Web Service Grids: An Evolutionary Approach

Malcolm Atkinson[1], David DeRoure[2], Alistair Dunlop[2], Geoffrey Fox[3], Peter Henderson[2], Tony Hey[4], Norman Paton[5], Steven Newhouse[2], Savas Parastatidis[6], Anne Trefethen[4], Paul Watson[6], and Jim Webber[6].

Abstract

The UK e-Science Programme is a £250M, 5 year initiative which has funded over 100 projects. These application-led projects are under-pinned by an emerging set of core middleware services that allow the coordinated, collaborative use of distributed resources. This set of middleware services runs on top of the research network and beneath the applications we call the 'Grid'. Grid middleware is currently in transition from pre-Web Service versions to a new version based on Web Services. Unfortunately, only a very basic set of Web Services embodied in the Web Services Interoperability proposal, WS-I, are agreed by most IT companies. IBM and others have submitted proposals for Web Services for Grids - the Web Services ResourceFramework and Web Services Notification specifications - to the OASIS organisation for standardisation. This process could take up to 12 months from March 2004 and the specifications are subject to debate and potentially significant changes.

Since several significant UK e-Science projects come to an end before the end of this process, the UK therefore needs to develop a strategy that will protect the UK's investment in Grid middleware by informing the Open Middleware Infrastructure Institute's (OMII) roadmap and UK middleware repository in Southampton. This paper sets out an evolutionary roadmap that will allow us to capture generic middleware components from projects in a form that will facilitate migration or interoperability with the emerging Grid Web Services standards and with on-going OGSA developments. In this paper we therefore define a set of Web Services specifications - that we call 'WS-I+' to reflect the fact that this is a larger set than currently accepted by WS-I – that we believe will enable us to achieve the twin goals of capturing these components and facilitating migration to future standards. We believe that the extra Web Services specifications we have included in WS-I+ are both helpful in building e-Science Grids and likely to be widely accepted.

Author affiliations:

1. National e-Science Centre, Universities of Edinburgh & Glasgow, 15 South College Street, Edinburgh, UK
2. OMII, University of Southampton, UK
3. Computer Science Department, Indiana University, US
4. EPSRC, Polaris House, Swindon, UK
5. Computing Department, University of Manchester, UK
6. School of Computing Science, University of Newcastle, UK

TAG revised version, 31st July 2004

1. Introduction and Motivation

1.1 The UK e-Science Programme

The £250M, 5-year, UK e-Science Programme has funded over a 100 separate e-Science projects, all of which involve one or more forms of distributed data, computation and collaboration [1]. Details of these UK e-Science projects may be found elsewhere (see for example http://www.nesc.ac.uk/projects). These projects are not all 'academic' in nature: around 80 companies are engaged with the UK e-Science programme. Underpinning all of these application-led projects are an emerging set of middleware services that enable the coordinated, collaborative use of distributed resources (e.g. computation, data sets, applications, etc.). This set of middleware services – determined by the application requirements – is what we call 'Grid' middleware infrastructure. The majority of the UK projects have chosen to build their distributed middleware infrastructure using Web Services technologies; a minority use Condor and/or the pre Web Services releases of the Globus Toolkit GT2; some projects have been experimenting with early releases of GT3, an implementation of the Open Grid Services Infrastructure (OGSI) which uses Web Services technologies.

We are now about three years into the 5-year UK e-Science initiative and some of the early 'pilot' projects are nearing the end of their 3-year lifetime. The e-Science Core Programme has therefore established an Open Middleware Infrastructure Institute (OMII), based in the University of Southampton, to act as a repository for UK Grid middleware and as a centre of expertise in software engineering. The OMII remit is to identify potentially generic Grid middleware components from the UK project portfolio and either work with the project teams or use OMII software engineers to bring the quality of these components up to 'production' level. The OMII will then ensure that these components are well documented and are maintained in a suitable middleware repository. The OMII will undertake integrative testing of these UK middleware components for interoperability with components produced outside of the UK. Close collaboration with the US NMI GRIDS Center (http://www.grids-center.org) and other international middleware centres is envisaged. The resulting Grid middleware will be open source and be available for deployment by the UK National Grid Service and by other users. A key role for the OMII will be testing middleware components to ensure interoperability with open Grid and Web Services standards. We see open source reference implementations of the commercially agreed standards as very important both for the UK academic community and for SMEs wishing to develop commercial applications.

1.2 Grids and Web Services Standards

It is with the standards process that we have a short-term problem. At the March 2004 Meeting of the Global Grid Forum (GGF), IBM and the Globus Alliance, together with some other companies, presented a proposal for an evolution of OGSI based on a set of new Web Services specifications - WS-ResourceFramework (WSRF) and WS-Notification (WSN) [2]. Since this GGF meeting in Berlin, there has been the welcome news that these proposals have now been submitted to the OASIS standards organisation for development and standardisation. However, the OASIS standards process could take up to 12 months to come to a conclusion. Thus, complete, stable and robust implementations of WSRF and WSN cannot be expected until mid 2005. Although there is widespread agreement within the Grid community about the long-term wisdom of adopting a Service Oriented Architecture, using Web Services technologies as the basis for building a robust and stable Grid infrastructure, there is a short-term problem with such an approach. Web Services are still very much 'work in progress'. Unfortunately there are at present many proposed Web Services specifications but very few mature WS standards that have gained general acceptance and are supported with robust tooling.

To summarize, given that a significant number of the UK e-Science projects will finish *before* many of the proposed Web Service standards are ratified by OASIS, the UK and the OMII need to find a pragmatic, evolutionary way forward. This must enable us to capture generic middleware infrastructure components from projects in a form that will allow migration or interoperability with the Web Services standards that emerge and with the on-going OGSA developments within the OGSA-WG at GGF. By developing such an evolutionary roadmap, we hope to minimize the risk to the UK investment in e-Science caused by the slow pace of WS standards development. Most importantly, such a pragmatic approach will provide a mechanism by which we can protect the academic and industrial application community from the inevitable uncertainties and changes in these low-level standards. This problem – 'Grids and Web Services' - was the subject of a 'Town Meeting' for the UK e-Science community in April 2004: this paper summarizes the conclusions of that meeting. The purpose of this white paper is therefore to present a strategy for the UK e-science community to cope with the very fluid situation with respect to these low-level standards. The detailed implementation of this strategy will be described in future OMII White Papers. It should be emphasized that this is an evolutionary strategy to converge with the results of an ongoing standards process. We therefore envisage that this interim approach will only be needed for the next twelve months or so.

1.3 A UK e-Science Strategy

Our proposed strategy must address the following specific aspects of the UK e-Science programme. Firstly, there is a wide diversity of software platforms being used by UK e-Science projects. These include the Globus Toolkit versions 2 and 3 as well as the Apache Axis and .Net Web Services platforms. Some scientists are involved in several projects built on different platforms. Secondly, there is undoubtedly some re-invention of similar services in multiple projects. There are a number of core functionalities that several projects have developed from scratch in the absence of suitable off-the-shelf solutions. These include both registries and workflow enactment engines. Thirdly, there is limited sharing of software prototypes between projects. Moreover, most of our projects have little experience of developing software to a level whereby it could be deployed outside the community in which it was initially conceived.

Our strategy is summarised as follows:

1. We identify a set of Web Services specifications that can be safely adopted now (termed WS-I+) to build interoperable Web Service Grids.
2. The OMII will provide downloadable implementations of a set of interoperable, high-level services built using the specifications in WS-I+ and provide advice on how they can be used to build interoperable Grid applications. This advice will be encapsulated in a forthcoming 'Building Grids with WS-I+' OMII White Paper.
3. We intend that those providing deployed services to communities (e.g. for job scheduling, data access and file movement) will be able to utilise the services provided by the OMII, so making it straightforward for developers to incorporate those services into their applications.
4. For those projects building Grid applications using components from the Globus Toolkit, Condor, EDG/EGEE, SRB or from other sources, support for interoperation of these components with the OMII WS-I+ middleware releases will be provided when appropriate Web Services interfaces are available for these components. Wherever possible, agreements will reached with providers of such widely used components so that combined use is simplified.

The strategy is explicitly intended to be evolutionary - in that we need to plan for interoperability with, or migration to, Web Service and Grid standards emerging from OASIS, W3C and GGF. In particular, the reason for building on the 'WS-I+' set of specifications identified in this White Paper is to provide stable medium-term access to the core

functionalities required by a wide range of our e-Science applications. It is envisaged that the set of specifications in 'WS-I+' will expand over time as other specifications emerging from OASIS, W3C, GGF and other standards bodies gain acceptance. We hope that within twelve months we will have converged on a widely accepted set of standards for Web Service Grids.

In the meantime, these WS-I+ specifications will provide the UK e-Science community with a stable development environment available in multiple products on several platforms. It is therefore important that members of the UK e-Science community continue to contribute to the development of these standards and to conduct research and develop prototypes that will help to inform the development of future standards. Further, the set of services that will be initially provided by the OMII are deliberately narrower in scope than those proposed in the Open Grid Services Architecture (OGSA) of the GGF [2]. However, the OMII services will evolve over time to utilise new specifications that will be added into the WS-I+ set as they become standards with wide community support, adoption and tooling. Support and advice will also be offered to users in choosing when and how to make direct use of the new standards. Wherever possible, the OMII service interfaces will be kept stable across the introduction of new standards and underlying technologies, and interfaces will be flagged as deprecated substantially before the functionality ceases to be supported.

The plan of the paper is as follows. Section 2 introduces Service Oriented Architectures and Web Services and describes the general state of Web Services standards and proposals, while Section 3 defines the conservative subset – 'WS-I+' – that we recommend as being stable for those building services. Section 4 contains some conclusions.

2. Service Oriented Architectures, Web Services and Grid Applications

Before we start on a detailed discussion of Web Services we should first provide a clear definition of the concepts related to Web Services and Service Oriented Architectures.

2.1 Service Oriented Architectures (SOA)

The idea of service-orientation is not new [3, 4]. Distributed application developers have long deployed services as part of their infrastructure. For example, the set of services found in CORBA [5] is an example of the community's efforts to standardise on a number of services that provide the functionality needed to support loosely-coupled, distributed object-based applications.

After a significant period of time where object-orientation was the primary methodology for building software, the emergence of XML [6], XML Schema [7], SOAP [8], and WSDL [9] has refocused the development community's attention on service-orientation as a means to implement loosely-coupled distributed applications. Unfortunately, the term SOA has become overloaded. In the absence of a well-accepted definition of a service or SOA (some can be found in [10-12]) we define a service in a deliberately minimal fashion as follows:

> *A service is the logical manifestation of some physical or logical resources (like databases, programs, devices, humans, etc.) and/or some application logic that is exposed to the network;*

and

> *Service interaction is facilitated by message exchanges.*

The structure of a typical service is shown in Figure 1. It consists of resources (e.g. data, programs, devices, etc.), application logic, and a message-processing layer that deals with message exchanges. Messages arrive at the service and are acted on by the service logic, utilising the service's resources as required. Services may be of any scale: from a single operating system process to enterprise-wide business processes.

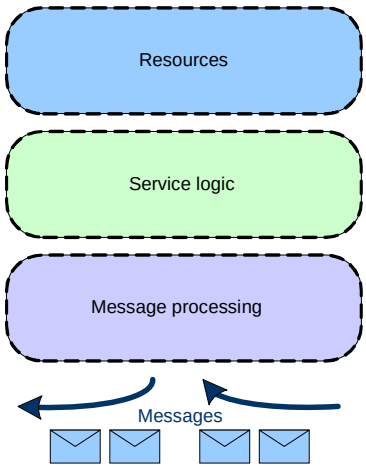

Figure 1. Anatomy of a service.

When building an application composed from such services, we believe that there are a number of design principles that should be followed [3]:

- **Boundaries are explicit:** The boundaries of a service are well defined when they are incorporated into a distributed application. Other services do not see the internal workings, implementation details, or resource representations of a service.
- **Services are autonomous:** Service implementations are developed and evolve independently from one another.
- **Services can be aggregated:** Services defining their interfaces and policy can be linked together into a larger composed Web service whose detailed composition need not be exposed to other services invoking the aggregate service.
- **Services share schema and contract, not classes:** In service-oriented architectures, no single set of abstractions (classes) spans an entire application. Services share schemas and contract that define the structure of the information that they exchange, not information about their underlying type systems.
- **Policies determine service compatibility:** Services interact with one another only after it has been determined – based on policy assertions – that they can meaningfully exchange information.

2.2 Web Services

Web Services technologies are an attempt to define the building blocks for building loosely-coupled, distributed applications, based on the SOA principles. As defined in [13], Web Services interact by exchanging messages in SOAP format while the contracts for the message exchanges that implement those interactions are described via WSDL interfaces.

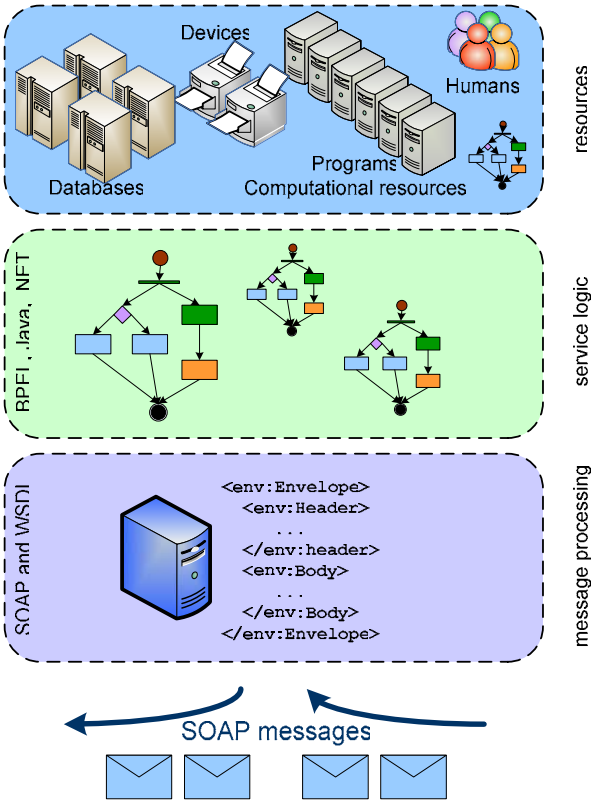

Figure 2 Anatomy of a Web Service

Figure 2 shows the logical structure of a Web Service. When a SOAP message arrives at a Web Service it is first handled by the service's message processing logic which transforms network level SOAP messages into something more tangible for applications to deal with (such as domain-specific objects). Once the message has been consumed, its contents are then processed by the application logic, making use of the resources available to the service. Typically some response is then generated which is fed back via one or more messages.

By encapsulating those internal resources within the service, and providing a layer of application logic between those resources and the consumers, the owners of the service are free to evolve its internal structure over time (for example to improve its performance or dependability), without making changes to the message exchange patterns that are used by existing service consumers. This encourages loose-coupling between consumers and service providers, which is important in inter-enterprise computing, as no one party is in complete control of all parts of the distributed application. However, loose-coupling does not mean that the functionality of applications is compromised, since the set of existing and emerging Web Services specifications will allow distributed application builders to model complex interactions between services.

Web Services specifications can be divided into two classes. Infrastructure specifications define generic aspects of services (or other specifications), e.g. WSDL, WS-Security and WSRF. High-level specifications define domain specific aspects of services, e.g. a data access and integration service specification. As indicated in section 2.1, policy also plays a key role in a service oriented architecture. While WSDL describes the functional characteristics of a Web Service - such as operations supported, messages sent and consumed - the non-functional requirements associated with service invocation are also a very important aspect of Web Services and service oriented architectures in general. WS-Policy [14] and WS-PolicyAttachment [15] describe a foundation policy framework within which the behaviours associated with a service - such as security, transactionality, reliability and so on - can be specified. Conceptually, WSDL and WS-Policy are peers in the Web Services stack.

By leveraging the developments in Web Services technologies, Grid architects can exploit the tools, documentation, educational materials, and experience from the Web Services community when building applications without having to create a parallel set of solutions. This allows the Grid community to concentrate on building the higher-level services that are specific to the Grid application domain while the responsibility for the underlying infrastructure is left to IT industry. The software vendors will work on standardising the WS technologies, developing production-quality tooling, achieving wide adoption, testing for the interoperability of the implementations of those standards, educating developers, etc.

2.3 Web Services Standards

Web Services standards are important for building distributed applications, which are typically constructed from a set of services that are independently designed, deployed and managed. The widespread adoption of standards offers advantages including: interoperability, stability, tool support and implementation re-use. This is particularly important as Web Services technologies can be acquired from a variety of sources, including major vendors and the open source community.

However, at this point in time there are a large number of industrially-led standardisation efforts, and it is difficult for a user to identify those that have completed the standardisation process, those that are ad hoc standards or indeed those that may have little future in terms of broad acceptance. The sheer number of specifications and the mixed signals coming from industry due to competing specifications in similar areas can leave application architects with an impression that there is no single clear vision for Web Services technologies. Even where there is a clear need for a standard (e.g., workflow, security, transactions, notification), it is still taking a long time for a widely accepted one to emerge. This is partly due to the fact that different sets of vendors are producing competing specifications: it will therefore take time to resolve the differences in a manner that is both technically and commercially acceptable. Examples are WS-Notification [16] and WS-Eventing [17], WS-Coordination [18] and WS-CoordinationFramework [19], WS-AtomicTransaction/BusinessActivity [20, 21] and WS-TransactionManagement [22].

The uncertainty that this range of specifications creates becomes a real problem for developers who must choose which specifications to use in their implementations. If a specification is chosen too early in its lifecycle, then developers may suffer from lack of tool support as well as instability due to changes incurred as the specification evolves through a standardisation process. In the worst case, a specification may never be widely adopted, and so will over time wither and die, adversely impacting any services that chose to adopt it.

Experience suggests that a great deal of effort over a long period of time is required to get the entire WS community to agree on adopting a particular WS specification. In this paper, with WS-I+ we are attempting to identify a minimal set of WS specifications that will not only provide the necessary functionality for e-Science applications but also offer a low-risk choice for developers. Clearly this set will evolve over time as new, widely agreed specifications emerge from industry. The OMII will therefore support a process of monitoring and assessment of the evolving WS landscape and update WS-I+ as and when appropriate.

In the next section we describe the current set of specifications that we consider to be stable, and so suitable for building e-Science “Production Grid” applications.

3 Building on Stable Web Services Specifications: the WS-I+ Approach

3.1 Web Services Specifications: Interoperability and Support

The criteria we have adopted in selecting a conservative subset of WS specifications include:

- Support from the key industrial players. The support of both Microsoft and IBM is seen by many to be important, along with others such as Sun, Oracle, BEA, Fujitsu and HP.

- Stability. Specifications often change significantly as they go through the standardisation process and therefore, if possible, it is better to choose those that are nearing the end of the process.
- Multiple implementations available from vendors and the open source community. The aspiration of the WS-I specification is that implementations provided by different organisations are interoperable. If such interoperability can be achieved, this could significantly reduce development costs and risks. High quality platforms for hosting Web Services are available from a range of vendors (e.g. Microsoft, IBM, BEA and Sun) as well as the open source community (e.g. Apache Axis).
- Tooling to reduce the cost of using the specification.
- Composability with other specifications. It is important to ensure that the use of a specification does not conflict with the use of other, existing WS specifications.

Determining the scope of the applicability of these specifications in detail requires project-specific risk assessment and management. Our more general approach is to reduce risk as much as possible for our "Production Grid" applications, i.e. to select those specifications which are stable and on which most users depend. Picking stable specifications reduces risk and increases the lifetime of the implementation. Choosing specifications with high-quality implementations and tooling will encourage developer productivity and allow a focus on the application, rather than on wrestling with changes in the low-level technology.

The Web Services Interoperability (WS-I) organisation is a crucial element of this low-risk strategy [23]. It was formed to address the specific issue of interoperability between different implementations of WS technologies and is supported by all the major WS technology vendors and a great number of other WS technology companies (close to 200). WS-I produces documents that define how existing, stable, and widely accepted WS standards should be used. Hence, the output of WS-I can be used as a safe basis on which to define the set of specifications used by our WS-based distributed e-Science applications.

The WS-I Basic Profile v1.0 [24] (with draft updating to v1.1) and WS-I Basic Security Profile1 v1.0 [25] describe the way in which the SOAP, WSDL, UDDI, and WS-Security v1.0 specifications should be used in order to guarantee interoperability between different implementations [24 – 28]. WS-I also produces tools to help toolkit vendors assess the conformance of their products to these interoperability profiles. We now consider the key specifications covered by WS-I in turn:

- WSDL gives a standard way of specifying the service interfaces that forms the basis of inter-service contracts.
- SOAP denotes the structure of the messages. Each SOAP message has a header and a body, with the header holding the information to be processed by the "system" and the body passed to the application to be interpreted.
- UDDI is a service discovery standard suitable for relatively static applications which can be supported by a database of service locations and a description of their use. The interaction patterns between services and their consumers can be described through WSDL.

Building services using the WS-I adopted specifications guarantees that:

- The specifications are supported by production-quality tools from the industry and the open source community;
- Good documentation and user education material exist;
- The interoperability concerns at the infrastructure level are at worst minimal; and

- The specific versions of the specifications will not change.

More detailed information on how these WS-I specifications can be used, with the extensions proposed below, when designing services will be contained in a future OMII White Paper "Building Grid services using WS-I+".

We note that security is a well-known hard problem area for Grids and Web Services. Other areas of Web Services are uncertain and immature but usually we are choosing between several reasonable solutions. In the case of security in a Web Services environment, appropriate inter-organisational solutions are not very evident at present. WS-Security provides a basic foundation, including security token transfer, message-based encryption, and message signing. We shall address the subject of security for Web Service Grids in a future UK e-Science White Paper.

3.2 The WS-I+ Set of Web Services Specifications

Our proposal for WS-I+ adds certain critical specifications to those already present in the WS-I Profile. The latter gives us XSD, WSDL, SOAP, UDDI and parts of WS-Security. We discuss below how the additional specifications in WS-I+ are used in the key areas of Service Discovery, Workflow, Messaging, Addressing, and Notification:

Core Service Architecture: XSD, WSDL and SOAP

XSD, WSDL 1.1 and SOAP 1.1 are part of the WS-I profile and will be used by the OMII in providing the infrastructure for Web Service based Grids. Newer versions of these specifications, e.g. WSDL 2.0, will be used as they become accepted.

Service Discovery: UDDI

Since UDDI is part of the WS-I profile it may be used by the OMII as part of WS-I+ to build its registry infrastructure. However, existing projects have built their own registries from WS-I Web Services and it is not clear that the present UDDI specification offers very much to e-Science applications. In the future, the OMII may look to supplement the core registry service with extensions, especially those based on the Semantic Grid approach [29] which adds further semantic information to service discovery or solutions that extend R-GMA from the European DataGrid project (EDG) [30].

Workflow: BPEL

Workflow - or 'programming the Grid' - is recognized as a critical capability for e-Science and work in this area was recently summarized at a GGF10 workshop [2]. Workflow is the general way of linking services together and there are likely to be several standards. Since BPEL [31] is supported by IBM, Microsoft, Oracle and others, we include this standard in the WS-I+ set. However, since BPEL is not specifically directed at e-Science applications, we will explore its applicability and suspect we will need to use its built-in extensibility to support high performance transport and dataflow.

Messaging: WS-ReliableMessaging

WS-ReliableMessaging (WS-RM) and WS-Reliability are almost identical [32] and use message sequencing and acknowledgements to ensure guaranteed delivery of messages delivered in streams. However, federation of the two proposals is possible in this case since the two agree on architecture and only disagree on exact syntax. Thus a filter can straightforwardly map between the two standards. We intend to adopt WS-RM in WS-I+ for

use when reliability is a critical requirement, and provide a simple tool to allow mapping to WS-Reliability if necessary.

Addressing: WS-Addressing

WS-Addressing [33] is a simple specification proposed by IBM and Microsoft that virtualizes addressing and is also proposed for use in other specifications, including WSRF [34]. It allows "end-point references" to be defined independently of the transport protocol. We believe that this is an important capability for building Grids. However, WS-Addressing has not yet been submitted to a standards body and a competing proposal, WS-MessageDelivery [35], from Oracle, Sun and others, has been submitted to W3C and offers a richer set of features than WS-Addressing. These features include the interesting concept of "abstract message delivery properties" defined in a broad context including non-SOAP transport. The proposal also includes many "message exchange patterns" and allows scenarios where multiple protocols are needed to transport messages between two end-points. The latter is often needed to support the interaction between mobile clients (PDA) and Web Services.

Since we need the capability of virtualizing addresses, we will provisionally adopt WS-Addressing for inclusion in WS-I+, since it is much simpler than WS-MessageDelivery yet still offers useful functionality for building Grids. Although WS-Addressing is not undergoing a formal standards process, it is used in WSRF and we believe that adopting it in WS-I+ will help integration with future Globus GT4 releases and other WSRF-based solutions. Nevertheless, a study of the rich features of WS-MessageDelivery will be useful, especially since the e-Science community needs standards for high performance transport between Web Services and clients. We hope that a resolution of the present WS-Addressing/WS-MessageDelivery standoff will take place in the near future.

Notification

Notification covers messages used to provide status alerts and this is often implemented by clients subscribing or listening for such "interrupts". This interaction paradigm underlies event-based programming models. The Java Message Service JMS is a well established solution that can operate in point to point (service to service) mode or through a facilitating broker. WS-Eventing [36] is a simple point to point standard from Microsoft whereas the WSN notification proposal [37] is a family of specifications proposed by IBM and others that offers a more comprehensive set of services including brokered as well as point to point solutions. This publish-subscribe area is reasonably well understood and we expect that some reconciliation of these different standards will be achieved relatively soon. However, given the present state of flux in this area we propose to omit both at present from WS-I+.

Summary

WS-I+ defines a set of Web Service specifications for e-Science Grids that includes WS-I, plus BPEL, WS-RM and WS-Addressing. At present, in the absence of any agreed notification standard, several existing e-Science projects have developed their own notification service based on the WS-I Web Services. These implementations can be maintained in their present form, and adopted as necessary by new projects, until a generally agreed standard emerges.

4 Conclusions

Whilst stable specifications are important, it is also important for communities to explore interesting, emerging Web Service specifications that are likely to be important for Grid application design - even though it would be best to avoid using them in "production" applications until they stabilise. We are therefore adopting a twin-track approach to using WS

specifications in the UK e-Science programme. In the 'production' track, only stable or low-risk versions of the middleware infrastructure will be adopted – generally using specifications that have completed or have entered a standards process. However, in an 'experimental' track, new solutions together with proof-of-concept implementations are being investigated in research projects. These research projects can more easily absorb the risks associated with adopting 'high velocity' specifications that are not well supported by tooling and that may not become standardised or widely accepted by the WS/Grid community. Such experimentation will be vital in determining the value of a specification before it is adopted in production deployments. Experiences will be fed back into any standardisation efforts that are underway in order to steer the evolution of the specification. One issue that is very important for e-Science applications is performance. End-users will only adopt solutions that offer acceptable performance, for example, in the transfer of large volumes of data. We anticipate that Grids built from Web Services will show performance problems, at least initially, and that experimentation and development of alternate, more efficient implementations of some of these Web Services will be necessary.

This paper has described an approach to building Grid applications that attempts to promote interoperability, stability and productivity, despite the fact that Web Services are a young technology with few standards. At the heart of our strategy is the identification of a core set of stable or low-risk WS specifications that we call WS-I+. This e-Science White Paper is intended as the start of an evolving process during which the state of emerging specifications will be regularly monitored. Additional specifications will be recommended when they meet the required criteria of stability, wide industry acceptance, available implementations and tooling. Similarly, over time, we anticipate that the set of services supported by the OMII will broaden as the community identifies and delivers further generally useful services for building Grid applications.

We believe that the approach proposed in this paper will ensure that the UK investment in Grid middleware is protected and that the OMII software distributions can follow an evolutionary path in adopting the emerging Web Service and Grid standards.

Acknowledgements

We would like to thank the many people with whom we have had discussions that have helped us to arrive at this strategy. These include: Jennifer Schopf, (Globus Alliance); John Shewchuck, Jim Gray and Andy Herbert (Microsoft); Dieter Gawlick, Benny Souder and Jeff Mischinsky (Oracle): Robert Fogel (Intel); Brian Hammond and Art Pasquinelli (Sun); Dave Snelling (Fujitsu); Ian Baird (Platform); Steve Loughran (HP) and of course many colleagues from IBM. These acknowledgements do not imply that all of the above necessarily endorse our UK e-Science strategy; merely that discussions with these individuals and companies at various technical and/or political levels have taken place. Useful comments on an earlier version of this paper from Wolfgang Emmerich (UCL) and David Williams (CERN) are gratefully acknowledged.

5 References

[1] T.Hey and A.Trefethen, 'The UK e-Science Core Programme and the Grid', Future Generation Computer Systems 18 (2002) 1017-1031

[2] GGF, "Global Grid Forum." http://www.gridforum.org.

[3] D. Box, "Service-Oriented Architecture and Programming (SOAP) - Part 1 & Part 2." MSDN TV archive, 2003.

[4] M. Mullender and M. Burner, "Application Conceptual View." http://msdn.microsoft.com/architecture/application/default.aspx?pull=/library/en-us/dnea/html/eaappconland.asp: Microsoft, 2002.

[5] OMG, "CORBA/IIOP Specifications." http://www.omg.org/technology/documents/corba_spec_catalog.htm.

[6] W3C, "Extensible Markup Language (XML)." http://www.w3.org/XML/.

[7] W3C, "XML Schema." http://www.w3.org/XML/Schema.

[8] W3C, "SOAP Version 1.2 Part 1: Messaging Framework." http://www.w3.org/TR/soap12-part1.

[9] W3C, "Web Services Description Language (WSDL)." http://www.w3.org/2002/ws/desc.

[10] "Service-Oriented Architecture (SOA) Definition." http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html.

[11] H. He, "What is Service-Oriented Architecture." http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html, 2003.

[12] W. Vogels, "Web Services Are Not Distributed Objects," *IEEE Internet Computing*, vol. 7, pp. 59-66, 2003.

[13] W3C, "Web Services Architecture." http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/, 2004.

[14] "Web Services Policy Framework, WS-Policy." http://www-106.ibm.com/developerworks/library/ws-polfram/

[15] "WS-PolicyAttachment". http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/ws-policyattachment.asp

[16]OASIS, "Web Services Notification (WS-Notification)." http://www.oasis-open.org/committees/wsn.

[17] "Web Services Eventing (WS-Eventing)." http://msdn.microsoft.com/ws/2004/01/eventing.

[18] "Web Services Coordination (WS-Coordination)." http://msdn.microsoft.com/ws/2003/09/wscoor.

[19] OASIS(WS-CAF), "Web Services Coordination Framework (WS-CF)."

[20] "Web Services Atomic Transaction (WS-AtomicTransaction)." http://msdn.microsoft.com/ws/2003/09/wsat.

[21] "Web Services Business Activity (WS-BusinessActivity)." http://msdn.microsoft.com/ws/2004/01/wsba.

[22] OASIS(WS-CAF), "Web Services Transaction Management (WS-TXM)."

[23] "Web Services Interoperability (WS-I)." http://www.ws-i.org.

[24] WS-I, "Web Services Interoperability (WS-I) Interoperability Profile 1.0a." http://www.ws-i.org.

[25] WS-I, "Basic Security Profile Version 1.0 (Working group draft)." http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2004-05-12.html, 2004.

[26] OASIS, "Universal Description Discovery and Integration (UDDI)." http://www.uddi.org/.

[27] OASIS, "Web Services Security (WS-Security)." http://www.oasis-open.org/committees/wss.

[28] W3C, "Simple Object Access Protocol (SOAP) 1.1." http://www.w3.org/TR/2000/NOTE-SOAP-20000508/.

[29] D.De Roure, N.R.Jennings and N.R.Shadbolt, 'The Semantic Grid: a future e-Science infrastructure', in 'Grid Computing: Making the Global Infrastructure a Reality', edited by F. Berman, G.C. Fox and T. Hey (Wiley 2003) 437-470.

[30] The European DataGrid project. http://www.eu-datagrid.org

[31] BPEL: Business Process Execution Language for Web Services (OASIS) V1.1 May 2003. http://www-106.ibm.com/developerworks/library/ws-bpel

[32] S. Pallickara, G.C.Fox and S.Lee, 'An Analysis of Reliable Delivery Specifications for Web Services'. http://grids.ucs.indiana.edu/ptliupages/publications/WSR-AnalysisPaper.pdf

[33] WS-Addressing. http://www-106.ibm.com/developerworks/library/specification/ws-add

[34] WSRF. http://www.oasis-open.org/committees/tc_home.php?wgabbrev=wsrf

[35] WS-MessageDelivery. http://www.w3.org/Submission/2004/SUBM-ws-messagedelivery-20040426

[36] WS-Eventing. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/WS-Eventing.asp

[37] WS-Notification. http://www-106.ibm.com/developerworks/library/specification/ws-notification