Laying the Foundations for the Semantic Grid

Steven Newhouse, Anthony Mayer, Nathalie Furmento,
Stephen McGough, James Stanton and John Darlington
London e-Science Centre,
Imperial College of Science, Technology and Medicine,
180 Queen's Gate,
London SW7 2BZ, UK
icpc-sw@doc.ic.ac.uk
http://www-icpc.doc.ic.ac.uk/components/

Abstract

Information relating to the resources, applications and the user's wishes are key to the transparent and effective exploitation of the federated resources within Computational Grids. These federated data, computational or software resources are owned by real organisations and made available as services to different computational communities or virtual organisations spanning multiple administrative boundaries. Higher-level services use the information relating to the resources' capability and the mechanisms for service fulfilment to automatically discover interoperable services and select the most appropriate service for the user with minimal human intervention.
Within this paper we describe a service-oriented Grid architecture that utilises existing web service protocols to federate resources into computational communities. The service-oriented information contained in the computational communities is exploited by higher-level services to ensure effective utilisation of the resources by both its providers (the resource owners) and consumers (the users). We believe the systematic definition, presentation and exploitation of such information constitutes the first step towards the construction of a Semantic Grid.

1 Background

Computational Grids have provoked widespread interest within the scientific and engineering community over the last decade (1). There are now many global projects focussing on the development of Grid middleware and the mechanisms needed by the applied science community to effectively exploit these infrastructures (2). These projects are all being driven by the needs of a diverse applied science community involving both data and compute intensive applications (e.g. EU Datagrid, PPDG, Griphyn, NASA's Information Power Grid).

Heterogeneous resources within these Grids are forged into a single virtual organisation through the use of middleware. The Grid middleware collects and publishes information relating to the resources within the organisation while providing secure platform neutral interfaces to the underlying resources. Globus is one such widely deployed middleware that uses the Metacomputing Directory Service (MDS) to hold and distribute information through a hierarchical network of MDS server within a virtual organisation (3). Access to the resources within a virtual organisation is controlled through a 'gatekeeper' which verifies a user's identity through an X.509 public key certificate.

The specification, development and standardisation of Grid related protocols is taking place through the Global Grid Forum (http://www.gridforum.org/), formed through the merging of activities in North America, Europe and Asia. Its role, through the activity of its research and working groups in areas such as security, information management, applications, etc., is to provide a collaborative forum for researchers in industry, computer science and the applied science communities. Its meetings now attract several hundred researchers from around the world.

The activity within the Grid community can be compared to that within the Web community a decade ago. At that time there was an active worldwide research community developing competing and incompatible protocols, and innovative functionality within the server and client browsers. Coordination and standardisation of these activities was left by the community to the World Wide Web Consortium (W3C). Since 1994 it has guided the evolution of the Hyper-text Markup Language (HTML) and produced the several new standards such as the Extensible Markup Language (XML).

The recent emergence of business to business e-commerce has exposed the limited capabilities of existing web protocols when used to develop higher-level services. Currently, the division between page content and its presentation is frequently blurred within HTML encoded web pages. The W3C has been instrumental in developing approaches that enable a clear separation between the content (encoded as an XML schema) and its visual repre-

sentation using existing HTML in a browser. This approach enables other tools, say for the visually impaired, to browse and comprehend the page. The ability to directly describe the content provided through the web is seen as the first stage towards the construction of the Semantic Web (http://www.w3.org/2001/sw/).

Web service protocols enable a clear distinction between the content and the transport mechanism. An XML encoded request or response is encapsulated in a SOAP (Simple Object Access Protocol - http://www.w3.org/TR/SOAP) message and delivered using HTTP (Hyper-text Transport Protocol) or some other mechanism. The endpoints and transport mechanisms to these services are defined within WSDL (Web Services Definition Language - http://www.w3.org/TR/wsdl).

This paper draws on the experiences of the web community in building the Semantic Web to illustrate the emergence of a 'Semantic Grid'. We describe the architectural requirements needed to initiate construction of a Semantic Grid through our experiences with the ICENI framework that is being developed at the London e-Science Centre at Imperial College. We also describe how we are extending ICENI to use a well defined XML derived protocol between its services which will allow us to interoperate with other infrastructures through the use of web services.

2 ICENI

The Imperial College e-Science Networked Infrastructure (ICENI) has been developed by the London e-Science Centre (http://www.lesc.ic.ac.uk) as a research platform to meet the needs of our local applied e-science community. It is being used to define the protocols necessary for:

- capturing user intent through the specification of execution policy and proxy authority (e.g. completion deadlines or the restriction of execution to a specific set of resources)
- application construction by characterising performance and behaviour on different resources and mapping to a specific resource
- resource characterisation by describing its capability and usage policy
- distributed collaborative visualisation and computational steering of applications from several locations

The initial implementation of the ICENI architecture is based upon Java and Jini. Jini provides a framework for distributed service discovery and joining that is fundamental to any Grid middleware. Existing services in the Jini community are notified of the arrival and departure of new services through a distributed event notification mechanism.

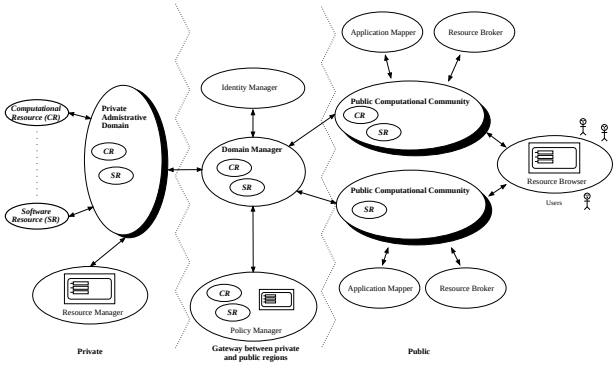

Figure 1: Building a public computational economy through Federated Resources.

The ICENI architecture is shown in Figure 1. It consists of private administrative domain and a public computational community, both of which are derived from Jini Lookup Servers. Resources within a real organisation are represented in the private administrative domain as Jini services. These resources are characterised by an XML encoded series of static attributes (e.g. operating system) and dynamic attributes (e.g. queue status). The 'resource manager' can configure these resources remotely (4).

The 'domain manager' publicises the capabilities of the resources within the private administrative domain into several public computational communities. The computational communities and the usage and access control polices for each resource in a particular computational community are defined within the 'policy manager'. Different resources can appear with different policies in different computational communities. Incoming requests to use a resource are delegated to the 'identity manager' for verification through an X.509 public key infrastructure.

The resource information stored within the computational community is access by the user through tools such as the 'resource browser' and is also available to higher level services such as the 'application mapper' and 'resource broker'. The application mapper uses information relating to the construction of the application and its performance on different resources to maximise its performance on a subset of these resources (5). The mapping of an application to a set of resources is described by an 'execution plan'. The resource broker uses the functionality within the application mapper to balance the execution of a particular application in the computational community with other applications that are already present to provide a globally optimal mapping of applications to resources.

The information relating to the application's construction and the user's wishes relating to execution are placed into the computational community as an 'application specification' and made available to the other services. As resources are allocated to the application and execution is initiated the application specification is updated to reflect this information and provides a contact point for future interaction with other services. This could include

rescheduling as 'better' resources become available or computational steering of the application (6).

3 Grid Programming Model

It is impossible to proscribe a Grid programming language on the diverse applications within the e-science community. Likewise, it is difficult to proscribe a programming model unless it provides the flexibility to encompass existing legacy applications and those compatible with current software engineering practices.

We are prototyping a component based model that allows legacy applications to be encapsulated as a single component with defined interfaces (7; 8). Our model allows a component's interface to be matched to any number of valid implementations. The decoupling between a components interface and implementation allows the component to be deployed on the 'best' currently available execution platforms within a distributed Grid environment (9). For example, each component may have several implementations such as different algorithms (e.g. iterative or direct solvers) for different platforms (e.g. Solaris or Linux) for different architectures (e.g. serial or parallel). We describe the component interface, its implementations and capabilities through CXML - an XML schema for Component applications (10)

More generally, we define an application as a network of linked components where we are able to describe the frequency and data volume of their interactions. By combining our knowledge of these interactions with information as to how a component's implementation will behave on a particular platform, with data relating to the platforms current state, we are able to optimise its performance within a particular policy (5). The decoupling between an interface and its implementation also allows us, by storing persistent data outside of the component, to migrate an application's components to other resources during execution.

In our model, a user submits a job (as an application specification) to the 'Grid' by placing a CXML description of the application network and the user requirements into the public computational community. The application mapper matches the components used in the application to the implementations that exist within the resources in the computational community. The application network is instantiated on distributed resources with the best component implementations to maximise a user defined criteria. The effective deployment of the application across distributed resources is enabled by the rich metadata relating to the resources, application structure and implementations available to the higher-level services. A component expects to be deployed into a local 'Grid container' that provides access to a minimal set of basic services. The detailed definition of these container services is currently under investigation.

4 Grid Services

Our experiences with ICENI and other infrastructures, such as Globus, have demonstrated the need for a minimum set of services to support e-science within computational Grids. We characterise these services into two groups: low-level services that interface directly to the underlying Grid fabric or provide essential services and higher-level services utilising these lower-level services. All services are registered with a Registry Service in the local computational community that exchanges information with its peers in other organisations.

4.1 Registry Service

The Registry Service (like the Jini Lookup Service) provides a 'known' point for information exchange between clients and other services. The local private registry service within an organisation federates its internal services into a community wide infrastructure through the actions of the domain manager. The domain manager 'pushes' information relating to capability and usage policies of the local community's services to (potentially) several public community wide Registry Services within different virtual organisations.

The domain managers and community wide Registry Service can be viewed as part of a peer-to-peer network. We intend to use this structure to propagate the services within one community to other computational communities. Propagation continues while there is an intersection between the acceptable useage policies of services from the remote domain with the local user community. This approach should ensure that all users allowed to use a resource will have access to it providing there is at least an indirect link between the two communities. Modelling of this infromation structure and an examination of the service propagation distance is underway.

4.2 Low-level Services

The local private Registry Service will contain a number of low-level services:

- Service Register – provides a mechanism for service registration, discovery and instantiation
- Authentication – verification of the user or host using X.509 certificates
- Authorisation – expression of a resources' usage and access control policy
- Execution Resources – invoke a deployed component on a resource
- Software Resources – an index of the locally deployed software components

- Storage Resources – a representation of the stored files and disk space including temporary storage space
- Data Resources – encapsulation of stored structured data such as virtual collections of flat files or relational databases
- Networking Resources – a representation of networking that can reserve networking bandwidth to meet an application's quality of service (QoS) requirements
- Messaging Service – reliable and secure delivery of messages to other services
- Accounting – recording of resource utilisation against (potentially) defined quotas

All resources have to be associated with an authorisation and authentication service before they can be used. The authorisation service verifies the ability of a user to use a resource before the service itself is invoked. It should be noted that not all resources need to be present within a single local directory service as a remote reference to the service may be given.

4.3 Higher-level Services

The higher-level services include:

- Data Mover – a service that finds the 'best' mechanism to move a file from one storage resource to another.
- Workflow Processor – manages the execution of an application specification consisting of one or more components defined using CXML
- Component Repository Resource – a repository of deployable Grid components
- Application Mapper – to find the optimal distribution of an application over a set of resources
- Resource Broker – to find the most 'cost-effective' use of resources for a particular application inheritance
- Bank Accounting – the operation of a computational economy requires a secure 'currency' abstraction to allow the operation of banks and user accounts

Much of the functionality provided by these services is already under development by ourselves (5; 9) or other groups. The low-level execution and authentication infrastructure already exists within the Globus toolkit. Resource brokers have been developed to support computational economics through projects such as Nimrod/G (11). However, protocols within experimental services are generally poorly documented inhibiting third party implementations.

To improve the transparency between the clients and service abstractions within ICENI all interactions are encapsulated within XML encoded messages. Our current XML schemas are being expanded to form a set of integrated protocol for interaction with other services. We see these protocols as being standardised through the relevant working and research groups within the Global Grid Forum.

5 Web Services within a Grid Environment

Future Grid services, like the networking and file transfer (FTP) protocols in use today, will be accessed through protocols with community wide acceptance. To enable interoperability between different implementations it is important that these protocols be open, self-documenting and extensible whenever possible to allow innovation from within the community. The web service protocols being developed within the W3C and other industry bodies provide a defined structure that encapsulates these requirements.

Current web services are defiend through multiple protocols. An XML syntax SOAP defines an envelope structure with a defined header and body containing the user's message in XML. WSDL specifies the 'endpoints' and delivery mechanism for a particular SOAP message. WSDL defined services (and other meta-data) may be registered within a community through the UDDI (Universal Description, Discovery and Integration - http://www.uddi.org) schema. Other organisations are able to search the meta-data within a UDDI repository to find the desired services.

This mechanism for service registration, discovery and usage using web protocols is similar to the use of MDS within Globus and the Jini Lookup Server in ICENI. A current concern with a UDDI repository of web services is the apparent assumption of service stability. Both the Jini and MDS mechanisms recognise that the Grid is a dynamic environment and that services will appear, disappear and possibly reappear. The MDS architecture specifies the frequency of updates and the duration of their validity. Jini services are made available for a period defined in its 'lease'. A lease may be renewed while the appearance and disappearance of Jini services can be propagated to other services through an event model.

It is not clear if protocols such as UDDI are able to reflect the dynamic composition of the Grid. Additional house-keeping processes may be needed to ensure that the UDDI registry contains only active services. Our approach is to develop a hybrid architecture that retains the flexibility of the Jini service model while ensuring compatibility with othe web services infrastructure.

The existing client and server interaction through XML encoded messages within ICENI is retained along with

the Jini service discovery mechanisms. The ICENI services are made available to clients using a web service protocol through a proxy embedded within an application server. See Figure 2. Both the internal ICENI and SOAP encoded web services messages use the same XML schema to describe the message content.

6 The Semantic Grid

The Grid middleware now being developed to support e-Science within the UK and elsewhere is evolving into a service-oriented architecture built upon standard web protocols such as XML, SOAP, WSDL and UDDI. While developing protocols within such a framework promotes interoperability there is no certainty that these protocols will always be understood. It is here, we feel, that a Semantic Grid will start to develop as services become capable of discovery and self-organisation.

Tim Berners-Lee defines the Semantic Web as 'an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.' (12). Likewise, the Semantic Grid can be described as an 'extension of the current Grid in which information and services are given well-defined meaning, better enabling computers and people to work in cooperation'. In such an environment it is essential that information relating to the needs of the user and their applications, and the resource providers and their networking, storage and computational resources all have easily discovered and defined meaning that can be used by higher-level services to effectively exploit the Grid.

The development of these service-oriented ontologies will enable compatible services to autonomously build the large complex distributed computing environment that constitutes the Grid. Fundamental to this goal is the expression of Grid services through well defined protocols. From these protocols we are able to understand the capabilities of the service and, potentially, reason as to how it can interact with other services to meet the needs of the users. This reasoning can take place at several levels, from service composition and application assembly to the representation and exploitation of knowledge generated through data and computation services (13). The mechanisms for exploiting the service and data ontologies within the Semantic Grid may range from simple schedulers, application mappers and resource brokers to sophisticated, autonomous, mobile and intelligent agents.

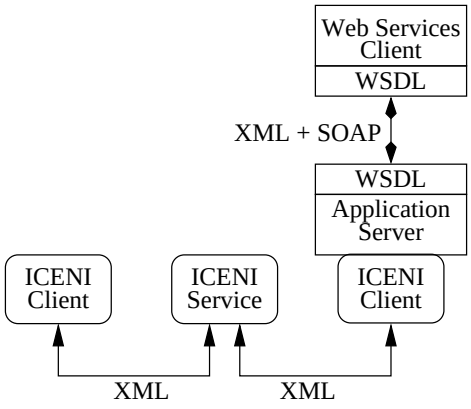

Figure 2: Extending ICENI to support interaction with web services.

7 Conclusions

Computational Grids are demonstrating a convergence of many existing areas of computer science research. The high performance computing community are developing new algorithms to support heterogeneous wide area computation between different supercomputers. Applications are having to deploy to new resources expecting only the services available within a 'standard container' environment. Users and applications are discovering distributed resources and services maintained through a network of peer-to-peer directory services. The autonomous discovery and assembly of a Grid environment from the available services is reducing the complexity involved in constructing a complex software environment while improving robustness through multiple services.

We consider the movement towards a Semantic Grid (at both the service and knowledge layers) as essential in simplifying the effective utilisation of sophisticated distributed services. The description of these services (using existing web-service protocols) will enable their intelligent composition and exploitation with minimal human interaction. The transparent and optimal delivery of sophisticated computational and data services to the applied science community will be key to the successful adoption of e-science.

We have shown how ICENI will continue to use Jini as a registration and service discovery mechanism while exposing its functionality within a web services framework to promote interoperability with other infrastructures. The XML schemas governing the interaction between different ICENI services will continue to be developed and encapsulated within SOAP for use by the web services interface. Services within our computational communities will be propagated to other organisations through a peer-to-peer mechanism.

Acknowledgements

Steven Newhouse gratefully acknowledge the formal and informal discussions that have taken place within the Architectural Task Force of the UK e-Science Core Programme which have helped clarify some if the issues discussed within this paper.

References

[1] I. Foster and C. Kesselman, editors. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, July 1998.

[2] I. Foster and C. Kesselman. The Globus Project: A Status Report. In *Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop*, pages 4–18, 1998.

[3] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. In *Intl. J. Supercomputer Applications*, 2001.

[4] N. Furmento, A. Mayer, S. McGough, S. Newhouse, and J. Darlington. Building Computational Communities for Federated Resources. Euro-Par 2001.

[5] N. Furmento, A. Mayer, S. McGough, S. Newhouse, and J. Darlington. Optimisation of Component-based Applications within a Grid Environment. SuperComputing 2001.

[6] James K. Stanton. Enabling Computational Steering through Visualization in a Distributed Environment. Master's thesis, Imperial College, Department of Computing, 2001.

[7] S. Newhouse, A. Mayer, and J. Darlington. A Software Architecture for HPC Grid Applications. In *Euro-Par 2000*, volume 1900 of *LNCS*, pages 686–689, September 2000.

[8] N. Furmento, A. Mayer, S. McGough, S. Newhouse, and J. Darlington. A Component Framework for HPC Applications. Euro-Par 2001.

[9] N. Furmento, A. Mayer, S. McGough, S. Newhouse, T. Field, and J. Darlington. An Integrated Grid Environment for Component Applications. 2nd International Workshop on Grid Computing 2001.

[10] Anthony E. Mayer. *Composite Construction of High Performance Scientific Applications*. PhD thesis, Department of Computing, Imperial College, 2001.

[11] R. Buyya, D. Abramson, and J. Giddy. Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid. In *The 4th International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2000)*. IEEE Computer Society Press, USA, 2000.

[12] W3C. Semantic Web. `http://www.w3.org/2001/sw/`.

[13] D. De Roure, N. Jennings, and N. Shadbolt. Research Agenda for the Semantic Grid: A Future e-Science Infrastructure. Technical report, EPSRC/DTI Core e-Science Programme, December 2001.