should be considered as two distinct activities. A different installation process is needed because of the potentially different skills of the user (client) or system administrator (server).
- Supporting all possible local environments and policies in large software deployments would be effectively impossible. A better approach is to provide a distribution that can be easily reconfigured or adapted to support customised deployments within an organisation.
- Support of multiple installations on both the client and server is needed. Production deployments may include deprecated, current, and new installations. Clients will need to be easily reconfigurable and easily switched from one to the other.

3.5.1 Deployment

Three uses cases were identified to motivate the deployment discussions:

- **Large-scale Linux deployment:** Within an organisation there will be a need to provide a client environment across many desktop systems that can be used to access Grid services directly through middleware or tools such as GSI-SSH. This software may be deployed in an ad hoc manner by the primary local user who has administrative access to the system, in a systematic manner through centralised computer support, or on demand deployment into user space through systems such as Java Web Start.
- **Server installation:** Installation needs to be simple, highly automated, and repeatable. Once the server has been installed, the configuration should be simple and successful operation verifiable by postinstallation tests, which should exercise both the services and the supporting security infrastructure (e.g., a “real” certificate). The server installation should use existing software whenever possible (e.g., the patched GSI OpenSSL issue) to let subsequent package upgrades coexist alongside existing local procedures.
- **Grid middleware:** Middleware such as GEMCLA needs standard mechanisms to deploy GEMLCA (Web) services in a user-space container and to allow the administrator to customise the configuration of the container in a user-friendly way.

These use cases provided some high-level requirements:

- Separation of client and server bundles that can be easily specialised to support local deployment environments. These bundles should be fine grained to enable the deployment of, say, just a binary bundle containing a Web services hosting environment.
- Separation between installation (getting the software into the machine in a specified location), configuration (either automatically or manually and as the administrator or normal local user) of the installed software and verification of the complete configured installation.
- Simplified verification of the installation. Either an existing high-value certificate specified during the installation should be used (eliminating the need for alternative packages, e.g., SimpleCA, to be installed), or a low-value certificate should be used to verify the installation and then replaced by a high-value certificate at a later date, at which point the installation should be reverified.
- Ability to locate installations from any specified install root.
- Ability for multiple live installations to exist on a single machine within isolated installation trees.

3.5.2 Packaging

Several packaging systems were identified that are in common use to support the deployment of Grid middleware. Support for both Windows and UNIX systems was felt to be a high priority, but this did not necessarily mean that a single system had to be used on both platforms; indeed, the support models could be very different.

- The standard GNU ‘./configure; make’, which is recognised for its flexibility (e.g., subtargets) and portability for both source and binary installations.
- GPT (Grid Packaging Tool), which supports installation under an arbitrary file path that can be relocated only through reinstallation.
- PACMAN, which uses a cache-based distribution mechanism to obtain packages. Its popularity may be due to the good postinstall scripts associated with the packages rather than the caching architecture that could be obtained through tools such as apt.
- Configuration management systems, such as RPM, Yum, and MSI through the WIX (Windows Installer XML <http://sourceforge.net/projects/wix>), which are integral to many different distributions. These could be used by Grid middleware to ensure its complete integration with the installed system software.
- Simple archiving software such as ZIP and tar.
- Graphical installers, such as InstallShield.

Being able to interrogate metadata associated with the installed software and the packages under consideration for installation was seen as highly desirable. This would allow the installation “agent” to install and verify the packages that are needed to be provide a particular capability and to determine how that capability could be provided from the available packages. Such an architecture could be used support the deployment of services (and supporting infrastructure) within a Web services hosting environment.

The packaging structure used with the Globus Toolkit was outlined. GPT is used as the primary package management system for both source and binary distributions. Binary RPMs can be generated from the binary GPT packages. From this baseline separate client and server packs could be built, although the client distribution (with only the server executables removed) would not be significantly smaller due to the shared infrastructure between the client and server (e.g., both need a container to support notifications).

3.6 Panel: Interoperability between Grid Middleware systems

Moderator: Carl Kesselman

Jarek Gawor - WS-RF

Steven Newhouse - OMII

Erwin Laure - EGEE

Mark McKeown - WS-RF Lite

Savas Parastatidis - WS-GAF

Carl Kesselman opened the session with a short explanation of two key terms:

- *Interoperability*, which allows different implementations to talk to each other across the network, regardless of which language the implementations are written in or the API they present to the rest of the program.
- *Portability*, which makes functionality available to applications, via a specific language and API, regardless of the underlying implementation.

Both interoperability and portability are important.

The panel discussed which specifications are used by the different middleware offerings. As expected, there are many areas of consensus and some disagreements. All agreed that the goal should be to use stable, widely used specifications. The areas of disagreement arise when judging which specifications are stable and how to lead the way in areas that are not addressed by existing systems. For example, Globus and WSRF Lite use WSRF, while currently the OMII and EGEE systems do not. The panel did not delve into details about how these systems might be made to interoperate.

There was a discussion about naming schemes. All the systems use different naming schemes. The panel agreed that middleware would have to deal with multiple naming authorities and that it would be useful to adopt a standard, extensible naming scheme. URIs were singled out as a potential candidate.

The panel also agreed about the utility of APIs to shield applications (and hence users) from changes in the underlying services. This benefit of APIs is in addition to the provision of portability.

4 Related Material

The workshop Web page: <http://www.nesc.ac.uk/esi/events/519/>

All talks from the workshop:

<http://www.nesc.ac.uk/action/esi/contribution.cfm?Title=519>

The Globus Toolkit: www.globus.org/toolkit/

Ian Foster, on GT4: <http://www-128.ibm.com/developerworks/grid/library/gr-gt4/>

Terry Harmer, Anthony Stell, David McBride
UK Engineering Task Force Globus Toolkit Version 4 Middleware Evaluation
UK Technical Report UKeS-2005-03 available from
http://www.nesc.ac.uk/technical_papers/UKeS-2005-03.pdf, June 2005

GT3.2 to GT4 migration guide: <http://www.qub.ac.uk/escience/howtos/index.php>