

SKUA project management plan (public version)

Norman Gray

30 June 2008

I Overview of Project

1 Background

Astronomy has been part of the UK's e-Science effort since its inception, the majority of this under the AstroGrid project. The focus of this effort, in the UK and within projects in at least 15 other countries, is the creation of a worldwide Virtual Observatory (VO), making astronomical data and applications easily available to astronomers regardless of their location and affiliation. The VO will, by defining and implementing standard interfaces, make it possible to access common resources from multiple applications. These resources are located via a globally distributed resource registry, which has been defined and working for over two years now.

The Virtual Observatory (VO) is a world-wide collaboration, supporting astronomical research through a network of projects to support data management, interoperability, portable workflows and common services. It is managed at the international level by the International Virtual Observatory Alliance (IVOA), acting as a standards body closely modelled on the W3C. The UK has a long-term leading role in the VO through the UK e-Science AstroGrid project, AstroGrid participation in the European VO Project, and the substantial UK investment in the European Southern Observatory (ESO), another Euro-VO partner. A primary focus of the various international VO projects is the continuing definition and maintenance of practical and internationally supported metadata describing archive data and web services; and one focus of the SKUA project is to add semantic value to the deployed VO metadata registries, aligned with ongoing VO efforts to develop ways of making these registries useful to astronomical applications.

The PIs are strongly connected with the VOs development plans, and are in a position to react quickly to, and support, the needs of VO application authors.

The VO has an existing distributed registry service, containing metadata about large numbers of resources, from organisations and institutions, to large-scale data archive services. This registry is deployed already, in the form of a network of database-backed services.

The global VO has long recognised both the necessity and the complexity of shared metadata, and has made substantial time and software investments in the VO registry network described above. It recognises, however, that the problem is not yet completely solved, and is moving towards semantic solutions compatible with the solutions in this JISC proposal. This proposal therefore represents an opportunity to give a JISC-funded project a leadership role in the design of a component crucial to the infrastructure of the UK, European and world-wide VO efforts.

2 Aims and objectives

The Semantic Web has, with startling speed, graduated from wild-eyed vision to deployable engineering. The goal of letting computers 'understand' has solidified into

Contents

I Overview of Project	1
1 Background <i>p1</i>	
2 Aims and objectives <i>p1</i>	
3 Overall approach <i>p2</i>	
4 Project outputs <i>p3</i>	
5 Project outcomes <i>p3</i>	
6 Stakeholder analysis <i>p4</i>	
7 Risk analysis <i>p4</i>	
8 Standards <i>p5</i>	
9 Technical development <i>p5</i>	
10 Intellectual property rights <i>p6</i>	
II Project Resources	6
11 Project partners <i>p6</i>	
12 Project management <i>p6</i>	
13 Programme support <i>p6</i>	
14 Budget <i>p7</i>	
III Project Planning	7
15 Workpackages <i>p7</i>	
16 Evaluation plan <i>p8</i>	
17 Quality plan <i>p8</i>	
18 Dissemination plan <i>p8</i>	
19 Exit and sustainability plans <i>p8</i>	
IV Appendices	8
A Budget <i>p8</i>	
B Development methodology: Extreme Programming <i>p9</i>	



established practice and competing implementations, so that now, with the bleeding edge moving off into yet more exotic directions, is the ideal time to bring the core technologies to practical application. Europe has a world-leading role in the worldwide Semantic Web (SW) community, the fruit of years of heavy EC investment in the technology. The SKUA project will embed this expertise in a UK project, thus disseminating it from the UK to the worldwide VO community, and within the UK to the other metadata projects supported by the JISC.

The SKUA Project (Semantic Knowledge Underpinning Astronomy) will implement a distributed architecture of semantically aware RDF stores. This ‘semantic layer’ will support a cluster of applications which will either directly support users in finding and recovering useful resources, or indirectly support them by supporting user-facing applications. Although the system we build will be specialised to astronomy, and proved by its interaction with, and eventual embedding within, the Virtual Observatory, the bulk of the semantic knowledge is localised in the RDF store, with the design goal that it could be replaced if desired by the analogous semantic knowledge of a different domain.

Specifically, the project will:

- develop the SKUA infrastructure using Semantic Web technologies, and deliver either a central service or user-installable desktop servers;
- validate the approach and APIs by developing new tools or adapting existing ones to add semantic annotation sharing; and
- in particular, we will produce a ‘Spacebook’ application, to support social networking between astronomers.

3 Overall approach

In order to achieve the planet-scale interoperability which the VO requires, we will build on emerging Semantic Web best practices and technologies.

Deployment and user buy-in will be our critical tasks. The PIs have a long and continuing involvement in the VO community, and so can lead this deployment and react quickly to user requirements. However, user acceptance can be encouraged by producing exemplar applications, which illustrate how the architecture can be used, and which are independently useful. We describe such applications here, which we will implement during the course of the project.

The applications we will develop include the following.

- Tagging resources and sharing bookmarks. The most basic use of the SKUA network, used by both of the applications below, and most immediately usable by existing user-facing applications, is to allow users to tag and bookmark resources on the web or within the VO, and share those tags with other users. There are two existing VO applications (VOExplorer and Paperscope) which have private tagging and sharing frameworks, demonstrating that the demand is present. We will add SKUA support to one or other of these two applications.
- Spacebook – a semantic VRE. As the name suggests, Spacebook has an interface and sharing model styled on the very successful social software application, Facebook. In the case of Spacebook, though, individuals will be able to create and share queries, workflows and assertions about VO resources, in addition to supporting a professional/social network. In this, Spacebook will be a type of Virtual Research Environment (VRE) with additional semantic functionality.
- Suggestions server. A continuing problem within the VO is that of browsing or searching the existing registries for resources of interest, since the obvious ways of doing this produce either too few, or far too many hits. We have preliminary designs for a ‘suggestions server’, acting as a web service, which would take



a list of one or more resources of interest, and return other sets of resources related to the initial ones by an open-ended set of algorithms, using server-side semantically rich algorithms.

Critical success factors:

- Have either or both of the VOExplorer and Paperscope authors adopt our technology, and thus indirectly proselytise on our behalf.
- Demonstrate a working Spacebook application, and associated users.
- There is support for a demonstrator suggestions-server already in the VOExplorer application: this sub-project will be successful if the associated developer decides that the service is sufficiently useful that this support should be enabled by default.

An overall success factor is to demonstrate that the infrastructure we build will be generic and usable enough that we can build these three applications without difficulty.

4 Project outputs

Outputs include the following.

1. SKUA architecture documentation, accompanying one or more services implementing this architecture. These may be in the form of online services (that is, as a multi-user service patterned after `del.icio.us`) or distributed service applications (that is, for installation by users or system administrators).
2. Implementation of Spacebook service.
3. Adaptations of existing applications, such as Paperscope and VOExplorer, exploiting the SKUA infrastructure.
4. Presentations at AstroGrid and IVOA workshops, and papers targeted at ADASS and ESWC conferences.
5. SKUA final report.
6. SKUA project workshop.

5 Project outcomes

Project outcomes include the following.

1. The project will facilitate JISC experience with semantic technologies, and participate in the cross-fertilisation of ideas between existing JISC semantic projects.
2. Enhancement of existing tools (within AstroGrid and the wider VO community) to work with semantic technologies, and a greater awareness of the possibilities of these technologies within the community.
3. JISC experience with an agile development methodology.

Stakeholder	Interest/Stake	Importance
Specific application authors	Enhancements to their product.	high
Broader developer community	Easy access to extra functionality.	medium
JISC	Experience of technology and methodology.	medium
Astrogrid (& other VO)	Functionality enhancements increase project visibility and success.	low
Leicester University	Success of AG project.	low
Application users	Functionality enhancements.	low

Table 1: Stakeholder analysis

Risk	Prob	Sev	Score	Mitigation
Query federation	3	3	9	simplify method
AstroGrid unfunded	3	3	9	identify other clients
Technology unsustainable	2	4	8	embed code in existing apps
No developer take-up	2	3	6	engage early
No user take-up	2	3	6	engage with developers
Personnel or requirements change	3	2	6	contract staff
Integration	1	5	5	run early trials

Table 2: Risk analysis (see Sect. 7)

6 Stakeholder analysis

For summary stakeholder analysis, see Table 1.

Somewhat paradoxically, we have given the eventual application users a rather low priority. This matches the high priority we have given the application developers, since this reflects our focus on providing infrastructure and services to support the latter, and thus relying on their UI skills, and pre-existing relationships with users, to pass the benefits on in the most user-friendly fashion.

The AstroGrid project, and the community of VO projects, will benefit from enhanced functionality for the VO as a whole – each application will potentially bring in users who are likely to then become users of the other applications in the VO. Their priority is low, however, since their stake is held indirectly by the application authors we are specifically targetting.

JISC's stake is the only one that is not a more-or-less direct consequence of the intended interest of the application authors.

7 Risk analysis

The project risks are summarised in Table 2.

Risk: Query federation difficult to implement (medium probability/medium impact). Mitigation: if true federation (decomposing and rewriting queries) proves more difficult than expected, then a simpler form – where SACs simply forward queries to their federation and combine the results – will provide most of the benefits. Conduct trials at outset of project, test existing technologies, identify potential solutions and engage with originating projects.

Risk: AstroGrid loses funding (medium/medium). The AstroGrid project is, at the time of writing (June 2008) waiting to hear about continuation funding for the third phase of the project. If this fails to appear, we will have lost our clearest 'customers'. However the project has fairly certain funding up until the end of 2008, so we can aim



to make best use of the relevant developers until then.

Risk: Cannot integrate semantic technology with VO (very low/very high). This is an integration project, combining existing technologies, all of which the project proposers are familiar with. Run early trials. Build on existing VOservice standards.

Risk: Lack of take-up of toolkit by astro-developers (low/medium). Engage with existing VO projects early and often. Build components useful to developers as well as applications. Fitting in with a larger project means that that project shares the work of technology evangelism.

Risk: Lack of take-up of applications by astronomers (low/medium). Build community of users to act as beta testers and promoters to other users. Enhance already deployed applications, working with their authors, so that we effectively have access to an already-existing user community. UI design of spacebook application is crucial, so concentrate resources and feedback on usability.

Risk: Technology not sustainable (low/high). Demonstrate to AstroGrid and other VO projects usefulness of technology, encouraging them to integrate it in their own code-bases. Show technology to e-Science projects in other elds. Engage with OMII for possible take-up. Produce a mixture of service and distributable server implementations.

Risk: Personnel changes or major requirements change (medium/medium-low). Because of the close relationship between this project and the AstroGrid and VOTech projects, to the extent that VO developers will work with early versions of the SKUA deliverables, it should be possible to contract staff from those projects in the event of unanticipated changes in staff or requirements (this has happened in fact, see Sect. 12 for more discussion).

8 Standards

The SKUA project expects to use the following standards or best practices.

RDF : a network of W3C Recommendations for supporting the integration and exchange of knowledge on the Web <http://www.w3.org/RDF/>.

SPARQL : a W3C Recommendation for making structured queries to RDF triple-stores <http://www.w3.org/TR/rdf-sparql-query/>.

Agile methodologies : a set of best practices for managing projects (most typically software design projects) in which the design and, to some extent, goals are not fixed at the start of the project, but are instead developed and delivered in relatively short iterations through the life of the project (see Sect. 9 and Sect. B).

Linking Open Data : a set of best practices for linking semantically enhanced data, via RDF, on the Web <http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>.

9 Technical development

The project will use an agile development methodology. In such a methodology, the project outcomes and design are specified beforehand only in broad detail, and the detailed planning and development is instead framed in terms of shorter-period *iterations* of, in our case, three months.

In an agile methodology, the project aims to produce a product with at least some basic functionality as quickly as possible, and make it available to users immediately (in our case, the users are VO application authors). Then, at the beginning of each iteration, the project identifies which functionality it could add next, based on specific user-stories. It then selects those features which can be reasonably added in a single iteration, implements those, and produces another release at the end of the iteration. The advantages of this methodology are as follows.

Personnel	contact	% effort
Norman Gray, Leicester (project manager)	http://nxg.me.uk	50%
Tony Linde, Leicester	tony.linde@leicester.ac.uk	33%
Kona Andrews, ROE	http://www.thingwright.com	50% × 12mth

Table 3: Project personnel

- There is a functioning product at almost all times.
- Therefore the product can be confronted with users early and often, while there is still time to amend the design and interface. User feedback and deployment experience influence the design of the project at all stages, and faulty architectural decisions can be identified early and fixed in time.
- When planning each iteration, the project can use the difference between the expected and actual implementation times of the previous iteration, and thus improve its time estimates for the next iteration.

The SKUA project will use a variant of the eXtreme Programming (XP) methodology adapted to a distributed development process (thanks to Neil Chue Hong and Ross Gardler of OMII-UK for guidance here). See Sect. B for a more detailed discussion of the methodology we will use.

10 Intellectual property rights

The project IPR will be owned by the University of Leicester. All software products will be released either under one of the BSD licences, or under the Apache licence (the precise licence will be decided by the project members soon after the project starts).

II Project Resources

11 Project partners

The project is based at the University of Leicester. The main contact is the project manager, Norman Gray <http://nxg.me.uk>.

The project is subcontracting 50% of the time of a developer from the AstroGrid project, based at Royal Observatory Edinburgh (see Sect. 12).

12 Project management

The project team is as shown in Table 3.

The project experienced some difficulties in recruiting for the developer post, with a two-month work-permit delay in appointing a developer who withdrew their application at the last minute, but by the beginning of May 2008 it had subcontracted a developer from the AstroGrid project, based at Royal Observatory Edinburgh (see Sect. 12). The initial agreement is for the project to buy 50% of the developer for 12 months, but depending on needs and availability, either the fraction or the duration may be increased. The subcontracted developer is at a higher grade than initially anticipated, but they are more experienced and will work for a shorter time, so while we expect the pattern of spend on this item to be different from that planned, the total will likely remain the same..

13 Programme support

The programme manager has arranged a one-day workshop, in June, for the three projects funded in this round which are working with semantic technologies.



14 Budget

The budget has been redacted from this version.

The project was unable to recruit the developer as planned – see Sect. 12 for discussion. Rather than recruit a 100% developer for the entire length of the project, we have been forced to subcontract 50% of a developer, at a higher grade, from the Royal Observatory Edinburgh (ROE); thus the spend on personnel will be different from that estimated in the budget.

III Project Planning

15 Workpackages

The project work-packages are as follows. The times T_n indicate the month numbers in which we expect the corresponding development to be concentrated.

The XP process gains much of its strength from its concentration on short-term goals. There is therefore the danger that it will lose track of long-term requirements, and it is for this reason that the process makes use of an explicit statement of ‘vision’. We believe the following deliverables are represented in the project’s ‘vision’.¹

Work Package WP1 Project management

15.1 Phase 1: Architectural elaboration (T0-T12)

This is a long thin phase. After an initial block of activity, an initial version of the SAC network will be quickly delivered to the case study developers, and effort shift to evolving it in the light of emergent requirements. This agile methodology ensures that applications work can start very promptly, and that the services and API closely match emergent tool requirements.

WP2 Implementation of SAC nodes and network

Deliverable D2.1 Set up RDF version of VO registry

D2.2 Implement SAC, building on well-established triplestore implementation

D2.3 Develop thin user-facing client for simple SAC management and setup

D2.4 Produce initial API documentation for client authors

15.2 Phase 2: Case-study implementation (T3-T16)

The case-studies will be implemented in parallel.

WP3 Spacebook application

D3.1 Develop spacebook application, and refine interface through regular releases

WP4 Suggestion service

D4.1 Develop core service, with a plugin architecture

D4.2 Implement plugins implementing various heuristics

15.3 Phase 3: Dissemination (T8-T18)

The dissemination activity will be more-or-less continuous, because of the continuous engagement of the PIs in the target developer community, and participation in project strategy meetings.

WP5 Documentation and dissemination

D5.1 Papers for appropriate semantic web and astronomy conferences

D5.2 Contributions to astronomy-specific publications (in particular IVOA Standards)

D5.3 SKUA final report, including discussion of applications outside astronomy

D5.4 Organise a workshop on the project outcomes; JISC dissemination/evaluation

¹<http://code.google.com/p/skua/wiki/Hallucination>

16 Evaluation plan

The agile development methodology we will use in the project (see Sect. 9) has as a core feature a tight coupling between ‘customers’ specification, code design, and its implementation, with the project ideally going completely round this cycle, and making a release, every iteration (two weeks, in our case). In this methodology the ‘customers’ are directly involved in the identification and prioritisation of features, and the resulting implementation is in principle never more than an iteration away from being evaluated.

While we may fall away from this ideal in practice – with three part-time developers, we are smaller than a standard XP team, and we have no budget to involve a set of ‘customers’ as intimately as a standard XP process suggests – our use of iterations and frequent releases mean that our implementation will be confronted with its intended users as early and as often as practical, with the goal of having those users indicate the future short- and medium-term direction of the project.

In this way, the project’s evaluation plan and its quality plan are coterminous: by having the project’s users continually evaluate the project’s outputs, the project’s quality is being continually monitored. Further to this, the project’s ‘vision’ statement (see Sect. 15) provides the link to long-term direction, and articulates explicit success criteria.

17 Quality plan

See Sect. 16.

18 Dissemination plan

Integration with target codebases The project team has access to the AstroGrid codebase, and good relations with that developer community. We will work with selected developers in the hope that support for the SKUA infrastructure can be integrated in those applications, if necessary doing the extension work ourselves. Since these developers are to some extent our ‘customers’, we will consult with them early in the project.

Publication of IVOA Note The IVOA is a technical standards body for the astronomical Virtual Observatory. Towards the end of the project, when there is a coherent software product, we will describe the SKUA work in an IVOA Note, included as part of the IVOA’s document series.

Workshop presentations Both AstroGrid and the IVOA have regular informal technical workshops. Once the infrastructure is mature enough that it is useful to developers not intimately associated with the project, we will offer presentations and tutorials at these meetings.

19 Exit and sustainability plans

All of the project materials are, and will remain, available in the subversion repository at <http://code.google.com/p/skua/source/>.

It is an explicit goal of the project that its products be attractive enough to be incorporated into existing applications, of which we have identified the Paperscope and VOExplorer applications developed by the AstroGrid community.

IV Appendices

A Budget

See Table 4.



The total budget requested from, and awarded by, JISC was £310752. The detailed budget has, however, been redacted from this version.

Table 4: SKUA budget (all figures in £)

B Development methodology: Extreme Programming

The following describes and, we hope, justifies the agile process we intend to follow in the SKUA project. Of necessity, we are adjusting the process to suit our situation, rather experimentally, and we expect to modify the process further in the light of our continuing experience.

The methodology is discussed at greater length at <http://code.google.com/p/skua/wiki/Process>. We expect that document to continue to evolve, and the consensus described there will supersede the account here.

B.1 Background

SKUA development will use an agile development methodology, rather than a heavy-weight process of specifying requirements, architecture, design, implementation, testing, and finally discovering that what works isn't what's wanted, and what was wanted doesn't work.

Agile methodologies in general aim for a very tight coupling between requirements, implementation and release, cycling round the three in iterations as short as a week. They are characterised by having very little up-front design, and having a functional and releasable system at all times, which is incrementally expanded in the light of direct 'customer' requirements.

Specifically, we plan to use a variant of the Extreme Programming (XP) methodology, building on Ross Gardler and Neil Chue Hong's first attempt at an XP variant for distributed teams², James Shore and Shane Warden's *The Art of Agile Development*³, and Kent Beck's original *Extreme Programming Explained*⁴.

The reason for using a variant is twofold. Firstly, XP methods are typically described as working for teams of between four and 20 programmers, while we have two or three times 50% FTE; and because a notable feature of XP is its emphasis on co-location, even to the extent of suggesting that all actual code is written by pairs of programmers sharing a single keyboard.

One cannot simply delete XP practices at random, however. The set of XP practices form a principled and coherent whole, and if we remove one practice, we must aim to replace it with one which has the same purpose. An important group of the XP practices which presume co-location are actually concerned primarily with communication, both between the members of the team and communicating the status of the project in a very immediate way.

B.2 Our process

The following are XP practices which are both clearly useful to us, and adaptable to a distributed team. We expect that both the list, and the way we adapt them to our situation, will change over the course of the project.

Customers XP relies on the notion of a 'customer', who takes an active role in the development process. The 'customer' is a representative of the person or entity who is going to receive the value, or benefit, of the delivered product. Their role is to act as a walking, talking, requirements document, ready to generate or elaborate

²<http://www.slideshare.net/rgardler/agile-and-open-development/>

³O'Reilly, 2008, ISBN: 0596527675

⁴Addison-Wesley, 2004, ISBN: 0321278658

stories, decide on priorities, and inform the programmers about how they wish or expect to use the final product. At present (June 2008), we have not made any provision for identifying such customers explicitly, though we have two applications in mind, written by AstroGrid colleagues (Paperscope and VODesktop) which we hope will be users of the SKUA product, and whose developers we aim to involve in the project in some way. Until this happens, we are going to have to be ‘customers’ ourselves.

User stories One of the core XP notions is the notion of *user stories*, which are brief accounts of discrete blocks of functionality, described from the point of view of a user of the system, and small enough that they can be implemented in one or two days work. We maintain a wiki list of such stories, from which we select a set to implement at the regular iteration meeting.

The *planning game* is a precursor to the iteration planning meeting. At base, programmers and customers work through the list of extant stories, with the customers prioritising unimplemented stories, and programmers estimating the effort required to implement.

Iterations The other core XP notion is the *iteration*. At the beginning of an iteration, the group selects a set of *user stories* which they plan to implement in that iteration. At the end, they review what has been completed, comparing actual to expected progress, and release the improved software. That is, the software is re-releasable at the end of every iteration, and is released in fact on release dates decided well in advance.

XP suggests iterations of one or two weeks. Since the SKUA programmers are generally working only 50%, we felt that two-week iterations would be best.

We expect to use Skype to have iteration meetings, but will experiment to see how well this will work, or if other technologies can help.

The iteration planning meeting consists of:

1. Retrospective of previous iteration (primarily consisting of a demo, plus a count of the number of stories completed).
2. Select stories to include in next iteration. Based on the programmers’ estimates of required effort, the meeting selects a number of stories whose effort sums to the effort represented by the set of stories completed in the last iteration.

Informative workplace As part of its group of communication practices, XP promotes the notion of the *informative workplace*. Here, the status of the project – in terms of the stories pending and committed to for an iteration, planned release dates, the group’s velocity (the number of stories usually completed per iteration), and perhaps outstanding bugs – are made as immediately visible as possible. XP evangelists are very fond of the whiteboard; it seems likely that some combination of a wiki page and Google Docs would be able to do the same work.

As an initial attempt at this, we will maintain a wiki page which shows where we believe we are at any point.

Stand-up meetings A common XP practice – part of the group focused on communication – is the *stand-up meeting*. This is a daily meeting in which each participant reports very briefly – taking around 30 seconds – on what they got done yesterday, what they plan to do today, and what problems are blocking them. Like the *informative workplace* practice, this is intended to let status information percolate through the group, making it possible for all members to contribute to solving problems.

Although physical proximity is, again, one of the key components of this practice in the XP methodology, we hope to get much of the relevant benefit from regularly scheduled brief Skype conversations.