



Report of the User Requirements and Web Based Access for eResearch Workshop

Editors: Jennifer M. Schopf, NeSC and Argonne National Laboratory
Iain Coleman, NeSC
Rob Procter, NCeSS
Alex Voss, NCeSS/NeSC

Contributing Authors: Malcolm Atkinson, Robert Esnouf, Arnaud Marmier, David Meredith, Michele Osmond, Simon Peters, Rob Allan, Robert Stevens, Anne Trefethen, Jenny Ure, Choochan Youn, Stephan Zasada

Date: 30-10-2006

Abstract:

The User Requirements and Web Based Access for eResearch Workshop, organized jointly by NeSC and NCeSS, was held on 19 May 2006. The aim was to identify lessons learned from e-Science projects that would contribute to our capacity to make Grid infrastructures and tools usable and accessible for diverse user communities. Its focus was on providing an opportunity for a pragmatic discussion between e-Science end users and tool builders in order to understand usability challenges, technological options, community-specific content and needs, and methodologies for design and development. We invited members of six UK e-Science projects and one US project, trying as far as possible to pair a user and developer from each project in order to discuss their contrasting perspectives and experiences. Three breakout group sessions covered the topics of user-developer relations, commodification, and functionality. There was also extensive post-meeting discussion, summarized here.

Additional information on the workshop, including the agenda, participant list, and talk slides, can be found online at <http://www.nesc.ac.uk/esi/events/685/>

Reference: NeSC report UKeS-2006-07 available from http://www.nesc.ac.uk/technical_papers/UKeS-2006-07.pdf

Executive Summary

The User Requirements and Web Based Access for eResearch Workshop, organized jointly by NeSC and NCeSS, was held on 19 May 2006. The aim was to identify lessons learned from e-Science projects that would contribute to our capacity to make Grid infrastructures and tools usable and accessible for diverse user communities.

In general, the workshop noted the following:

- There is a need for closer interaction between end users and tool builders over the full design, development, and use lifecycle
 - Real successes came through committed and extended engagement with and by real end users
 - Embedding users or developers in each others' groups worked well for several projects
- There is a need for a better understanding of the makeup of successful e-Science project teams
 - Projects should include expertise sourced from the wider organisational environment
 - Support roles must be considered along with the more traditional roles
- Requirements gathering is complicated by different classes of users
 - User requirements will change depending on experience with the technology
- There is a lack of agreement about needed basic functionality and common practices
 - A mechanism is needed to communicate common approaches in terms of functionality, technology, or requirements gathering
- There may be a need for common base implementation
 - This should have basic functionality or design patterns that can be selected independently
 - It should be built on standards and supported to production quality levels
 - It should be capable of supporting domain specific extensions
 - Emerging technologies might allow for additional usability gains, but must be evaluated against possibly steep developer investment

A follow up consisting of a series of smaller, focused meetings has been suggested. These meetings would be small (approximately 10 people each), and by invitation only. They would be pragmatic, and would concentrate on a single topic for the day. Suggested topics include:

- 1) What is the makeup of a good team – from tool builder to end user?
- 2) What is the minimum functionality that needs to be offered to end users to make the NGS an attractive platform for a well defined set of application domains?
- 3) Given (2), what tools exist to meet these needs, and what would need to be done to extend them to the NGS setting?
- 4) What are the most appropriate methodologies for design and development?
- 5) Are there common lessons for developing Web-based interfaces that can be communicated more broadly, and how do we do this?

An additional suggestion was that one focus of the new OMII-UK funding within JISC could be on the integration, testing, documentation, and support of a core set of portal functionality for the NGS.

1 Purpose of the Meeting

Many e-Science projects have successfully demonstrated the potential of Grid-based infrastructures and tools within a number of disciplines, but these technologies have not yet reached their full potential for impact on research methodologies. The reasons are likely to be complex and multi-faceted, ranging from the technical to the socio-political.

This workshop was the first in a series planned to bring members of the e-Science community together to discuss how we might respond to the need for greater uptake. Its aim was to identify lessons learned from e-Science projects that would contribute to our capacity to make Grid infrastructures and tools usable and accessible for diverse user communities. Its focus was on providing an opportunity for a pragmatic discussion between e-Science portal users and developers in order to understand usability challenges, technological options, community-specific content and needs, and methodologies for design and development.

The meeting consisted of a broad set of walkthrough case studies of eResearch projects with Web-based interfaces in current use, from the viewpoint of both the users and developers. This was followed by a set of breakout sessions that focused on specific issues brought up by the case studies, namely user relations, commodification of solutions, and functionality.

In planning the meeting, the initial questions identified to be addressed included:

- What should our road map be for improving usability and engagement via portals? Are there unresolved challenges or is it just development?
- What aspects of portals are crucial to greater user engagement with e-Science?
- What are the key usability issues for new and diverse research communities?
- Is it necessary for work on portals to be led and strongly connected with research leaders in a discipline they serve? If so, how can we gain their commitment?
- In what ways can we capitalise on or reuse portals developed for one particular community?
- If we invest in portals, what else would we need to do before they are useful? E.g., what critical mass of services and data do researchers need access to?
- If we invest in portals, and we can't afford everything, what do we do first?
- If portals are important, how can we bring in more resources (e.g., from supported disciplines) to contribute to a portal?
- How will we decide whether portal investment has been successful?

Additional information on the workshop, including the agenda, participant list, and talk slides, can be found online at <http://www.nesc.ac.uk/esi/events/685/>

2 Case Studies

2.1 *Discovery Net – Michele Osmond*

Discovery Net allows users to create workflows and publish them to a Web portal in a simplified form for parameterisation and execution. The users of a Translational Medicine portal at Windber Research Institute and the bioinformatics team at an organisation carrying out crop research provided feedback on their experiences. Both groups highlighted data sharing and integration and support for dynamic, iterative workflow development as critical functional requirements for their system. After using the portal, the Windber users particularly valued ease-of-use and high interactivity in their portal tools, while the bioinformatics researchers appreciated the flexibility and power of the workflow system. They pointed out particular parts of the Java client and portal which should be improved from a usability perspective, or extended for more advanced use.

The Discovery Net portal is currently available in two flavours: a Java servlet-based portal (for simple and easy access to all Web-deployed workflows), and a JSR-168 portlet version bundled with Jetspeed 1.6 (allowing integration with custom portlets and development of more advanced, organisation-specific site layouts). The concept of Web-deployed workflows has been popular as an easy way to access functionality: typically, a small group of power users install the Java client to develop and deploy workflows, which are then available on the portal to a larger base of users. Usability design and testing of the portal has however been of lower priority than that for the more complex Java client, and as a result some aspects of the deployment tool and portal have suffered. Additionally, as users become more familiar with the tools available they expect further levels of interactivity and visualisation. These points will be examined in future portal development.

2.2 *eMinerals: User Perspective – Arnaud Marmier*

The aims and requirements of the eMinerals group are to tackle problems of environmental relevance using numerical simulations. In a Grid environment this mainly entails the efficient submission of a large number of simulations. Although portals have been trialled, they have proved unpopular with users. The main reason is that users are used to a command line/scripting paradigm when dealing with traditional resources and consequently portals are quite alien. In addition, the command line/scripting paradigm is regarded as more powerful. For this already sophisticated category of users, e-Science has reached a point at which it is accessible to the majority (c.f. the National Grid Service).

2.3 *eMinerals: Developer Perspective – Rob Allan*

eMinerals is a large project in its second round of funding. It is coordinated from the University of Cambridge with around six sites involved. There are e-Science developers

at Daresbury, Reading and UCL, with application developers and end users at other sites. All staff work in teams comprising both developers and users. Collaboration and communication are key issues with tools such as MAST, AccessGrid and instant messaging being used to enhance email and phone.

Development in other areas has focused on deploying a set of reusable services to access distributed compute and data servers. The eMinerals “mini-Grid” comprises resources at the participating institutions. Tools are as follows:

- Large compute clusters – Condor-G, Globus, PBS, LoadLeveler, SGE
- Compute “pools” – Condor-G, Condor
- Data stores – SRB vaults and client command interface

Workflow underpins the whole system with scripting using a combination of Condor-G and SRB with the RCommands for meta-data management. AgentX was developed as a semantic tool to enable output from one application to be interpreted by another or by a GUI. It uses RDF and OWL with ontologies to extend existing data models.

The services which have been developed mostly use Web services. They could be interfaced to portlets, but the users are happy with a script interface as they are mostly established scientists who are used to command-line programming.

2.4 GEMEDA – Simon Peters

Awareness of the potential for eResearch in the discipline of economics is minimal, even though areas such as empirical economics/finance, agent based modelling, and experimental methods have potential for user engagement. The present users’ activity in the first area came from dialogue with local computer research services personnel, rather than from within the discipline, and was prompted by a workshop sponsored by the Research Methods Programme initiative in UK Social Science. This dialogue contained two promises: improved workflow from Grid-enabling data sources, and instantaneous computation using high performance computing systems; and one difficulty: implementation of the “promises” (services, software, hardware, staff). Details of the latter are contained in the developer’s perspective; however, the essence of the solution was delivered by the NGS (National Grid Service), accessed via a portal, and resulted in a collaboration between the users (economists) and local research computing services (databases, middleware, visualization).

The problem at hand was empirical modelling that required using multiple data sets (aka statistical data fusion), and an element of this was deployed as a Grid service. This demonstrated the shortcomings of the two aforementioned promises of eResearch from the users’ perspective as compute intensive activity is batch, and the users’ (naive) interpretation of what Grid-enabled data entailed was different from the reality. Both of these facts, combined with the lack of locally available resources (equipment, staff, users’ human capital) constitute barriers to adoption in economics, in particular, and classical Social Science in general. Hopefully, these barriers may be broken as the National Grid Service and related services mature, coupled with engagement with all potential UK

users, not just the physical sciences, through association with initiatives such as NCeSS (National Centre for eSocial Science) and AHESCC (Arts and Humanities e-Science Support Centre).

2.5 *myGRID – Robert Stevens*

The ^{my}Grid project brings together many years of background in collaboration between computer scientists and biologists in developing software solutions in bioinformatics. Together with the European Bioinformatics Institute's experience, this has enabled the project to identify that workflow and services to support workflow are a good, general solution to data gathering and data analysis in biology. Gathering of general user requirements was not undertaken, except to prioritise services within this framework. The background within the project enabled both the functional and non-functional requirements to be gathered, namely a technology for joining distributed data resources and tools in an open system, and support for the scientific process (particularly through provenance) in an open, high-level and auditable manner.

Rather than a traditional requirements-gathering activity, ^{my}Grid has taken the approach of having bioinformaticians at all stages of design and development. In bioinformatics, people exist at levels of experience stages from near-biologists, who analyse data, through to near-informaticians who can develop applications and engineer software. ^{my}Grid has bioinformatics developers who are bioinformaticians right at the centre of the project. This helps to ensure that the tools do what the users need, as the tools are built by the users. This was done with prototypes, some of which were actually thrown away. The project avoided standards chasing, and used as much pre-existing software as possible, only developing in-house where necessary. The best example of this is the development of the XSCUFL high-level workflow language that suited the needs of our users in a way not supported by languages current at the time.

In addition, from the end of year one we have had bioinformatics Ph.D. students using the Taverna workbench and ^{my}Grid services to perform in silico analyses on biological data. This both tests the software and, more importantly, validates design and generates enormous amounts of requirements and feature requests. This embedding of users in the project also acts as a form of evaluation. It is a practice to be encouraged in all projects. Ph.D. studentships should be given to projects and it should be mandatory that these are given to colleagues who are potential users.

We identified our target users early in the project. As mentioned previously, bioinformaticians exist along a spectrum from occasional users of bioinformatics tools, through those who do complex tasks, but in only a narrow range, to those who do bespoke analyses and have a deep understanding of a range of tools and their technical aspects. It is this last, deeply skilled group that ^{my}Grid targeted. These users can develop workflows representing analyses that can be presented through portals to other users.

The primary usability aim was to be effective: getting the job done at all is a major advance. Micro-usability, at the widget level, was therefore not the highest priority.

Efficiency primarily comes in cutting time taken in analysis from potentially days to minutes. Furthermore, the computational aspect brings a systematic aspect to the analysis missing from manual counterparts. Satisfaction brings us back to effectiveness again: getting the bioinformatics job done is a major step. The emphasis on publication and changing pressures within the e-Science programme also influenced the priorities given to this important topic area. Nevertheless, myGrid has developed tools with which biologists have performed many experiments and generated new biology.

2.6 eHTPX: Developer Perspective – David Meredith

Experience gained in the eHTPX project indicates that addressing the following four key points helps to achieve increased usability of e-Science software.

1. Rich portal interface. Traditionally, Web applications are not as graphically rich and interactive as desktop GUI clients (next generation Web applications and tools are increasing usability through more reusable interface components, e.g. JSF, and dynamic interaction technologies, e.g. AJAX). The eHTPX portal interface requires a rich user-interaction experience akin to that provided through desktop applications in order to easily input necessary data/information. The new tools/APIs mentioned have helped address this requirement in eHTPX but at the expense of increased programmatic complexity (increased learning curve).
2. Customised (application-specific) portal interfaces to address specific requirements of client communities. Tailoring the portal interface to suit the requirements of a target audience has certainly increased usability within eHTPX. However, it has proved very difficult to balance portal interface flexibility (so that the portal is suitable for all users) with the correct level of customisation/tailoring. This has proved difficult even within the same project. Both user-interface flexibility and audience specific customisation are possible, but at the expense of greatly increased programmatic complexity.
3. Well defined service/resource interfaces (following the principles of a Service Orientated Architecture, e.g. WS, WSRF). This simplifies access to a service and encourages usability. Even if potential clients do not adopt the eHTPX portal interface, clients may still consume the Web services. However, poorly defined service interfaces immediately discourage the take-up of e-Science resources (e.g. language dependencies in WSDL).
4. Frameworks for modularity + reusable software (e.g., Portal/portlets, WSRP). The portal/portlet programming model should help software reusability, but JSR168 requires closer integration with well defined Web application frameworks such as JSF and Struts.

2.7 *eHTPX: User Perspective – Robert Esnouf*

The eHTPX project seeks to manage the flow of information between the processes involved in the protein crystallography pipeline. The main stages involved are target selection, protein production, crystallization, x-ray data collection, structure solution (phasing) and deposition in public databanks. Increasingly, especially in larger scale facilities, these different stages are performed by different people and at different sites, creating a need both for standardized, secure exchange of data and tools for project management. The volume of data is also increasing: with current technology, a single x-ray beam line at a bright synchrotron source can produce 74Mb images with 1s exposures. One ultimate goal of the eHTPX project could be seen as creating the ability for a researcher to manage the complete pipeline remotely. Thus, and in contrast to many other projects, the main goal of eHTPX is managing (meta-) data exchange and the potential user base is very large (>1000 active researchers in the UK alone), whereas issues of access to compute resources are of secondary importance.

The project has developed, or has contributed to the development of, several facilities that are increasingly used in the protein crystallographic community. In Oxford, a web-services based interface for remotely managing high-throughput crystallization trials is central to our work. It supports a user base of more than 100 researchers and has recorded over 10^6 trials and 35×10^6 images of trials. This tool is currently being generalized for wider distribution. Two prototype hubs/portals have been developed for exchanging information about protein crystals with synchrotron sites based on secure web service-based messaging. At the ESRF synchrotron, eHTPX has contributed to the development of an experiment recording tool that is now being rolled out for general use and allows remote users web-based access to diffraction images, and at both the ESRF and SRS synchrotrons the project has collaborated in the development of an expert system for automating x-ray data collection that can communicate directly with the beam line control software. For the molecular replacement approach to structure solution, the project has developed a fully automated system that can take advantage of Grid computing, and this is already in general use as one of the most convenient packages for this purpose.

By having prototype services in use, the eHTPX project has been able to receive much useful feedback from the research community that it supports. In order for these technologies to become more widely used we have to recognize that most biologists do not consider learning about and managing IT resources as a valuable use of time. They will adopt new tools and technologies if they offer a clear advantage over current methods, if they are easier to use, if they require no local administration, if they reduce the possibilities for human error in recording results and if they are well integrated with other tools, resources and equipment. To get to the stage where we can receive general user feedback has been very difficult, and we view the use of project ‘champions’ – researchers who will work with developers to tackle an initial problem and not be put off by teething troubles and technical difficulties – as essential to kick-start development. In summary, the ‘e’ is much less important to the researcher than the ‘science’ and the use

of new technologies should be invisible to the end user. A simple tool addressing a well defined problem is more likely to be taken up, although of equal importance to the protein crystallography field will be the data management policies adopted by Diamond, the new UK synchrotron.

2.8 *RealityGrid – Stefan Zasada*

The Application Hosting Environment (AHE), developed through the RealityGrid project, is a lightweight, WSRF based environment that allows scientists to easily share and use unmodified applications on a computational Grid. The AHE is designed to lower the barrier of entry for users of computational Grids, simplifying the process of launching jobs and staging files between machines. The design of the AHE was motivated by difficulties encountered using current Grid middleware solutions, such as complicated configuration and installation and dependencies on lots of supporting software being installed.

The AHE exposes a Web service interface, which the user interacts with using Java command line and GUI clients. This approach was taken, rather than using a Web portal interface, as it provides a more flexible model for user interaction, for example allowing end users to call the command line clients from Perl scripts to orchestrate scientific workflows.

2.9 *TeraGrid – Choonhan Youn*

GEON (GEOscience Network) is creating an IT infrastructure to enable interdisciplinary geoscience research by utilising state-of-the-art information technology resources. The research challenges include the management of vast amounts of data, access and preservation of data, data integration, simulation in computational Grids, advanced visualization, and the archiving and publication of the scientific results.

From the user perspective, GEON provides a resource registration system, a discovery system, a user collaborative workbench, computational applications, and a workflow system as main eResearch facilities. Users may have difficulty in providing a more dynamic workflow system and the computing facilities for simulating the large applications. The GEON portal is targeting ease of use in all GEON services, and integrating access to tools and resources in the user workspace, called “myGEON”. The lessons learned from the GEON work are that its main strengths are its standards-compliant approach and its use of open source libraries and tools for most implementations. The main weaknesses are issues regarding the implementation of a highly user interactive, friendly interface within the portlet framework.

From the developer perspective, GEON has decided on the design and implementation through workshops, GEON PIs, and major geoscience conferences. Prototypes are built using the gathered requirements and then the spiral model of software development is followed to enhance the prototype. The architecture of the GEON portal is based on the service oriented approach around the Web services model, and also uses the portlet-based

system for portal reusability. The technical strategy in GEON is a two-tier approach: on the one hand using best practice, including the use of commercial tools and open standards where applicable, while on the other hand developing advanced technology and doing computer science research. The constraints in development include interoperability issues due to use of existing tools, usability problems due to the unfriendly portlet user interface, and the lack of user inputs in some cases. The community is still evaluating this system.

Future plans for GEON include adding and developing new functionalities based on requests from PIs and the community, and further improvements to the usability of the portal. Furthermore, GEON will be provided with a secure role-based authorization control to fully integrate into the portal, as well as added WSRP and WSRF services.

3 Breakout Sessions

There were three breakout sessions during this workshop. Each session was tasked with addressing specific questions as follows:

User-developer relations:

- How can we ensure that users are taken seriously by the development community?
- How can users and developers learn each others' language and how do we sustain the dialogue?
- How can we resource and manage projects such that user requirements remain the driver?

Commodification of solutions:

- How can we package up technologies so as to flatten the learning curve?
- How can we improve reuse, and what role does standardisation play?
- How can we better align technologies with the wider world of organisational IT?
- How can we provide support that scales to larger user communities?

Functionality:

- What is the right granularity of services?
- Can we identify a critical mass of services and/or resources?
- Do we need generic or domain specific capabilities?
- How can we link new tools with ones that scientists currently rely upon (LiMS systems, SPSS, etc.)?

Following the breakout sessions, the groups were asked to report back their answers to the following questions:

- What are the FIVE good things that we are doing now we should communicate to & sustain for the community?
- What THREE things do we want to do in the next 12 months?
- What is the ONE big thing we need to start planning for delivery in 12-24 months?

3.1 Breakout 1: User-Developer Relations – Anne Trefethen, Jenny Ure

We were asked to consider three questions relating to the theme, and to make recommendations. The discussion covered the issues of sustaining engagement between users and developers, and facilitating the communication of domain specific knowledge. It also highlighted the need to look at the different rewards and pressures on users and developers in practice if engagement is to be sustained.

3.1.1 Questions

How can we ensure that users are taken seriously by the development community?

Sustaining the engagement of busy users in this process is seen as something that is always hard, as engagement with developers can be perceived as absorbing precious time without immediate benefits. It can be a focus for conflict as compromises and misunderstandings are teased out.

Reward

The provision of early rewards for users was seen as one way of supporting continued engagement by users – this might be faster processing of results, for example, that affords clear benefits in terms of user requirements.

Opportunity for Engagement

The group also felt strongly that Access Grids, including personalised Access Grid (PIGS) could go some way towards making regular access to distributed users easier on a virtual basis where real face to face dialogue was unrealistic. Providing shared spaces, real or virtual, was seen as a requirement.

Context for Engagement

The use of rapid mock-up or early prototypes for collaborative design was also discussed as a way of supporting the engagement of users and developers in a context with shared visual reference points. Some of the participants had found the use of Dreamweaver or other packages useful for 1:1 mock ups of interfaces with users.

Opportunity for Sharing and Re-Use

The group felt unanimously that there was a need to share experience of best practice, of problems and solutions, and of case studies through AccessGrid, websites and workshops. The discussion suggested that there is not enough sharing and dissemination of recurring generic problems and solutions.

How can users and developers learn each others' language and how do we sustain the dialogue? How can we resource and manage projects such that user requirements remain the driver?

These two questions are addressed here together as the flip-sides of the same issue.

There was clearly a concern with the difficulty of achieving real conceptual understanding of very complex processes and dependencies in other domains. Many projects depended in part on the presence of multi-disciplinary people on the team who were able to mediate some of the exchange. 'Tame' users who had some IT experience or project involvement also played a part in this. It became evident however that many projects depend on a small core of such users who may not represent the naïve user in practice. Naïve users were less likely to be included, communication was more difficult, and their input was often late in the process.

Early mock-ups: 'Give them something to hate'

As in the earlier section, providing something early was a good vehicle for generating engagement – especially if it was not what users wanted!

Rewarding sustained dialogue

There may be a conflict between the concerns of developers to provide a robust early design, and the tendency of real users to provide additional requirements that entail late changes with consequent impacts on the cost, scheduling and elegance of the final design. Currently, the short term rewards for designers may militate against the broad and sustained engagement of users.

Recommendations here include consideration of how both users and designers can be rewarded for sustained engagement in such projects, given that engagement can carry penalties for both communities. Engagement is time-consuming, and often raises issues that were not initially evident, that can be contentious and political, and that generate further requirements for development or redesign.

Mediation

In practice, many projects depended on multi-disciplinary people who were able to mediate this process throughout the project. Their role was clearly important, and perhaps there are implications for this in the design of project teams, and in professional development.

3.1.2 Recommendations

Most of the strategies mentioned were based on ensuring more sustained, contextualised and in-depth dialogue as a basis for constructing common ground and as a basis for more situated decision-making. Some also related to the difficulty of aligning the very different, often competing, aims and criteria of developers and users which also shaped preferences.

1. Opportunities for sharing best practice and experience including case studies

The main recommendation was the provision of more opportunities for real and virtual sharing of case studies and best practice in this area through AccessGrid meetings, a website of these resources and more small-group opportunities for discussion of best practice. NeSC was seen as a focus for this. Shared tools were also seen as important here, with a wish to re-use what was there rather than re-invent.

2. Opportunities for more extended, in depth collaboration in small groups

- More small groups where ‘real questions’ get asked and everybody participates
- Maximum of 12 members, so that all participate as opposed to collaborations where the ‘usual suspects’ ask all the questions

3. Early engagement in participatory design (‘Give them something to hate’)

- For example, mockups e.g. using Dreamweaver to provide a tangible and situated context for visualising and ‘walking through’ possibilities and potential problems. (There was concern among some of the group however that this could be a distraction, and that user feedback was unreliable / likely to change)
- Co-realisation and participatory design approaches

5. Explicit rewards for engagement and dialogue

- Highlight the benefits of engagement in the requirements process to users and developers – case studies of good and bad examples
- Build in explicit rewards for users and developers as part of projects – currently there are sometimes perceived penalties for developers in practice (late or additional requirements with all the implied costs and delays), and penalties for users (time for meetings and the often contentious/political nature of the trade-offs involved in aligning different requirements).

In conclusion, there was a perception by some in the group that the effort of producing a good design was not valued as much as publishing a paper about a problem, and ‘changing the RAE metrics’ was one of the recommendations from the back row!

3.2 *Breakout 2: Commodification of Solutions – Alexander Voss*

The commodification breakout group discussed the need and potential for the provision of Grid resources and technologies in the form of generic packages, widely adoptable practices or service offerings. Our discussion started off with a focus on the needs of the e-Social Science community and the problem of deploying Grid technologies in research areas not traditionally associated with high-performance computation. People working in these areas face the problem of assembling the resources needed for eResearch, both in terms of compute resources and skilled staff. Another question related to the problem of deploying Grid technologies in environments where policies regarding IT security and confidentiality are at odds with the idea of sharing data across organisations. This would be true in the case of (potentially) identifying data held by such organisations as the office of national statistics or the NHS. The problem of adoption of eResearch technologies is clearly an organisational problem, so usability needs to be seen as a wider issue than the user interface alone: it needs to encompass topics of scientific modes of investigation and working practices as well as issues of technology supply and organisational arrangements.

While resources such as the National Grid Service can provide an excellent starting point and underlying infrastructure, there are questions about the fit of Grid technologies with the modes of investigation and research methods in the social sciences which may differ significantly from those recognised elsewhere. An example would be qualitative research methods, such as ethnographic observation. Such differences may lead to different requirements, perhaps not so much at the level of generic middleware or the provision of basic capabilities but at an application level. Given these potential problems, how can we identify where the overlaps and differences between different user constituencies are, understand the demands for commodities, and provide ways of packaging functionality so that it is useful in sufficiently many contexts? Our feeling was that the most immediate need is for practical and flexible steps to help with the appropriation in a range of contexts, e.g. the provision of templates, patterns and examples or demonstrators. These should help to raise interest amongst those not yet committed to using Grid technologies and should reduce the initial effort required for adoption by, for example, providing conveniently packaged technologies that can form starting points for further exploration and systems building.

In the longer term, if the vision of a pervasive eResearch infrastructure is to become a reality not just for university researchers but for the wider public, there will have to be clear models for adoption that are aligned with the different ways in which different kinds of users source and use information technologies. The use of Web browsers as a common baseline that can be assumed across context and provided for through portal technologies is one crucial step, but there are questions as to what extent this limits the kinds of involvement users can have in eResearch projects. Different user constituencies are likely to require different functionality, levels of control and support. The current complexity of the software stack needed to create virtual research environments needs to be made more manageable (by reducing the number of options, providing templates, packaging up working partial solutions, etc.) and support structures need to be developed that scale to

the user communities envisaged. Only in this way will we be able to realise the vision of “democratic eResearch” (à la Malcolm Atkinson).

Five good things to communicate and sustain:

- Portal standard (JSR-168), higher level interfaces.
- Stable portal frameworks are readily available.
- Inspectability of technologies (open source and repositories of code).
- Commodification of services, e.g., NGS.
- Community, and acceptance that community tools (Sakai) need to be part of a VRE.

Three things to do in the next 12 months:

- Identify where the overlaps and inherent differences are (avoiding fragmentation as uptake increases), understanding the demand for commodities – different kind of requirements problem.
- Software Stack – what can and should be commodified.
- Provide paths, templates, etc. – exemplars and demonstrators that end users can access.

One big thing to start planning:

- Long-term sustainability: professionalisation – can mean commercialisation but not necessarily.

After the workshop, we have continued the discussion by email as there were a number of issues that could not be addressed in the short time available for the breakout sessions. Issues raised included the role of industry involvement and the need for a development to be driven by ‘demand and use’ rather than technology push. The question of the role of open source software and open standards has also been flagged as being important in its own right and also in relation to the other topics discussed.

3.3 Breakout 3: Functionality – Rob Allan

The functionality breakout group discussed the different kinds of interfaces that are, or may be, desired by users. We considered command line interfaces, web interfaces, and portals. Users who are not computer literate may adopt the simplest solution, such as a command line interface, even though a portal might better suit their requirements. Portals are community interfaces and need to meet the needs of a wide range of users. Do users want a range of services available in one environment? There was a lot of discussion of workflow and scripts. Web interfaces are familiar, so we assume people like them, but this needs to be tested. People also seem to want the interface to be customisable, a bit like a desktop tool. Portals can provide integration of functionality, aggregating multiple services and encapsulating pre-defined workflows.

We also looked at whether we can identify a critical mass of services and/or resources. At present we are exploring the space between web applications and desktop applications. The JSF framework also gives a richer client in a portal. Users who are reluctant to install software might try a portal first, and perhaps go on to install something else later if they find that they require the additional functionality. Simple functionality might be available from pervasive Web interfaces (e.g., task monitoring), more complex tasks from desktop clients. We can use a common underlying set of remote services.

The consensus of the group was that, while some services can be common and centralised, solutions should not be enforced. There could be more than one similar service available. One example that we discussed briefly was that we might want several different levels of security depending on the type of data (many portals can use a security stack). How we know what level of security a particular dataset requires depends on who the user is.

We went on to discuss how can we link new tools with the ones that scientists currently rely upon, such as LiMS, SPSS, and so forth. The ISME VRE project uses MatLab, Amira, ABAQUS etc., as well as spreadsheets such as Excel. We would like to use this on a portal. Can we link to it or have the functionality in a portal? We want to launch existing applications in portlets. In general is it possible just to download the dataset and run the application? Things like AccessGrid are exceptional applications: their interfaces need to be understood and linked into portlets. Portals will not be able to do everything - we also need richer clients. It was noted that Discovery.Net also interacts with applications in a richer Java client. Applets can be used for some tasks. There are already ad hoc integration solutions in some cases – can we learn more general lessons from these? There are also products such as SpotFire that can launch portal services – an inversion of the usual model. Tools for converting Java code to JavaScript/AJAX might be useful.

We reported back as follows. We recognised that we need to further engage with the users so that they can help us to answer the questions and validate our initial response.

Five good things that we are doing now that we should communicate and sustain:

- We are learning/re-learning tricks about user engagement and usability.
- Getting feedback on possible solutions enables us to reframe the question -> software life cycle.
- Lots of good technical work going on in projects, but not always being communicated across – if this were done some of it could be reused to grow the critical mass of services/ resources.
- Good knowledge of state of the art technology, JSR-168, JSF, WSRP, etc. We need to communicate this and collectively learn how to use it.
- Passing on our knowledge of technical fixes for what we learn of requirements that emerge in different situations, and our understanding of the nature of those requirements themselves.

Three things we want to do in the next 12 months:

- A repository or some kind of tool for sharing outputs – NeSC, OMII, JA-SIG? Decide how we preserve/maintain/distribute material which has taken effort to produce
- Find a community process to share the kind of knowledge described above.
- Find a way to engage with users and check our answers.

One big thing we need to start planning in 1-2 years' time:

- Coordination of the important issues, such as how e-Science can reach out to a larger community base and integrate large sets of resources, and how to enable delivery of existing services to new and different projects.

4 Post-Meeting Discussion

The workshop drew attention to several shortfalls in the current approaches to Web-based interface design and development, but discussions included the broader issues of uptake of eResearch tools in general. This section summarizes post-meeting discussion on a variety of topics.

4.1 Collaboration and Interaction

As noted in other workshops, there is still a need for closer collaboration between application scientists and tool builders. The message of strong user involvement is reinforced by the fact that it is not clear whether developers are building tools with the functionality that users are actually requesting. This was most notable in one pairing of talks where the developer discussed in detail the portal that was developed, only to have the application user say in the following talk that they were not using a portal at all, preferring command line tools with basic functionality.

The lack of collaboration has some facets that are not necessarily specific to e-Science, and in fact have been previously addressed in the software engineering literature. The importance of user involvement has been long argued for and has now been widely accepted (for a good summary of the arguments, see [1]). In addition, successful technology transfer and adoption has been observed in other application domains through the use of *hybrids*, user-developer cross-over project members [2, 3, 4].

In e-Science we see two types of hybrids. In the first instance of hybrids, users are embedded in the development teams, for example as practiced by bioinformaticians in the ^{my}Grid project, where this dual role allows users to directly influence the direction taken by code development. An alternative approach is to embed technical developers within domain research teams, as was done with the Condor team and Particle Physics Data Grid (PPDG). This is often easier from the developer point of view since immediate feedback is available; however, as users are not necessarily a homogenous group, the developer embedding approach may make it difficult to reflect this heterogeneity in the resulting product.

Projects like ^{my}Grid have also made significant use of contributions from their hybrid user-developers. It remains an open question as to what extent the hybrid approach will be replicable in other eResearch application domains and communities where the separation of these roles is becoming more common [5]. In some fields, such as bioinformatics and particle physics, the requisite technical skills often go hand in hand with the nature of the research practices, but this is not necessarily the case in other disciplines. Moreover, becoming a hybrid is sometimes a problematic strategy career-wise for researchers, which must be taken into account. More commonplace, perhaps, but equally important is the contribution of ‘project champions’ such as used by eHTPX (Sections 2.6 and 2.7), sometimes referred to as alpha testers by development teams. These are often bleeding-edge end users who are willing to comment in detail on prototype software.

Creating a partnership between users and tool developers calls for creating a common ground between the two, not only so that the latter grasp users’ requirements but so that the former grasp what the technologies are capable of. The need to avoid exaggerating available functionality in order to attract end users and avoid falling into the trap of technology push was emphasized in the discussion of GEMEDA (Section 2.4).

4.2 Requirements Gathering

The scale and distributed nature of many e-Science projects also complicates the interactions between end users and developers. Breakout 1 (User-Developer Relations, Section 3.1) found the problems faced in e-Science to be an extension of those found in all heterogeneous and distributed systems, only scaled up. Candidate answers may be found in other areas but will need to be appropriated and adapted with a view to the specific conditions of eResearch. In addition, the distributed nature of e-Science projects places additional stress on the approaches to user involvement which have relied on proximity to create and sustain common ground between the participants [6].

The interdisciplinary nature of many e-Science projects also adds additional difficulties. Diverse user populations can have vastly differing requirements. The eHTPX discussion (Sections 2.6 and 2.7) emphasized how reconciling different user requirements can be not only a conceptual challenge for developers but also a significant practical one.

In addition, as users gain more experience, common use cases change, as emphasized in the discussion of Discovery Net (Section 2.1). It remains an open question as to how user requirements for projects with these characteristics be generated. In the field of requirements gathering itself, a recent position paper on this subject [7] went as far as to state that “We know of no scalable methods of requirements analysis that document the needs of vastly different user populations, continue to document changing needs over time, coordinate investigation at multiple sites of use, design for large distributed entities, and absorb transformative changes in practice.”

4.3 *Building Successful Teams*

Related to this, we need a better understanding of the composition of teams involved in e-Science research, and the roles and activities of team members. Previous work has show e-Science users can fall into several categories: *end users* in the science domain, *technologists* who take generic software and alter it for domain-specific tool use, *builders* of generic software, and *support services*, which includes system administration. This workshop saw several projects, especially Discovery Net and myGrid, that were successful in part due to a complementary team that included different classes of end users, ranging from hybrids, who are able to carry out development work, to researchers, who don't do any programming but will provide their time for activities such as requirements capture and testing. This diversity of user types should also serve to caution developers against relying on only the most enthusiastic and committed as informants for requirements.

Beyond the immediate research team are system administration support, application analysis support, and other roles which together are constitutive of a potentially quite diverse array of local services provided by the hosting organisation. In other words, the work of making technologies work is often borne by a wide variety of players whose contributions and needs will have to be factored into any particular technical solutions [8,9]. When we say that we should be acquainted with the roles of research team members, we are also aware that to be useful and usable, tools must fit in with the broader research and support activities of the adopting organization.

4.4 *Agreement of Common Practices and Dissemination*

Many approaches for Web service interfaces were discussed, but there was a lack of agreement of common practices between approaches. Many specific implementations solved the same types of problems (job submission, interaction with security systems, file transfers, defining work to be done, tracking of work in progress, etc.), but each in a bespoke manner. Not all approaches even followed the basic standards available, thereby limiting interoperability as well.

Related to the need for common practices, a better understanding of the basic functionality needed by e-Science users would allow tool builders to make better design and implementation decisions. Simple tools are needed to provide the basic building blocks, but we still have problems with tool builders creating what's interesting for them – often bells and whistles that can complicate a system unnecessarily from a users' point of view. Unfortunately, this is also what funders ask for on occasion, so ongoing discussions for the need for basics that are well supported may be required with funding agencies as well.

In addition, this workshop revealed that there is a lot of duplicated effort which is wasteful and the attention being paid to user requirements both methodologically and practically varies considerably with no agreement on best practices. There is no

consistent pattern evident for the handling of usability issues, evaluation, or managing user involvement. Consequently, while the overall quality of the results is quite impressive, there is quite a lot of variability in evidence.

What this highlights is the absence of appropriate mechanisms for coordinating learning across individual e-Science projects, and for harvesting, promulgating, and embedding best practice (whether it be technical solutions or methodologies for design, development, and user involvement) in future activities. There is clearly a lot of learning going on but it is happening too informally for the e-Science programme as a whole to be able to exploit it. This poses a serious threat to the advancement of UK e-Science, especially as it now looks to widen its user engagement beyond the early adopters.

One follow on to the workshop is a discussion of how to better coordinate the individual e-Science project learning across the UK, in terms of technologies, design methodology, and user involvement. There is a need for coordination mechanisms and dissemination of these fundamentals. We need to examine if there are existing models and examples where this is being done already that we can borrow or adapt.

4.5 *Base Implementation*

If there were agreement on the basic functionality needed, we might be able to have a common base implementation, one of a number of issues explored by Breakout 2 (Commodification, Section 3.2). If this were available, then projects would have a solid starting point for a system that could be supported in a production manner, and made available to a wide set of users. This would also provide a means to embed and propagate lessons learned through the technologies themselves.

For a Web-based interface to be useful, we also need domain specific extensions to the common base system. Within a given application domain space, sets of extensions could be developed in a straightforward manner to allow additional functionality. A slightly different approach would be to use the idea of design patterns [10, 11] or portlets to identify function components that would potentially have cross-domain applicability but not necessarily be seen as relevant by each and every research community. Individual projects would be able to determine how any supplied set of functionality would integrate with other tools which researchers routinely use. This would allow users to ‘pick-n-mix’ to suit their needs.

Complementing the technical issues surrounding the common base system, we should consider support issues for this varied user base. As has been shown, without adequate support new functionality is rarely adopted. Issues to be considered include what kinds of skills are required, how quickly they can be assembled, and how they would be integrated within existing support arrangements.

4.6 *Use of Emerging Technology*

A number of talks raised issues about the use of Web browsers to access e-Science applications, and how these might be addressed by emerging technologies. There is a push to offer a rich user experience that matches the capabilities of common desktop applications while utilising the ubiquity of Web browsers to reduce costs of installation and, to some extent, training. Technologies emerging under the Web 2.0 banner, e.g. AJAX, SVG/Javascript, or ASP.NET, make it possible to develop applications that are more responsive than traditional Web applications and offer interaction styles such as drag-and-drop that are familiar to most users.

However, use of more recent and complex Web 2.0 technologies will likely introduce additional complexity, as highlighted by the eHTPX discussion (Sections 2.6 and 2.7). The question arises whether the resources needed to build such applications are well invested given that, in contrast to Google Maps or amazon.com, e-Science applications are not normally used by unknown, occasional users but rather by known users who rely on e-Science applications to do their daily work. RealityGrid discussions supported the view that there was a demand for doing one thing and doing it well, and perhaps time spent on new technology should be invested only cautiously.

There exists an ongoing trade-off for most groups between usability and time spent on improvements. Much project-based development is done by technologists (research associates) who will move on to pursue their academic careers and will stop being developers relatively quickly, so there is only very little time to make the learning pay off. The ^{my}Grid discussion noted that usability in the traditional sense is not always the most important goal, that there are situations where it is much better to take a pragmatic approach focused on getting the job done.

There are ways of minimising the effort involved in developing usable systems and reducing the learning effort that may be investigated. One example is the Eclipse Rich Client project, which provides a framework for rapidly developing user interfaces and prototypes which have a look and feel that will be similar to other such systems users may be familiar with, thereby making the interactions with an eResearch application easier to understand. Because technologies are developing rapidly, however, the trade-off between effort and pay-off can change quickly. It would be useful to have an appraisal of the available technical options including different Web technologies, rich client, and fat client approaches, their likely trajectories and the strategies available for making the best of each of them.

5 Conclusions

In general, the workshop and follow on discussions noted the following:

- There is a need for closer interaction between end users and tool builders over the full design, development, and use lifecycle
 - Real successes came through committed and extended engagement with and by real end users
 - Embedding users or developers in each others' groups, called hybrids, worked well for several projects
- There is a need for a better understanding of the makeup of successful e-Science project teams
 - Projects should include expertise sourced from the wider organisational environment
 - Support roles must be considered along with the more traditional roles
- Requirements gathering is complicated by different classes of users
 - User requirements will change depending on experience with the technology
- There is a lack of agreement about needed basic functionality and common practices
 - A mechanism is needed to communicate common approaches in terms of functionality, technology, or requirements gathering
- There may be a need for common base implementation
 - This should have basic functionality or design patterns that can be selected independently
 - It should be built on standards and supported to production quality levels
 - It should be capable of supporting domain specific extensions
 - Emerging technologies might allow for additional usability gains, but must be evaluated against possibly steep developer investment

It has been suggested that a series of smaller, focused meetings might help toward the design of a common system and better understanding of user needs. These meetings would be small, pragmatic and would concentrate on a single topic for the day. The number of participants would be limited to fewer than 10. Suggested topics include:

1. What is the makeup of a good team – from tool builder to end user?
2. What is the minimum functionality that needs to be offered to end users to make the NGS an attractive platform for a well defined set of application domains? Additional follow up meetings could expand the scope of this initial set.
3. Given (2), what tools exist to meet these needs, and what would need to be done to extend them to the NGS setting?
4. What are the most appropriate methodologies for design and development?

Additional topics to be addressed might include

- a. How to address the distinctive methodological challenges of e-Science projects.
- b. What is the right mix of skills?
- c. What are the on-going support costs?

- d. What are the socio-political aspects to be aware of?
5. Are there common lessons for developing Web-based interfaces that can be communicated more broadly, and how do we do this?

An additional suggestion was that one focus of the new OMII-UK funding within JISC could be on the integration, testing, documentation and support of a core set of portal functionality, based on the results of meetings 2 and 3 above, for the NGS.

Acknowledgments

This work was supported in part by the Joint Information Systems Committee (JISC) the National eSocial Science Centre (NCeSS) and the National e-Science Centre.

References

- [1] Greenbaum, J. and Kyng, M. (eds.) (1991). *Design at Work - Cooperative Design of Computer Systems*. Lawrence Erlbaum Associates Publishers, Hillsdale, NJ.
- [2] Williams, R., Stewart, J. and Slack, R. (2005). *Social learning in technological innovation: Experimenting with information and communication technologies*. Edward Elgar, Aldershot.
- [3] Hartswood, M., Procter, R., Rouchy, P., Rouncefield, M, Slack, R. and Voss, A. Co-realisation: Towards a Principled Synthesis of Ethnomethodology and Participatory Design. In Berg, M., Henriksen, D., Pors J. and Winthereik, B. (Eds.), special issue on Challenging Practice: Reflections on the Appropriateness of Fieldwork as Research Method in Information Systems Research, *Scandinavian Journal of Information Systems*, 14(2), p. 9-30, 2002.
- [4] Voss, A. (2006). *Co-realisation: A Radical Respecification of the Working Division of Labour in Systems Development*. Unpublished PhD Thesis, School of Informatics, University of Edinburgh.
- [5] Schopf, J.M. and Newhouse, S.J. (2007). Grid User Requirements – 2004: A perspective from the trenches. Invited paper, “Best of CLADE 2006”, special issue *Journal of Cluster Computing*, to appear.
- [6] Lawrence, K. (2006). *Walking the Tightrope: The Balancing Acts of a Large eResearch Project*. Jirotko, M., Procter, R., Rodden, R. and Bowker, G. (eds.) Special Issue on Collaboration and eResearch. *Journal of Computer-Supported Cooperative Work*.
- [7] Zimmerman, A. and Nardi, B. (2006). Whither or Whether HIC: Requirements Analysis for Multi-Sited, Multi-User Cyberinfrastructures. Position paper, Workshop on Usability Research Challenges for Cyberinfrastructure and Tools, ACM CHI Conference.
- [8] Bowers, J. (1994). The work to make a network work: studying CSCW in action. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*.
- [9] MacLean, et al. (1990). *User-Tailorable Systems: Pressing the Issue with Buttons*. CHI'90, Seattle, Washington, pp.175-182.
- [10] Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*, hardcover, 395 pages, Addison-Wesley. ISBN 0-201-63361-2.
- [11] Murray I. Cole. (1989). *Algorithmic skeletons: a structured approach to the management of parallel computation*. MIT Press & Pitman.