# UNICORE/w3*

R. Menday and B. Hagemeier

Central Institute for Applied Mathematics,
Forschungszentrum Jülich, D-52425 Jülich, Germany
r.menday@fz-juelich.de

**Abstract.** Drawing on the core values of UNICORE - "seamless, intuitive and secure access" - in this paper we propose building the next generation of UNICORE by closely aligning the Grid and the Web, namely by using the Web as the homogenizing middleware layer for the Grid. The RESTful use of HTTP coupled with a unified, RDF-based model, results in a loosely-coupled, global scale architecture. Therefore the Grid as a very rich source of information contributes to the Semantic Web of data. This is the foundation for the strong focus on usability in UNICORE/w3. Navigation of the linked resources, filtering, powerful searching functionality, the annotation and sharing of resources and monitoring using Web syndication techniques are some of the features proposed.

## 1 Introduction

Key characteristics of the Web are that it is simple to understand and use, deeply integrated in the user's desktop and working practices, navigatable through linking, loosely coupled and has unparalleled support in terms of tools and support. The initial Web can be likened to a global document repository, the future evolution is towards the Web is a single global database. Human readability will be augmented with data published in a form that can be usefully processed by machines.

On the evolutionary path towards the Semantic Web [13], so-called Web 2.0 added a new dimension in the richness and interactivity of the experience for the user. It added customised content, and the notion that other users are 'out there' consuming the same content. Importantly, we have also seen progress on the machine readable Web characterised by Web sites offering 'web apis' explicitly designed for direct consumption by machines. This has enabled another feature found in Web 2.0 applications - the 'mash-up'. This is the combining and re-use of data published on the Web, to build new applications. This is often based on XML and JSON, so whilst good results can be achieved, e.g. Google Maps mash-ups, the free joining of data from sources distributed across the Internet is not fully realised. This can only really happen when a unified model for describing the data in the Web is in place. With the maturing of the Semantic Web we

* This work is partially funded through the European A-WARE project under grant FP6-2005-IST-034545.

L. Bougé et al. (Eds.): Euro-Par 2007 Workshops, LNCS 4854, pp. 72–81, 2008.

are starting to see this happening. A result of this can be seen in the W3C Tabulator[11], an RDF browser allowing the user to explore the linked 'Web of data'. Furthermore, LinkingOpenData[6] is an interesting initiative targeting the 'bootstrapping' of the Semantic Web with the goal of making "... various open data sources available on the Web as RDF and to set RDF links between data items from different data sources". In future we will see more re-use, re-publishing, merging and integration of information over the Web. This is the Web as the global-scope, homogeneous layer providing a role similar to that of middleware for machine consumers of the data. Furthermore, as the Web evolves towards this Web of data, so to do the opportunities for using the Web as a middleware for Grid computing.

**Why the Grid and the World Wide Web ?**

Grid Computing is the "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organisations" [20]. It is fair to say that the initial target audience was the scientific community. It is interesting to compare with the initial impetus driving Tim Berners-Lee's vision of the World Wide Web. The Web was initially conceived due to a frustration preventing scientists from effectively collaborating and sharing scientific results[16]. Between the Web and the Grid there seems to be some shared purpose, although it is surprising that none of the main Grid middleware platforms have attempted to integrate, to a deeper extent, Grid middleware with the infrastructure of the Web.

Our reservations regarding the suitability of Web services for building a single Global grid, leads us to this paper where we present our vision for a Web-based Grid, as a starting point for discussion and future activity. We believe that building UNICORE as a Web application addresses the shortcomings of the previous versions of UNICORE, and provides new opportunities for our users.

Thus this paper makes the case for the single Grid - one which is realised as part of the Web. The vision is that every supercomputer (or any other computing resource) has a projection of it and its resources published and accessible over the Web. HTTP agents interact with these resources. The concept of navigating links (through clicking) to discover related information is highly intuitive for any computer user. With a resource-oriented approach we directly identify with a HTTP URI all our resources, for example, a File, Job, Storage, Reservation, User, etc. Given such a situation information on an executing job or a particular data artifact, for example, can be shared between collaborators or bookmarked. This is the participatory Grid where scientists are able to use the Grid infrastructure for the fine-grained sharing of data and jobs.

We cover many topics in the paper, and due to space restriction many technical details have had to be omitted. The paper continues as follows. In Section 2 we review some background to this work, in particular looking back at relevant architectural approaches and philosophy in the evolution of the UNICORE grid system since 1997. Section 3 covers some of the technologies which make the Web ideal for Grid computing. Section 4 describes the functionality and architecture of UNICORE/w3 - first by describing how the Grid of resources become part of

the Web, and then by describing some possibilities for higher-level services which provide aggregation points exploiting these distributed resources. We review relevant work in Section 5, and offer a summary and outlook to the future in Section 6.

## 2 Review

The original goal of UNICORE was to develop production quality software for accessing the distributed resources of high performance computing centres, and largely this still drives UNICORE development today[1]. The reader is referred to [18] and [25] for good overviews. UNICORE/w3 draws on experience with all versions of UNICORE. Relevant architectural concepts are summarised below.

**Abstract Job Object.** The central architectural concept in pre-Web Services UNICORE (versions 1-5) is the Abstract Job Object (AJO). The AJO is the unified model of the distributed computing world. In this model computer resource descriptions and requests use the same model. It is realised as Java objects and uses the Java serialisation format as wire format[2]. UNICORE recognised early on that the Grid can be best modelled as a graph, and the AJO reflects this. The goal of seamless computing is addressed by the abstract-ness of the AJO. Incarnation in the AJO processing chain maps the abstractly expressed user request into an executable form for the targeted system.

The document-centric design of the AJO is almost exclusively procedural. Whilst this is convenient for modelling Jobs running on a batch system, it is interesting to note that the AJO does not contain a object representing a File on the Grid (an XFile is used to structure the responses to ListDirectory and FileCheck, but not as a first class entity of the model) making it difficult to capture at a top-level abstraction the notion that a Storage contains a set of Files[3]. This is an example of one of the shortcomings of the AJO that is addressed in current modelling.

Another consequence of the procedural model is the unconstrained nature of the 'operations' (or processes) encoded into the AJO. Changing operation names or subtle differences in how these documents are conveyed, means that a server upgrade almost always prompted corresponding client updates. This is satisfactorily in a tightly-controlled environment, but not in a Global setting where for example multiple versions of the same software should be expected to be in use. Thus, whilst UNICORE achieved vertical integration with the AJO, this was at the expense of loose-coupling.

[1] UNICORE 6 was released mid-2007 and is based on Web services technologies.

[2] A recognised issue with the implementation of the AJO model is the strong dependency on Java, and as a result of this the model is rather closed and difficult to extend. However, this is more of a simple rendering issue not fault with the model as such.

[3] We note that the AJO Portfolio object captures a subset of this requirement

**Web-Services and XML.** UNICORE 6 builds on top of a Web services based foundation developed in the UniGrids[2] project. Most other popular Grid middleware, for example Globus 4[3], has also migrated to a Web services based infrastructure in recent years. UNICORE 6 boasts excellent performance enhancements over previous UNICORE versions, and uses XML to overcome the Java serialisation format limitation. However, Web services were supposed to address the issues of tight-coupling in Grid architectures, bringing Grid infrastructure to the Internet scale. We claim that this has failed.

Much has been made of WSRF providing support for stateful Web services. Whilst the name of the Web Service Resource Framework (WSRF) seems to indicate a resource-centric approach, the resource in question is obscured by the service managing each WS-Resource. One re-occurring criticism of WSRF is that it is a distributed object system (perhaps more realistically a distributed object facade), and in essence it is. The state is hidden behind the service interface. Manipulation of the state of the WS-Resource is done through service-specific set of operations (although WSRF attempts to standardise some common state retrieval operations), and there is no explicit concept of navigatable state. In a Web services based SOA whilst there is a good loose-coupling of implementation (primarily through the use of XML) such that unlike the Java based AJO, in theory client and server do not have to be written in the same language. However, the strictly-defined and specific nature of the interface results in an early binding and consequent tight-coupling between client and server. A Service-oriented approach also makes the service the primary entity, and not the resources it manages. This has major implications regarding caching.

Whilst an improvement on the binary format of serialised Java objects, one can question semantic sparseness of XML as a document format. What emerges are XML 'islands' of information which make powerful search, query and joining of data difficult.

Therefore whilst the AJO does not make enough of its natural ability to describe a graph, in principal it is in a better position to model the richness and complexity of the Grid. We note that the graph-oriented modelling of the Grid evolved further in the UniGrids project [12], [23] and we take these ideas further in UNICORE/w3.

## 3 The World Wide Web

The standards driving the Web are absolutely established and stable. The same can be said for the Web infrastructure - HTTP servers, routers, firewalls, etc. This section describes some key characteristics and features of a Web oriented architecture.

**Single Resource, Multiple Representations.** As described in [17], we make the distinction between two kinds of resources: Information resources and non-information resources. Non-information resources are 'real world objects' which exist outside of the Web, but are identified by HTTP URIs. Therefore, rather

than being directly de-referencable, following the advice of [24] we can use the Web as a lookup mechanism to find more information about these resources. We use the '303 URIs' method where a HTTP 303 redirects a HTTP request for a non-information resource, using content negotiation to redirect to the relevant representation of the information resource (for example, the RDF document describing the resource, or a human readable HTML representation). UNICORE/w3 should support HTML, XML, JSON and RDF representations as a minimal set.

**REST and ATOM.** Millions of Web sites are in a constant state of evolution but this happens without forcing the upgrade of each users' Web browser. Client and server are freer to evolve independently. The uniform interface constraint of REST[19] limits the set of allowable operations on resources, and REST advances this characteristic as one reason for the loose coupling of the Web. For HTTP, the essential operations are GET, PUT, POST and DELETE. As an example, taking the `getResourceDescription` and `getJobs` AJOs and migrating to one possible RESTful scheme, these become just GET on a VSite and VSite Jobs collection resources respectively. REST proposes a resource-oriented approach, where the state of a users interaction is reflected by the HTTP URIs they are currently 'talking to'. The state of an application is published as a navigatable set of resources, and through the navigation of links the state of the application is transferred back to the user as a URI. The agent can de-reference this URI to discover a new set of navigatable URIs. Furthermore, a resource-oriented approach is actually something very useful and beneficial from a user's perspective too as it is likely to achieve a desirable quality of 'bookmark-ability', and in the Grid domain, resources which a user might want to bookmark are very easy to identify. Furthermore, ATOM and the ATOM Publishing Protocol[1] offer a convenient payload for distributing Web content as a chronological sequence of items and for creating and updating Web resources. Both are milestone technologies and play an important role in this work.

**The Semantic Web.** By itself a Web based system based on HTTP and its URIs, with HTML, XML, JSON and ATOM representations is a significant advancement for UNICORE. Going further, the Grid is part of the Web, and the Web is a graph. An ability to describe this graph is extremely powerful as it brings the data together into a single unified model in a manner far more flexible than XML or Database structures. RDF is a language for representing information about resources, in the form of statements about these resources, including the type of a particular resource and the nature of the relationship with other resources. For example, in a supercomputer, a File is related to a particular Storage where it is held, and the nature of this relationship can be explicitly stated in a computer processable manner. Ontology (expressed using RDFS/OWL) is used to define the vocabulary for the resources, their types and the relationships between them. This can be used to infer non-explicitly stated relationships between the resources in the system, and the Semantic Web query language SPARQL[10] can be used to query this information.

## 4 Grid Computing with UNICORE/w3

UNICORE is the "Uniform Interface to Computing Resources" and it summarises its character very well. UNICORE/w3 software acts a gateway to the business and scientific resources of a single site, projecting representations of each resource onto the Web for those authorised to view and manipulate them. HTTP agents interacting at this interface perform the functionalities traditionally associated with Grid computing, namely resource management, information services and data management. All of which are dependent on a common security infrastructure. We explore each briefly in the following sections. Then we take a look at what is often referred to as 'higher-level services' - the Google and Yahoo Pipes of UNICORE/w3.

**Information.** Information publishing at a single resource (VSite in UNICORE terminology) is the publishing of information about the Grid resources at this particular site - Jobs, Storages, Files, etc. We use a lightweight ontology (inspired to some extent by the UniGrids ontology [12]) expressed using OWL to define the vocabulary for this description. We note that an attractive feature of RDF is that it encourages a data-first approach, whereby the collection of RDF statements can use vocabulary from a number of ontologies, and one is not constrained by defining a schema first and populating according only to that particular schema. In this sense the information model for UNICORE/w3 is highly flexible and extensible.

**Resource Management.** In the simplest case, resource management at a single VSite is achieved by POSTing activity descriptions to a Job collection resource using the 'process this' semantic associated with HTTP POST, resulting in the creation of new resources. The execution is managed with the various stages in the execution modelled as subordinate resources. An ATOM feed publishes the changes to the Job resource as it evolves. Complete Job removal uses HTTP DELETE on the Job resource which also deletes the subordinate resources. A possible enhancement is through publishing Application resources reflecting applications discovered on the target system (application software resources in UNICORE terminology). Application resources publish template request documents, bootstrapping resource negotiation.

**Data Management.** UNICORE/w3 enables a projection onto the Web of files and directories in storage systems. For UNICORE/w3 each entity is individually identified with its own HTTP URI. By default files in a storage are only accessible by the owner of the file. Basic information (modified times, size) and data management functionality (copy, move, delete) should be offered. Furthermore, it should be possible to instruct one site to initiate a transfer of a file to another remote storage. Following a mechanism similar to UNIX file permissions, we can allow the owner of a single file resource to make it accessible to a group of authenticated users. Furthermore, a file could be published as freely readable by everyone. This is similar to the publishing of Google Docs[4], which by default are private, but on request can be made viewable by all. Finally, the actual

contents of a file should of course be retrievable. HTTP transfer is supported, but also a user may wish to take advantage of other transfer mechanisms, and XMPP and Bittorrent are potential candidates here.

**Security.** HTTP requests for a particular resource are subject to authorisation checking based on the URI of the resource and the identity of the authenticator. Controlling who is able to view/modify resources is important. For example, usage information from a group of users should only be accessible by administrators. Organising around resources makes an implementation of a robust and fine-grained authorisation policy cleaner. Authentication in such architecture also needs to be addressed robustly. Due to space constraints we have not analysed in depth the issues, but at the global scale, there is a growing need for a Web-native identity solution. We foresee leveraging the open, rapidly emerging OpenID standard. With OpenID, users identify themselves with a URI, aligning nicely with the RESTful, resource-centric design.

### Higher-Level Functionality

For many Web users a search engine, such as Google, provides the 'jump-in' point to the Web. We see specialised search engines - indexing a subset of the Web related to a particular user's known Grid resources - offering a similar functionality for the Grid of resources of UNICORE/w3. Leveraging the rich graph of information and the search capability, UNICORE/w3 users will be able to view many presentations of their Grid of resources in ways most useful to them. These aggregation services, whilst accessing information from the underlying resources, can also assert additional RDF statements regarding these resources. Of course, such annotation carry less provenance, but can be used in many interesting ways. For example, the search engine can allow the user to tag particular resources in the underlying Grid of resources.

**Searching and Tagging.** Information from the Grid, once published as RDF, can be queried and joined with any information, including information from other resources. SPARQL[10] is the standard RDF query language. Provided with an RDF description of resources, SPARQL can be used to express queries across them. This may be used for a multitude of purposes - for resource selection, querying for support of a particular application, mining for trends, etc. As described above, the user is able to 'tag' particular resources with a free-form string. Queries can then take the tagging attributes on a particular resource also into consideration. Examples of typical queries include,

- find all Sites, current status and other useful monitoring information
- find the total number of Jobs executed at each site over the past 30 days
- find all Jobs tagged 'hot', which have appended files in the last 30 minutes
- find Files in directory with a specified file name pattern.

Once a new SPARQL query has been POSTed a new resource is created for that particular query. This can be re-run at any point by GETting that resource.

The results of a SPARQL SELECT query results in a table of information, each column header for each variable in the query and the results of the query in each row. Moreover an ATOM feed is associated with each query. Over time, changes in the results of the query are published in the feed allowing changes in the SPARQL result to be identified. We believe this to be a unique and extremely useful merging of SPARQL and Web syndication.

**Workflow.** Other services may offer to manage long-running executions, such as the coordinated usage of multiple computational resources. Following the REST principle that 'hypermedia is the engine of all application state' we have the foundation of a interactive and modifiable/dynamic workflow model. Workflow structures as RDF graphs are stored as additional statements related to the underlying resources. One can view a Web-based workflow as a collection of Web resources, and the state of the users interaction is captured by these resources, where every stage in the workflow is a resource in the graph, and can be de-referenced. This is a very powerful concept for when it comes to monitoring, tracking and re-writing.

## 5 Related Work

In this paper we propose using the Web and HTTP as a end-to-end infrastructure for the Grid, and we therefore note differences between this and many projects building portals over existing Grid infrastructures, for example, GridSphere[5]. The Semantic Grid[7] has made many explorations in the area of Semantic technologies applied to Grids. Much work from this working group is connected with adding Semantic support to Web services infrastructure to build Semantic Web Services, although recent activity at the OGF[8] is exploring synergies between Web 2.0 and Grid including a workshop at OGF 21, where we see our work as highly relevant. Of course a number of others have done some extremely interesting work in the application of Semantic and Semantic Web technologies to Grids. In 2001, [21] evaluated the applicability of using RDF schema as a vocabulary for Grid resources, and in [26] the authors propose using Semantic Web technologies for resource selection. In [14] a delegation ontology is derived and SPARQL is used to evaluate policy.

A proposal for layering SPARQL functionality and a HTTP interface over existing Grids appeared in [22], and this can be seen as forerunner to this work. Regarding other work in the direct UNICORE community we are not aware of any related work which builds upon the core UNICORE values, wholly recommending building on a RESTful Web foundation and using the Semantic Web technologies to add rich structure.

## 6 Outlook and Summary

In this paper we have outlined our vision for a Web-based future for UNICORE. We can summarise a number of possibilities for further exploration. The use of

OpenID[9] as an identity solution looks very appealing, but needs further analysis. For example, provision for delegated authorisation could possibly leverage the additional functionality offered by OpenID 2.0. For the searching functionality provided by the information services we would like to investigate other RDF query languages (of which there are many). We would also like to iterate further on the core UNICORE/w3 ontology, re-using existing ontologies where appropriate. A negotiation process to guarantee a particular quality of service (QoS) is usually managed using WS-Agreement[15] in SOA based Grids. We note that the Web naturally provides much of the capabilities necessary for resource negotiation - the notion of the Web as protocol state machine, and a template repository and would also like to examine this area in more detail. Finally, the nuts-and-bolts of taking the UNICORE/w3 concepts and making a production system from the current prototype adds a number of other topics for further discussion.

From the notion of clicking to navigate, to receiving a HTTP URI in an email, the Web is an extremely familiar environment for most people. Using the Web the middleware layer for the Grid is provided by HTTP. The constrained interface of HTTP, together with a RESTful design of the resources, offers much in terms of genuine loose-coupling between client and server, cacheability, scalability, stability and accessibility. In addition using the Semantic Web provides a basis the for flexible and powerful higher-level services going well beyond todays state of the art. We offer searching and the notion of the participatory Grid where scientists are empowered to easily share results computed on the Grid.

It is worth noting that an early prototype already demonstrates a number of these concepts. It is the strong conviction of the authors that the Web (more precisely the Semantic Web) will subsume the Grid as we know it today. Building on the core values of UNICORE - "seamless, secure, and intuitive access" - this paper is intended as a sincere exploration of the further opportunities which will come with a full migration to the Web.

## References

1. ATOM, http://tools.ietf.org/html/rfc4287
2. European UniGrids Project, http://www.unigrids.org
3. Globus, http://www.globus.org
4. Google, http://www.google.com
5. GridSphere, http://www.gridsphere.org/
6. Linking Open Data, http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpen Data
7. OGF Semantic Grid, http://www.semanticgrid.org/
8. Open Grid Forum, http://www.ogf.org/
9. OpenID, http://openid.net/
10. Sparql, http://www.w3.org/TR/rdf-sparql-query/
11. The Tabulator, http://www.w3.org/2005/ajar/tab
12. UniGrids Ontology, http://www.unigrids.org/ontology.html
13. W3C Semantic Web, http://www.w3.org/2001/sw/

14. Ahsant, M., Basney, J., Mulmo, O., Lee, A., Johnsson, L.: Toward An On-demand Restricted Delegation Mechanism for Grids. In: Proceedings of the 7th IEEE/ACM International Conference on Grid Computing, September 28th-29th, Barcelona (2006)
15. Andrieux, A.: et al. Web Services Agreement Specification (2007)
16. Berners-Lee, T.: Weaving the Web (Hardcover). Texere Publishing Ltd. (November 1999)
17. Bizer, C., Cyganiak, R., Heath, T.: How to Publish Linked Data on the Web, `http://sites.wiwiss.fu-berlin.de/suhl/bizer/pub/LinkedDataTutorial/`
18. Erwin, D. (ed.): UNICORE Plus Final Report – Uniform Interface to Computing Resources. UNICORE Forum e.V. (2003), ISBN 3-00-011592-7
19. Fielding, R.T.: Architectural styles and the design of network-based software architectures. PhD thesis, Chair-Richard N. Taylor (2000)
20. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: Enabling scalable virtual organizations. In: Sakellariou, R., Keane, J.A., Gurd, J.R., Freeman, L. (eds.) Euro-Par 2001. LNCS, vol. 2150, Springer, Heidelberg (2001)
21. Gunter, D., Jackson, K.: The applicability of rdf-schema as a syntax for describing grid resource metadata (2001)
22. Menday, R., Streit, A.: SPAQRLing UNICORE. In: Proc. of the 6rd Cracow Grid Workshop (CGW 2006) (2006)
23. Parkin, M., van den Burghe, S., Corcho, O., Snelling, D., Brooke, J.: The Knowledge of the Grid: A Grid Ontology. In: Proc. of the 6rd Cracow Grid Workshop (CGW 2006), October 15–18 (2006)
24. Sauermann, L., Cyganiak, R., Vlkel, M.: Cool uris for the semantic web. DFKI Technical Memo TM-07-01 (2007)
25. Streit, A., Erwin, D., Lippert, T., Mallmann, D., Menday, R., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Wieder, P.: Unicore - From Project Results to Production Grids. In: Grandinetti, L. (ed.) Grid Computing and New Frontiers of High Performance Processing, Elsevier, Amsterdam (2005)
26. Tangmunarunkit, H., Decker, S., Kesselman, C.: Ontology-based resource matching in the grid—the grid meets the semantic web (2003)