GGF11 Semantic Grid Applications Workshop

Honolulu

June 10, 2004

Editors:	Danius Michaelides
	Luc Moreau
Coordinators:	David De Roure
	Simon Cox
	Geoffrey Fox
	Carole Goble

Workshop organizers

The workshop is being held by the GGF Semantic Grid Research Group (SEM-RG) in conjunction with the Applications and Testbeds Working Group (APPS-RG). The Workshop Co-Chairs are:

David De Roure (Chair)	Semantic Grid Research Group
Geoffrey Fox	Grid Computing Environments and Semantic Grid Research Groups
Carole Goble	Semantic Grid Research Group
Simon J. Cox	Applications Working Group

Proceedings

Luc Moreau (proceedings chair)	University of Southampton, UK
Danius Michaelides (submission chair)	University of Southampton, UK

Program Committee

Mark Baker	University of Portsmouth, UK
Jim Blythe	Information Sciences Institute, USC
William Johnston	Lawrence Berkeley National Laboratory
Kerstin Kleese	CLRC Daresbury Laboratory, UK
Libby Miller	ILRT, University of Bristol, UK
Jim Myers	Pacific Northwest National Laboratory
Marlon Pierce	Indiana University

Table of Contents

Foreword to the GGF11 Semantic Grid Applications Workshop

David De Roure

University of Southampton, UK

Fundamentally, Grid computing is about bringing resources together in order to achieve something that was not possible before. In the early days there was an emphasis on combining resources in pursuit of computational power and very large scale data processing, such as high speed wide area networking of supercomputers and clusters — a view caricatured now as 'big iron and fat pipes'. As Grid computing has evolved it continues to be about bringing resources together but the emphasis has shifted to the notion of Virtual Organizations, defined by Foster et al in [1]:

> The real and specific problem that underlies the Grid concept is coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations. The sharing that we are concerned with is not primarily file exchange but rather direct access to computers, software, data, and other resources, as is required by a range of collaborative problem-solving and resource brokering strategies emerging in industry, science, and engineering.

In 2001 a group of researchers recognized that this vision of the Grid is closely related to that of the Semantic Web — which is also, fundamentally, about joining things up. The value of applying Semantic Web technologies to the information and knowledge in Grid applications was immediately apparent, especially in fields which were already going down this route such as life sciences. In 2001 we already anticipated the service-oriented world that was soon to follow in the form of the Open Grid Services Architecture, and we also saw the potential of the Semantic Web in working with service descriptions. Hence we knew Semantic Web technologies were useful upon the Grid infrastructure but also within it, directly addressing the 'Grid problem' through describing resources and services.

Over three years we have been building a bridge between these research communities. The report 'The Semantic Grid: A Future e-Science Infrastructure' [2] was influential in the UK e-Science program, initially as a samizdat publication mid-2001. This was followed in 2002 by bringing people together at the 11th International World Wide Web Conference and at the 1st International Semantic Web Conference, with papers reaching out into different communities [3, 4] and a series of talks at various international events. Through the vision of researchers at the intersection of these communities, a series of Semantic Grid research projects have been launched. Two years after the publication of the original Semantic Grid report, the IEEE Intelligent Systems special issue on e-Science [5], published in January 2004, reported on Semantic Grid activities across four continents, and a chapter in The Grid 2 was dedicated to 'Enhancing Services and Applications with Knowledge and Semantics' [6].

The Semantic Grid vision brings a challenging research agenda but also a promise of immediate practical benefit through deployment of available Semantic Web technologies in Grid applications currently under development. For practitioners outside the Semantic Web community it is important to understand what can be achieved immediately and what is a research issue. This is the thrust of the Global Grid Forum Semantic Grid Research Group which was created in November 2002, after successful 'Birds of Feather' sessions at GGF5 in Edinburgh and GGF6 in Chicago. We held our first workshop in October 2003 at GGF9 in Chicago, consisting of invited papers from a selection of leading Semantic Grid activities.

For the GGF11 workshop we have very deliberately taken an applications focus, and we are pleased to hold this event jointly with the GGF Applications and Testbeds Working Group. The Semantic Grid Applications workshop is one of three events this summer promoted by the Semantic Grid Research Group — the others are the Semantics of Grid and Peer-to-Peer Computing workshop at WWW2004, and the ECAI workshop on Semantic Intelligent Middleware for the Web and Grid. All these events have had open calls for papers and we are pleased to see the community growing.

There are many challenges ahead — some of them are set out in [7], where we remember the 'Web' in Semantic Web and the benefits of the network effect and distributed working. There are also many bridges under construction — to established communities such as Agent Based Computing but also new communities like Semantic Web Services and Ubiquitous Computing. We are also seeing increasing interest in Semantic Grid from the broadening range of disciplines turning their attention to Grid computing, notably Arts, Humanities and Social Sciences.

Thanks to everyone involved in Semantic Grid activities. The Grid, the Semantic Web and now the Semantic Grid are all about joining things up. We hope that joining people together in this workshop is another step towards achieving the vision.

References

1. Foster, I., Kesselman, C. Tuecke, S. (2001), The Anatomy of the Grid: Enabling Scalable Virtual Organizations, International Journal of Supercomputer Applications, 15(3).
2. De Roure, D., Jennings, N.R. and Shadbolt, N. R. (2001) Research Agenda for the Semantic Grid: A Future e-Science Infrastructure. Technical report UKeS-2002-02, UK e-Science Technical Report Series, National e-Science Centre, Edinburgh, UK. Also appears in Berman, F., Fox, G. and Hey, A. J. G., Eds. Grid Computing - Making the Global Infrastructure a Reality, 437-470. John Wiley and Sons Ltd.
3. Goble, C.A. and De Roure, D. (2002) The Grid: an Application of the Semantic Web. ACM SIGMOD Record 31(4): 65-70.
4. Goble, C. and De Roure, D. (2002). The Semantic Web and Grid Computing, in Kashyap, V. Ed. Real World Semantic Web Applications, IOS Press.
5. De Roure, D.C., Gil, Y. and Hendler, J.A. eds. (2004) IEEE Intelligent Systems, Special Issue on E-Science, Volume 19, Issue 1 (January/February).
6. Goble, C.A., De Roure, D., Shadbolt, and Fernandes, A.A.A. (2004) Enhancing Services and Applications with Knowledge and Semantics in Foster, I. and Kesselman, C., Eds. The Grid 2: Blueprint for a New Computing Infrastructure (2nd edition), Morgan-Kaufmann.
7. De Roure, D. and Hendler, J.A. (2004). E-Science: the Grid and the Semantic Web. IEEE Intelligent Systems, 19(1), 65-71.

Designing Ontologies and Distributed Resource Discovery Services for an Earthquake Simulation Grid

Mehmet S. Aktas[1,2], Marlon Pierce[2], and Geoffrey C. Fox[1,2]

[1] Indiana University, Computer Science Department
Lindley Hall, Room 215 150 S. Woodlawn Ave., Bloomington, IN, USA 47405
[2] Indiana University, Community Grids Labs.
501 N. Morton Street, Suite 222, Bloomington, IN, USA 47404
{maktas, marpierc, gcf}@indiana.edu

Abstract. Locating resources is a fundamental problem within large scale distributed environments. An effective way of solving this problem is providing and managing metadata about resources. In this paper, we describe an initial ontology design and the architecture of a distributed information system that we are implementing. We introduce an efficient notification based caching mechanism to make metadata available in open hypermedia peer-to-peer systems.

1 Introduction

This paper describes our initial work designing and developing a metadata management and discovery system using Semantic Web [1] technologies, focusing particularly on the problems and needs of the SERVOGrid (Solid Earth Research Virtual Observatory Grid) [2–4] project. SERVOGrid integrates measured and calculated earthquake data (such as fault models, seismic events, GPS measurements) with simulation codes, all of which are accessed through various Web Services. SERVOGrid resources are located at various institutions across the country, with growing interest in participation from international ACES [5] partners.

SERVOGrid requires Grid services for remote code execution and data access, as in traditional science Grids. However, our challenges also include modeling and managing metadata information generated by the Grid services. We must also assume that resources are volatile and not suitable for centralized management: Grid resources tend to evolve over time, may include very distributed partners, and may be temporarily inaccessible. Thus information management is crucial.

This paper introduces our efforts to represent and facilitate the use of metadata in distributed systems using Semantic Web technologies. The Semantic Web adopts the Open Hypermedia [6] model. In this model, resource information, that points users to resources, is indicated in separate documents. This information is the semantic metadata. We consider the SERVOGrid environment as an open hypermedia peer-to-peer system. In this paper, we describe our initial ontology design and the architecture of a distributed publish/subscribe system for managing metadata discovery and access in the SERVOGrid environment.

2 SERVOGrid Resources Ontology

In this section, we describe our effort to create ontology aided metadata instances for SERVOGrid resources. The SERVOGrid project has a collection of codes, visualization tools, computing resources, and data sets that are distributed across the grids. Instances of a well-defined ontology will be used to describe specific resources as metadata. We outline the steps of creating a SERVOGrid ontology as follows.

Defining Classes. The first step is to group together related resources in order to create an ontology. There are three major groups of SERVOGrid resources that need to be classified. These groups are ServoCodes, such as simulation and visualization codes, ServoData, such as fault and GPS data, and ComputingPlatforms, such as computers and web services. We observe the following hierarchical classification of classes to group together SERVOGrid resources in Figure 1.

For lack of space, the full descriptions of the ontology classes are not given here but they can be found from the preliminary schemas available at [7].

Defining Properties. Ontology classes are created to group together similar resources. Since the class hierarchy does not give any information about the classes themselves, we defined various meaningful properties for each class of the ontology. In addition to these properties, we have also used DC (Dublin Core) [9] and vCard [10] metadata definition standards in our ontology. In Figure 2, we sketch the properties that link SERVOGrid Resources. The detailed descriptions of the properties are available at [7].

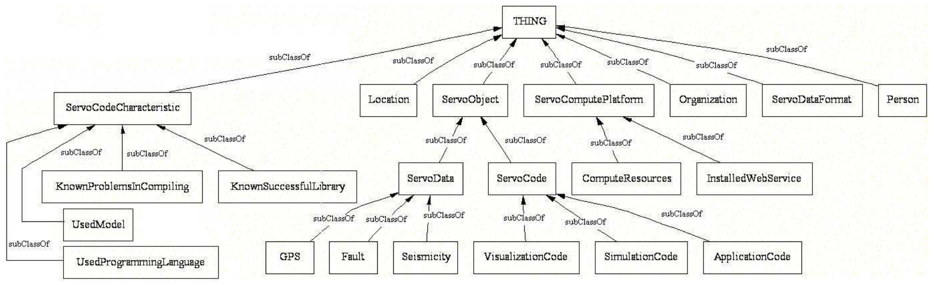


Fig. 1. Class Hierarchy of SERVOGrid Ontology

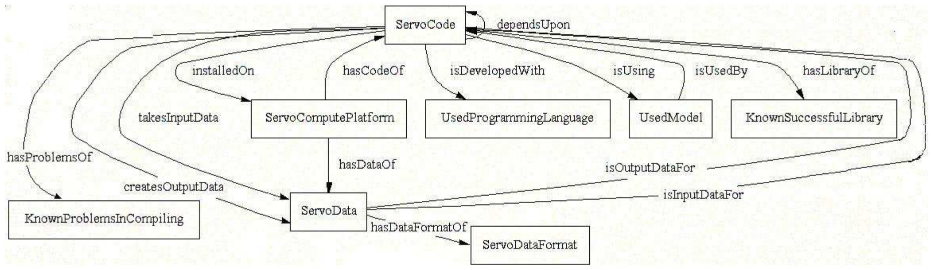


Fig. 2. Properties of SERVOGrid Ontology

Generating the Ontology and Metadata Instances. As a next step, the description of these classes and properties are written in the semantic markup languages RDF/RDFS [11] and OWL [12]. The full ontology and examples of the metadata instances are available from [7, 8].

As mentioned above, a decentralized approach to metadata management is desirable. Decentralization of metadata management, however, leads to potentially low search performance than centralized approaches. To improve response time and lower network traffic, we introduce a caching mechanism which is based on notification-based publisher/subscriber interactions. We discuss the details of our proposal in the following section.

3 Notification based caching approach in managing metadata

Our system is designed to run as a standalone Java program on distributed servers. These servers might accommodate a metadata repository or might be interacting with a separate database repository through Web Services. The principal feature of the system is that after an initialization phase achieved through crawling, it establishes updatable caches that subscribe to remote resources and listen for updates. We introduce proxy caching to distribute the copies of the metadata across the peers in the network. When a user posts a query to a proxy cache, starting from this cache, at each node, the cache is queried for the requested metadata and then the query is passed to nodes that are semantically connected. Results are returned to the proxy cache that initiated the query and the proxy cache returns results to the client. At each step, each cache relays the query only to the nodes that it is aware of. We use breadth first search to simply explore the next nodes if the requested metadata is not already cached.

When using cached copies across the nodes of the network, information consistency becomes an issue. To avoid information inconsistency, we utilize a notification based interaction pattern between the source cache and other caches where remote copies exist. The caching system is based on a notification-based brokering system such as NaradaBrokering [13]. In this scenario, each server interacts with the NaradaBrokering system. Our system is designed to work with one or multiple NaradaBrokers. Each server can be both a subscriber and/or publisher to different topics. Subscribers listen to updates of the metadata instances, so that the cached local copy remains consistent with the original metadata. Publishers are responsible for publishing any changes in the metadata to the related topics to keep the cached copies up-to-date. In this scheme, RDF predicates serve as topics. Subscriber nodes must subscribe to predicate topics to receive the updates for the resources that are in the range of that predicate. We can illustrate this in the following example which is a triple regarding a simulation code “disloc.c”:

Subject:	http://www.servogrid.org/servo/instances/servoCode#disloc_instance
Predicate:	servo:installedOn
Object:	http://www.servogrid.org/servo/instances/servoPlatform#grids_instance

In this example, the topic would be “servo:installedOn”. Remote cache copies that carry the instances of the “grids_instance” metadata (e.g. set of triples describing the grids instance) would listen to the topic “servo:installedOn” for any updates. The origin cache for the “disloc_instance” resource metadata publishes any changes to the “servo:installedOn” topic to keep the remote copies up-to-date. The predicates are uniquely defined by the ontology. The range of each predicate is defined based on the SERVOGrid ontology. This provides an automatic way of subscribing to topics (since we know which resources can subscribe which topic), that is simple to implement.

4 Related Work

Finding resources by stepwise exploration has been first studied in the hypertext community [14][15]. The Lagoon Cache [16] is similar to our approach, using the proxy caching method to improve search performance and bandwidth consumption. Our main difference is that we utilize the idea of "caching by enforcement", meaning the resource provider is expected to propagate the updates to all remote cached copies (by using a notification system) to keep the remote copies up-to-date. The Globus Alliance [17] has released WSRF Specifications [18] which include WS-Notifications [19]. We find the following similarities with our approach. Both approaches give the responsibility of propagating the updates to the service providers. Also, both approaches have notification based publish/subscribe interactions to notify the corresponding nodes regarding updates. There are differences however. WS-Notification Specifications define the topics with XML syntax and the specifications do not define any rules or restrictions regarding becoming a publisher or a subscriber to a topic. We use RDF syntax for the topics and define each topic as predicate of a triple. The SERVOGrid ontology defines the rules regarding which resources can subscribe which topics.

5 Conclusions and Future Work

This paper has described the ontology development work for describing SERVOGrid resources. We are coupling this information representation to a notification-based caching system that will allow stepwise exploration of metadata instances by using Semantic Web technologies. In future work, we plan to further study fast and efficient ways of exploring open hypermedia peer-to-peer systems for the SERVOGrid project. Our goal is to provide an efficient metadata management system where semantic metadata is used to describe resources in a distributed fashion.

Acknowledgement. The work for this project was funded through NASA Ames Research Center through Grant 1247333. The SERVOGrid project is funded through the NASA Computational Technologies program under Grant 1253656.

References

1. W3C Semantic Web Site: http://www.w3.org/2001/sw
2. Choonhan Youn, Marlon Pierce, and Geoffrey Fox: Building Problem Solving Environments with Application Web Service Toolkits, ICCS03 Australia June 2003
3. Marlon Pierce, Choonhan Youn, Ozgur Balsoy, Geoffrey Fox, Steve Mock, and Kurt Mueller: Interoperable Web Services for Computational Portals. SC02 2002
4. Marlon Pierce, Choonhan Youn, and Geoffrey Fox: Interacting Data Services for Distributed Earthquake Modeling. ACES Workshop at ICCS June 2003 Australia
5. ACES - APEC Cooperation for Earthquake Simulation Web Site: http://www.aces.org.au.
6. Wiil, U.K.: Open Hypermedia: Systems, Interoperability and Standards: Journal of Digital Information, Vol.1, No.2 (1997)
7. Link to the SERVOGrid ontologies: http://ripvanwinkle.ucs.indiana.edu:4780/examples/download/schema
8. Link to the SERVO ontology instances: http://ripvanwinkle.ucs.indiana.edu:4780/examples/download/data
9. Dublin Core Meta-data Initiative Web Site: http://dublincore.org/documents/dces
10. The vCard version 3.0 Specifications: http://www.ietf.org/rfc/rfc2426.txt
11. W3C - Resource Description Framework Site: http://www.w3.org/RDF
12. W3C Web Ontology Language Overview: http://www.w3.org/TR/2004/RECowlfeatures20040210
13. NaradaBrokering Project: http://www.naradabrokering.org
14. P. De Bra, R. Post: Searching for Arbitrary Information in the WWW: the Fish-Search for Mosaic, WWW Conference, 1994.
15. Michael Hersovici, et al.: The Shark Search Algorithm. An application: Tailored Web site mapping: In WWW7, 1998.
16. P. De Bra, R. Post: Information retrieval in the World Wide Web: Making client-based searching feasible: Proceedings of the 1st WWW Conference, Geneva, 94.
17. Globus Project Website: http://www.globus.org
18. Czajkowski K., et al.: The WS-Resource Framework, http://www106.ibm.com/developerworks/library/wsresource/wswsrf.pdf
19. Graham S., et al.: Publish-Subscribe Notification for Web Services, http://www-106.ibm.com/developerworks/library/ws-pubsub/WS-PubSub.pdf

Collaborative Tools in the Semantic Grid

Michelle Bachler[1], Simon Buckingham Shum[1], Jessica Chen-Burger[2], Jeff Dalton[2], David De Roure[3], Marc Eisenstadt[1], Jiri Komzak[1], Danius Michaelides[3], Kevin Page[3], Stephen Potter[2], Nigel Shadbolt[3], and Austin Tate[2]

[1] KMI, The Open University, UK
[2] AIAI, University of Edinburgh, UK
[3] ECS, University of Southampton, UK

Abstract. The CoAKTinG project aims to advance the state of the art in collaborative mediated spaces for distributed e-Science. The project is integrating several knowledge based and hypertext tools into existing collaborative environments, and through use of a shared ontology to exchange structure, promotes enhanced process tracking and navigation of resources before, after, and while a meeting occurs. This paper provides an overview of the CoAKTinG tools, the ontology that connects them, and current research activities.

1 Introduction

The CoAKTinG project[1] aims to advance the state of the art in collaborative mediated spaces for distributed e-Science through the novel application of advanced knowledge technologies. It comprises four tools: instant messaging and presence notification (BuddySpace), graphical meeting and group memory capture (Compendium), intelligent 'to- do' lists (Process Panels) and meeting capture and replay. These are integrated into existing collaborative environments (such as the Access Grid [2]), and through use of a shared ontology to exchange structure, promotes enhanced process tracking and navigation of resources before, after, and while a meeting occurs.

Section 2 provides an overview of the tools, Section 3 describes the ontology that interconnects them, and Section 4 gives a glimpse of current work using the tools.

2 Tools

2.1 Buddyspace

BuddySpace is an Instant Message client (based on the Jabber protocol) with features that enhance presence awareness. Specifically, it introduces the graphical visualisation of people and the presence on a image or map, as can bee seen in the figure. This allows for multiple views of collaborative workgroups and the immediacy or "at a glance" nature gives users a snapshot of a virtual organisation. In a meeting, the instant message capabilities of Buddyspace naturally provide a "back-channel" to the meeting, for example, conveying URLs of documents discussed or as a non-disrupting communication. For distributed meetings, such Access Grid meetings, the presence of individuals gives an extra indication of co-location (especially if the videoconferencing technology is failing). The back-channel can also be used for meeting control tasks, such as queuing of speakers and voting on issues.

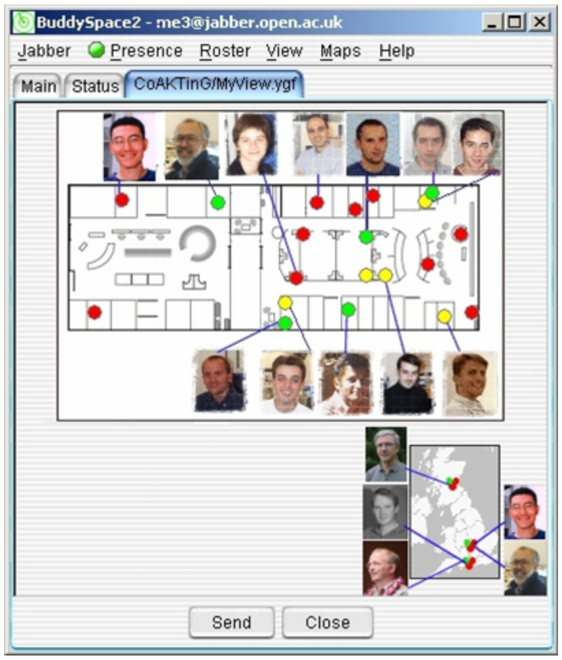


Fig. 1. Buddyspace showing a virtual organisation and presence indicators

For meeting capture purposes, logs of the channel conversations are made. Individual messages are timestamped and possibly examined to see if they control meeting specific messages.

2.2 Compendium

Compendium, first developed in 1993 as an approach to aid cross-functional business process redesign (BPR) teams, has been applied in several dozen projects in both industry and academic settings [3]. Its origins lie in the problem of creating shared understanding between the team members, typical of those attending teams working over weeks or months to design business processes: keeping track of the plethora of ideas, issues, and conceptual interrelationships without needing to sift through piles of easel sheets, surfacing and tracking design rationale, and staying on track and "bought-in" to the project's overall structure and goals [4]. The key feature of the early approach was the combination of an Issue-Based Information System (IBIS) concept-mapping tool [5], which supported informal and exploratory conversation and facilitation, with a structured modelling approach [6]. This allowed teams to move along the spectra of *formal to informal representation*, and *prescribed to spontaneous approaches*, as their needs dictated. It also let them incrementally formalise data [7] over the life of the project. As the approach was tested and refined over the course of several years, additional modelling methods were added, plus tools to transform Compendium's hypertext models into established organisational document forms, and vice-versa [8].

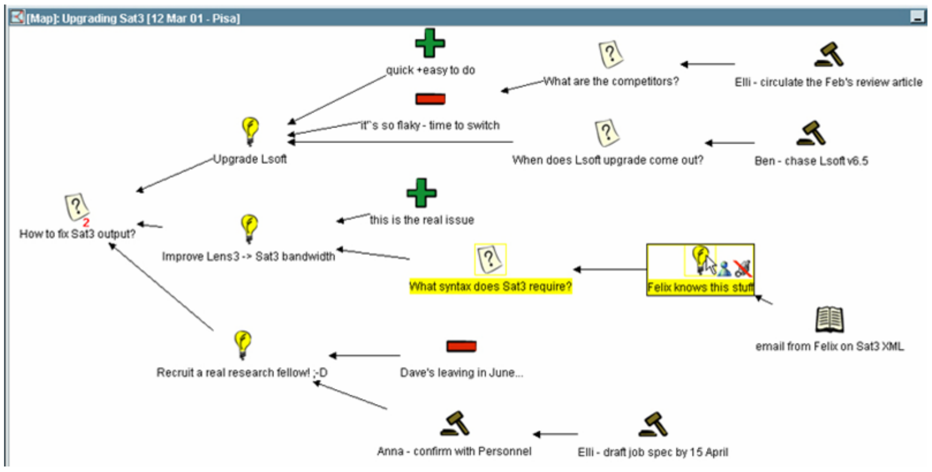
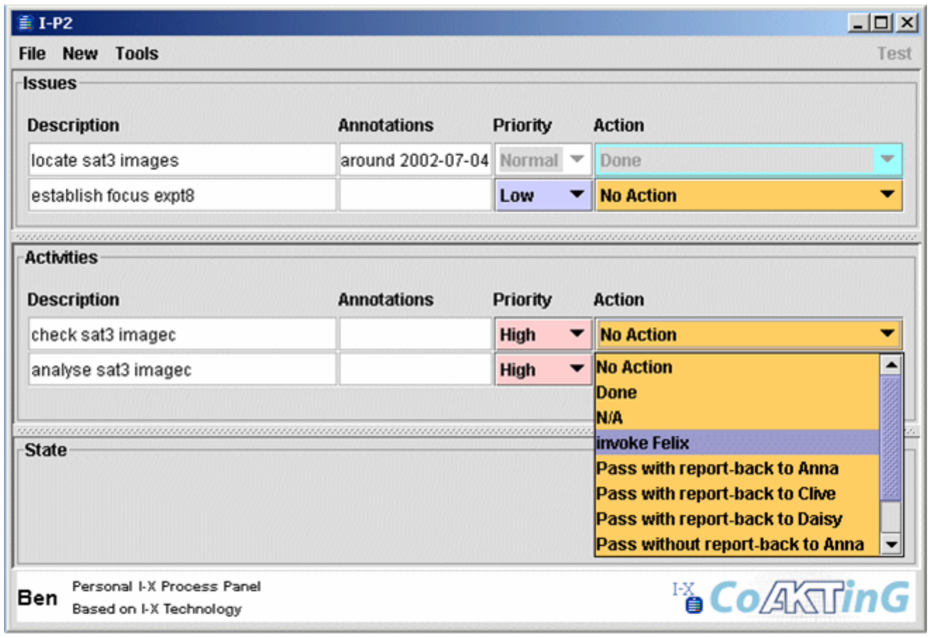

[Map]: Upgrading Sat3 [12 Mar 01 - Pisa]
quick +easy to do
What are the competitors?
Elli - circulate the Feb's review article
it's so flaky - time to switch
Upgrade Lsoft
When does Lsoft upgrade come out?
Ben - chase Lsoft v6.5
this is the real issue
How to fix Sat3 output?
Improve Lens3 -> Sat3 bandwidth
What syntax does Sat3 require?
Felix knows this stuff
email from Felix on Sat3 XML
Recruit a real research fellow! ;-D
Dave's leaving in June...
Anna - confirm with Personnel
Elli - draft job spec by 15 April

Fig. 2. A Compendium map showing various node types and links

In our experience, Compendium introduces a distinctive element to the design space of knowledge technologies, namely, making meetings into true events for group knowledge creation which leave a *trace* - a structured, collectively owned, searchable group memory that is generated in real time as a product of a meeting. Effective, on-the-fly construction of knowledge resources does not come "for free" - the lower the effort invested at the capture stage (e.g. simply video recording all meetings, or taking conventional minutes), the more work is required for collective reuse and computational support. Naturally, we want quality knowledge resources for minimal effort, and while smart analysis technologies will continue to push the boundaries, there are pragmatic factors to consider: what is possible *now*? Compendium tackles the capture bottleneck that any knowledge construction effort must confront, by investing effort in real time quality capture by a facilitator, mediated and validated by those at the meeting.

2.3 I-X Process Panels

I-X is a suite of tools[9] whose function is to aid in processes which create or modify one or more "product" (such as a document, a physical entity or even some desired changes in the world state). The main interface is the I-X Process Panel (I-P2) which, in its simplest form, acts like an intelligent "to do" list. The panel shows users their current issues and activities, on which Standard Operating Procedures can be applied to manage complex and long-running processes. I-X also has a collaborative element to it, in that issues and activities can be passed between different process panels to enact a workflow across an organisation. Web services can be called to automatically enact steps of the processes involved. Progress and completion reporting between panels and external services is possible. The underlying model on which I-X is based is the <I-N-C-A> Constraints Model[10]. In a meeting scenario, actions raised in a meeting have a direct mapping to <I-N-C-A> activities. Actions created in a meeting specific I-X panel are passed onto the relevant user panel's for individuals, which, on completion report back.


I-P2

File New Tools Test

Issues

Description	Annotations	Priority	Action
locate sat3 images	around 2002-07-04	Normal	Done
establish focus expt8		Low	No Action

Activities

Description	Annotations	Priority	Action
check sat3 imagec		High	No Action
analyse sat3 imagec		High	

No Action
Done
N/A
invoke Felix
Pass with report-back to Anna
Pass with report-back to Clive
Pass with report-back to Daisy
Pass without report-back to Anna

State

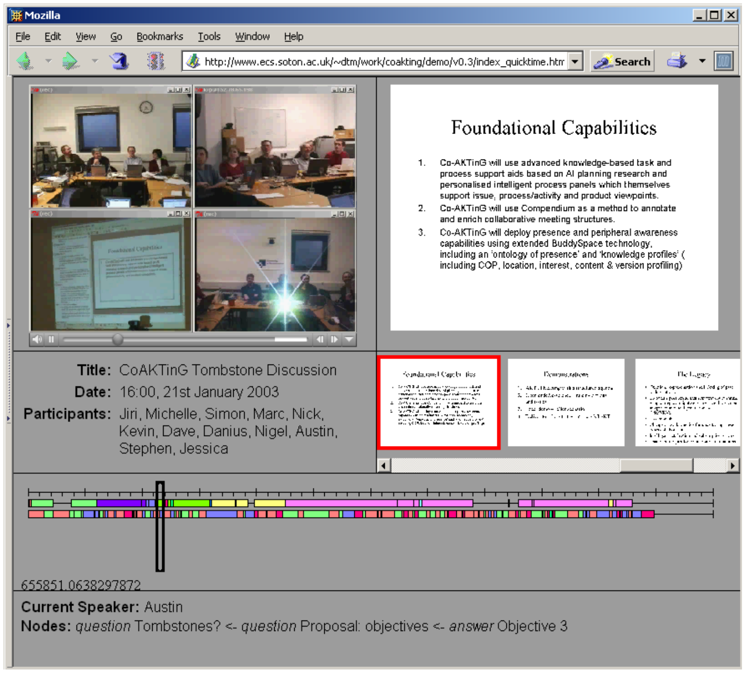
Ben Personal I-X Process Panel
Based on I-X Technology

I-X CoAKTinG

Fig. 3. A I-X Process Panel showing pending issues and activities

3 Meeting Replay

Once a meeting has taken place it can be useful to revisit the ideas and topics discussed. Traditionally, formal minutes are taken to record the salient points, but often these are too brief to be more than a simple aide memoire; in the typical CoAKTinG scenario (such as an Access Grid node) full audio and video logs are available, but conversely these are too verbose to be of practical use. We require the ability to select high-level points of reference from the meeting, then "zoom in" to view detailed records. e.g. a user sees from Compendium notes that a decision was made, but to understand the subtle reasoning behind that outcome wishes to view the video of discussion between participants. Each meeting is described using RDF conforming to the OWL meeting ontology; this represents resources such as: the meeting time, location, attendees, audio/video recordings, any presentations given (and associated web viewable versions), and argumentation annotation from Compendium. The Event / has-sub-event structure held within the RDF is mapped onto a more conventional time-line, which is automatically published using HTML and Javascript on a web site (figure 4). The user can navigate the meeting using the video timeline, or jump to a different point in the meeting by selecting a particular event, such as a slide being presented, or a Compendium node being created. By using the shared AKT reference ontology, we can also link to further information about resources held in other knowledge bases, e.g. when a person is referenced we link to information about them in the populated AKT triple store. We populate the timeline with any temporally annotated information about the meeting that would aid the user in navigation.



Mozilla
File Edit View Go Bookmarks Tools Window Help
http://www.ecs.soton.ac.uk/~dtm/work/coakting/demo/v0.3/index_quicktime.htm
Search

Foundational Capabilities

1. Co-AKTinG will use advanced knowledge-based task and process support aids based on AI planning research and personalised intelligent process panels which themselves support issue, process/activity and product viewpoints.
2. Co-AKTinG will use Compendium as a method to annotate and enrich collaborative meeting structures.
3. Co-AKTinG will deploy presence and peripheral awareness capabilities using extended BuddySpace technology, including an 'ontology of presence' and 'knowledge profiles' (including COP, location, interest, content & version profiling)

Title: CoAKTinG Tombstone Discussion
Date: 16:00, 21st January 2003
Participants: Jiri, Michelle, Simon, Marc, Nick, Kevin, Dave, Danius, Nigel, Austin, Stephen, Jessica

655851.0638297872

Current Speaker: Austin
Nodes: *question* Tombstones? <- *question* Proposal: objectives <- *answer* Objective 3

Fig. 4. The meeting replay tool

In CoAKTinG we have experimented with:

– Agenda item
– Slide exhibits
– Compendium node
– Speaker identification

– I-X activity(action item) creation
– Buddyspace chat

By providing all available information we hope to cater for the many activities and contexts of the user, in a seamful[11] manner.

We can categorise the information presented in the entire meeting replay in terms of the dimensions "structured" and "detailed", as shown in figure 5. Video, for example, is high in detail, in that it captures the entire audio and visuals of the meeting. Structurally, it is relatively low, since although there is implicit structure (image frames and audio samples) these do not directly contribute to navigating the structure of the meeting. Video processing could applied to segment the video into scenes but structurally this would not provide much more than Speaker Identification. The Agenda, conversely, is high in meeting structure, but relatively low in the details. Compendium captures a moderate level of detail in a highly structured representation.

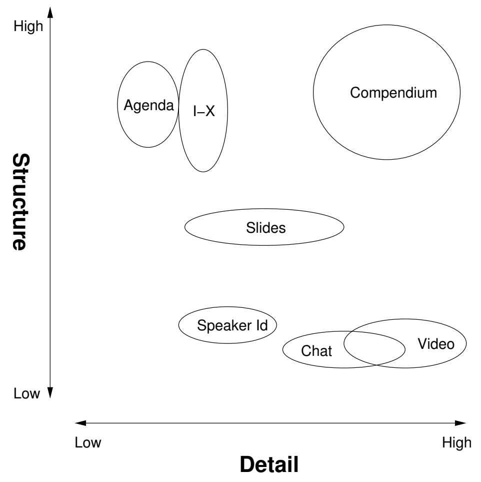


Fig. 5. Meeting Detail and Structure of recorded sources

4 Ontology

The Advanced Knowledge Technologies (AKT) project, with which CoAKTinG is affiliated, has developed a reference ontology [12] to describe the domain of computer science research in the UK, exemplified by the CS AKTive Space semantic web application. Within this domain, its vocabulary is able to express relationships between entities such as individuals, projects, activities, locations, documents and publications. For purposes of capturing meeting specific information, the reference ontology is already suitable for encapsulating:

- the meeting event itself
- meeting attendees
- projects which are the subject matter of the meeting
- documents associated with the meeting, including multimedia

For activities such as meetings, which we wish to index and navigate temporally, the way in which the ontology represents time is of particular relevance. The reference ontology contains the notion of an *Event*, which is a *Temporal-Thing* that can define a duration, start and end times, a location and *agents* involved in the event. More importantly, each Event can express a *has-sub-event* relationship with any number of other Events, and it is with this property that we build up our temporal meeting structure. Within the ontology there are also many Event sub-classes, such as *Giving-a-Talk*, *Sending-an-Email*, *Book-Publishing*, and *Meeting-Taking-Place*.

While the reference ontology provides a foundation for describing meeting related resources, the CoAKTinG meeting ontology (figure 6) extends the OWL version of AKT reference ontology to better encompass concepts needed to represent collaborative spaces and activities, including:

- time properties sufficient for multimedia synchronisation
- distributed gatherings to represent meetings which simultaneously take place in several spaces, both real and virtual
- exhibition of information bearing objects; e.g. showing a slide as part of a presentation
- compound information objects; e.g. to describe a presentation consisting of several multimedia documents
- rendering of information objects; e.g. JPEG image of a slide
- transcription of events; e.g. a video recording of a presentation, minutes of a meeting
- annotation of events; e.g. making a verbal comment, creating a Compendium node

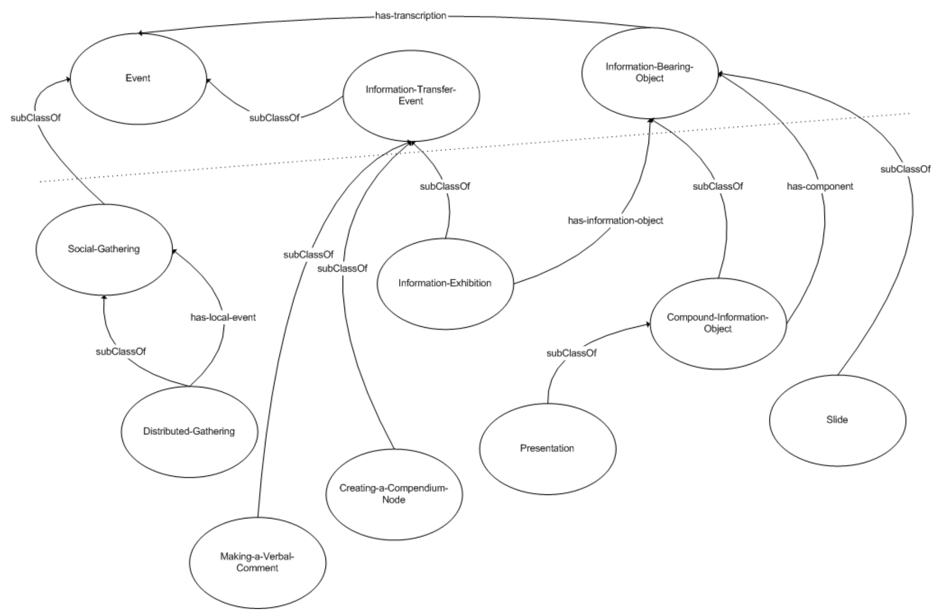


Fig. 6. A simplified representation of the meeting ontology

When a meeting takes place we "mark up" the event with meta-data - details such as those listed above - to build a structured description of the activities that occur. Through use of an ontology shared and understood by several different tools, we can lower the workload needed to provide usable and useful structure.

5 Case Studies

5.1 e-Response

One CoAKTinG demonstration scenario, termed e-Response, surrounds an evolving environmental emergency: an oil spill is threatening a sea-bird reserve. The response team (whose members are together assumed to have a wide-ranging scientific background) has to generate a plan for responding to this emergency – the creation of this plan is the synthesis task here.

In constructing their plan, the members of the team follow – individually and as a group – specific response procedures. While some of these may be extemporised and contingent on circumstances, others may be instances of 'standard operating procedures', generic approaches to archetypal activities, which can be downloaded from a central web-store. In addition to the human agents in this environment, automated agents exist to provide tide data and weather forecasts, simulate the progress of the oil slick, poll centralised data stores for details of available human expertise in specific fields and so on. The interactions are governed by the activities, issues and constraints that arise, and mediated by the I-X interfaces of the team members, which present to them the current state of the collaboration from their individual perspectives, and allow them to decompose activities, refine elements of the plan, delegate issues, invoke the automated agents etc, all serving to facilitate the team's task.

5.2 CombeChem

The CombeChem project aims to enhance structure property correlation and prediction by increasing the amount of knowledge about materials via synthesis and analysis of large compound libraries. Automation of the measurement and analysis is required in order to do this efficiently and reliably while ensuring that wide dissemination of the information occurs together with all the necessary associated background (raw) data that is needed to specify the provenance of the material. The project aims for a complete end-to-end connection between the laboratory bench and the intellectual chemical knowledge that is published as a result of the investigation; this necessitates that all steps in the process are enhanced by a suitable digital environment. CombeChem has achieved many parts of this ambitious programme, e.g. the smart laboratory (smarttea.org), grid-enabled instrumentation, data tracking for analysis, methodology for publication@source, process and role based security and high throughput computation.

The CoAKTinG tools provide support for the e-Science process in CombeChem and they also enable the digitisation of 'missing links' in the processing chain which form part of the typical collaborative scientific processes that we are attempting to enhance using the grid infrastructure: support of the experimental process, tracking and awareness of people and machine states, capturing of the discussions about data as well as the traditional metadata, and enriched metadata regarding these components to support interlinking.

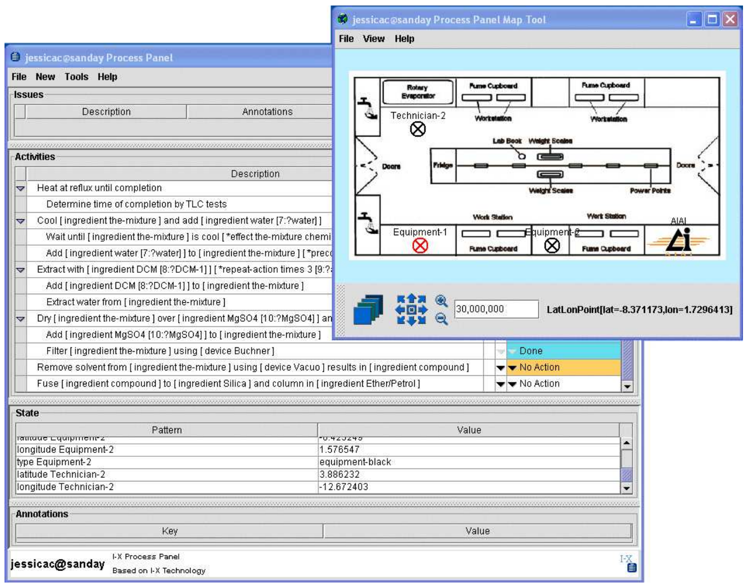
The BuddySpace systems can be adapted to show and track the interactions between the staff and equipment using the National

Crystallographic Service (NCS), providing information to their users about the state of the service. Compendium provides the harness to ensure more adequate capture of the discussions in analysis, while Process Panels provide the means to initiate and track key tasks and issues. Additionally the ideas from CoAKTinG provide different techniques to achieve the necessary multi-user interaction in real time over the network and give CombeChem the opportunity to implement the "video interaction" collaboration part of CombeChem using event based ontologies to annotate real time streaming media and content.

These various components are valuable complements to CombeChem individually but jointly are even more powerful. For example, Process Panels can exploit the presence information derived from BuddySpace with respect to instrument status and operator availability to offer more informed task delegation options. This completes the chain of digital support and capture, maximising the potential for re-use of the digital information in support of the scientific process.

The following figure illustrates one particular aspect of the deep integration – the application of the Process Panel tool to the laboratory, building on the process capture work of CombeChem's Smart Tea team.

Figure 7 shows a screen capture of an I-X Process Panel and its Map Tool resulting from our initial experiment. The Map Tool depicts a real Chemistry lab where both fixed and mobile entities are represented. The positions of mobile entities such as movable equipment and technicians are updated automatically through the (World) State sub-panel. By sharing information with BuddySpace, (dynamic) properties of devices are also described in the same panel. At this particular point in time, it shows Technician-2 is in front of the Rotary Evaporator and about to carry out the sub-process "Remove solvent from the-mixture using Vacuo results in Compound", having completed the previous steps in this process. In our investigation, the process decomposition facility of the I-X Activity sub-panel supports views of different levels of abstraction that fits nicely with different chemists' (and labs') practice. Activities, issues, annotations and constraints may be recorded directly or via Compendium where in-depth discussion has taken place. Static and dynamic process editing provide great flexibility as processes are modifiable at run-time in response to unexpected changes. The ability to store, retrieve and refine process models is important in the Chemistry domain where existing processes are constantly reviewed and modified to discover or synthesise new chemical compounds. This facility alone makes I-X a valuable back-end component for integration with the existing CombeChem Grid.



LatLonPoint[lat=-8.371173,lon=1.7296413]

Activities

	Description		
▽	Heat at reflux until completion		
	Determine time of completion by TLC tests		
▽	Cool [ingredient the-mixture] and add [ingredient water [7:?water]]		
	Wait until [ingredient the-mixture] is cool [*effect the-mixture chemi		
	Add [ingredient water [7:?water]] to [ingredient the-mixture] [*prec		
▽	Extract with [ingredient DCM [8:?DCM-1]] [*repeat-action times 3 [9:?		
	Add [ingredient DCM [8:?DCM-1]] to [ingredient the-mixture]		
	Extract water from [ingredient the-mixture]		
▽	Dry [ingredient the-mixture] over [ingredient MgSO4 [10:?MgSO4]] an		
	Add [ingredient MgSO4 [10:?MgSO4]] to [ingredient the-mixture]		
	Filter [ingredient the-mixture] using [device Buchner]		Done
	Remove solvent from [ingredient the-mixture] using [device Vacuo] results in [ingredient compound]		No Action
	Fuse [ingredient compound] to [ingredient Silica] and column in [ingredient Ether/Petrol]		No Action

State

Pattern	Value
latitude Equipment-2	-0.425249
longitude Equipment-2	1.576547
type Equipment-2	equipment-black
latitude Technician-2	3.886232
longitude Technician-2	-12.672403

Annotations

Key	Value

jessicac@sanday I-X Process Panel Based on I-X Technology

Fig. 7. I-X Process Panel configured for e-Chemists

6 Conclusions

This paper has introduced the tools that have been developed by the CoAKTinG project and identified how they are typically used in meetings, and also shown how they are being explored in scenarios such as e-Response and CombeChem.

Acknowledgments

This work is funded by the UK Engineering and Physcal Sciences Research Council under grant number GR/R85143/01, in association with the Advanced Knowledge Technologies Interdisciplinary

Research Collaboration. The AKT IRC research partners and sponsors are authorized to reproduce and distribute reprints and on-line copies for their purposes notwithstanding any copyright annotation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of other parties. We are grateful to members of the CombeChem team for their assistance with the case study: Jeremy Frey, Gareth Hughes, Hugo Mills, monica schraefel and Graham Smith. CombeChem is also funded by the EPSRC under grant number GR/R67729/01.

References

1. The CoAKTinG project. http://www.aktors.org/coakting/
2. The Access Grid. http://www.accessgrid.org/
3. Conklin, J., Selvin, A., Shum, S.B., Sierhuis, M.: Facilitated Hypertext for Collective Sensemaking: 15 Years on from gIBIS. In: Proceedings The Twelfth ACM Conference on Hypertext and Hypermedia (Hypertext '01). (2001) 123–124
4. Selvin, A., Sierhuis, M.: Towards a Framework for Collaborative Modelling and Simulation. In: Workshop on Strategies for Collaborative Modelling and Simulation, CSCW'96: ACM Conference on Computer-Supported Cooperative Work, Boston, MA, USA (1996)
5. Conklin, J., Yakemovic, K.C.B.: A Process-Oriented Approach to Design Rationale. Human-Computer Interaction **6** (1991) 357–391
6. Selvin, A.: Supporting Collaborative Analysis and Design with Hypertext Functionality. Journal of Digital Information **1** (1999)
7. Shipman, F., McCall, R.: Supporting Knowledge-Base Evolution with Incremental Formalization. In: Proc. ACM CHI'94: Human Factors in Computing Systems, Boston, Mass., ACM Press: New York (1994) 285–291
8. Selvin, A., Shum, S.B.: Repurposing Requirements: Improving Collaborative Sensemaking over the Lifecycle. In: Profess'99: International Conference on Product Focused Software Process Improvement, Oulu, Finland. (1999) 539–559
9. Tate, A., Dalton, J., J. Stader, J.: I-P2 - Intelligent Process Panels to Support Coalition Operations. In: Proceedings of the Second International Conference on Knowledge Systems for Coalition Operations (KSCO-2002), Toulouse, France (2002)
10. Tate, A.: <I-N-C-A>: an Ontology for Mixed-initiative Synthesis Tasks. In: Proceedings of the Workshop on Mixed-Initiative Intelligent Systems (MIIS) at the International Joint Conference on Artificial Intelligence (IJCAI-03), Acapulco, Mexico (2003)
11. Chalmers, M., MacColl, I., Bell, M.: Seamful design: Showing the seams in wearable computing. In: Proceedings of IEE Eurowearable 2003, Birmingham, UK (2003) 11–172
12. The AKT Reference Ontology. http://www.aktors.org/ontology/

Distributed Data Management and Integration: The Mobius Project

Shannon Hastings and Stephen Langella, Scott Oster, Joel Saltz

Department of Biomedical Informatics
Multiscale Computing Laboratory
The Ohio State University
Columbus, OH, 43210

Abstract. Advances in computational resources, the concept of *the Grid* and the development of middleware and applications to build and support the Grid will give Grid users the power to access and interpret large amounts of heterogeneous data. *Metadata*, data that describes data, can be used to represent and define the protocols of the Grid, as an abstraction between services and datasets, and to provide syntactic or semantic descriptions and annotations of any sized datasets so that they may be discovered, communicated, and analyzed efficiently. In this document, we will present "Mobius", which can be described as a generic and extensible set of protocols and services for managing data in a Grid environment.

1 Introduction

With the emergence of Grid computing and Web services architectures, it is increasingly critical to address the information service needs of loosely coupled systems. A multi-institutional Grid environment will contain many data sources, which maintain different types of data using different storage mechanisms. Management of data and metadata is a major task in such systems. Seamless integration of data can facilitate the discovery of data-oriented findings that would not be possible without a system that supports the distributed management of metadata and data in a scalable environment. A middleware system which provides basic building block services for metadata and data management can be extended by many other service areas such as semantic query services, ad-hoc data warehouse services, and specialized data integration services.

In this paper, we describe the architecture of Mobius, a middleware framework designed for efficient management of data and metadata in dynamic, distributed environments. Its design is motivated by the Grid [7, 5, 6] (in particular by the activities of the Data Access and Integration Services group at Global Grid Forum [8, 4, 1, 2]

), by earlier work done at General Electric's Global Research Center [13], and by particular domain application area requirements. Biomedical research studies, for example, can involve integration of data, including proteomic, molecular, genomic, and image data in a multi-institutional environment. Mobius provides a set of generic services and protocols to support distributed creation, versioning, and management of data models and data instances, on demand creation of databases, federation of existing databases, and querying of data in a distributed environment. Its services employ XML schemas to represent metadata definitions (data models) and XML documents to represent and exchange data instances.

As an example, consider biomedical research studies that collect data in complex data types, with partial syntactic and semantic overlap. A researcher can develop a hypothesis and accrue several types of patient and laboratory related data. The researcher needs to create databases to maintain data for patients and laboratory results. Data may have also been previously stored in multiple sources and databases, potentially created by other researchers. In which case, data must be integrated from these potentially heterogeneous sources for analysis. The researcher should be able to use a system to create an ad-hoc data warehouse spanning multiple databases (2D gel data, clinical data, lab data, drug treatment data, and molecular data) to enable distributed analysis. The researcher should also be able to use the ad-hoc data warehouse to carry out queries to test hypotheses. Any two databases may define data that contain the same semantic content with completely different structure representations. The analysis of data may lead to collection of new datasets as well as new types of data. This type of scenario can be supported by a system that will allow the researcher to:

– create schemas which describe their data models, register and share these data models with other collaborators, manage and version them while new data types are added or deleted,
– facilitate translation between data models that have the same semantic meaning, but different structure, and
– create, integrate, and manage databases and data that conform to these data models.

Mobius consists of three core service areas: Global Model Exchange (GME), Data Instance Management (Mako), and Data Translation Service (DTS). These service areas and underlying protocols support distributed creation, versioning, management of data models defined by XML schema, on-demand creation of distributed databases, federation of existing databases, querying of data in a distributed environment, and on demand translation of instance data from one data model to another.

Using Mobius, the biomedical researcher can develop databases and querying capabilities for her studies as follows. The researcher first designs XML schemas describing the data types she wants to maintain. The GME provides several alternatives for the researcher to create and register her schemas: 1) The researcher can search the GME for existing schemas and may use one that suits her research study. 2) She can version an existing schema by adding or deleting data attributes. 4) The researcher can create a new schema with new attributes and structure. 5) She can compose a schema using new attributes and by referencing multiple existing schema entities in her schema. Once the schema is created, it is registered with the GME so that other researchers can search for it, and the Mako services can use it to create new databases and validate data against its data model. The researcher can now instantiate one or more Mako servers to maintain the databases conforming to the schemas. She can also register the schemas with existing running Mako servers. The Mako servers create databases using the XML schemas so that new data can be entered and maintained across the system. When a new data set is submitted (as an XML document) to a Mako server, the server ingests the document, stores the data specified in the document in the databases, and indexes them. Any given data set can also be distributed across a collection of Mako servers and rematerialized as needed at query or retrieval time. With the data effectively stored, the researcher can then retrieve data from these databases using queries expressed in XPath [3].

2 Requirements

As mentioned earlier, the world is increasingly becoming more dependent on electronic data management from patient medical records and research data spread across institutions around the world to remote monitoring of airline engine sensor data. Data is everywhere, and the easier that it can be integrated and used together the more useful it becomes. This section will enumerate some basic requirements for managing data in a grid environment and elaborate on each in a use case driven approach.

2.1 Global Model Management

In order for services on the grid to communicate with each other, their data must be described in a format that is understood by each involved component. Thus a data management system for the grid must provide a method for defining metadata and data, we will refer to this definition as a data model. A data model is the specification of the structure, format, syntax, and occurrences of data instances. In order for services to communicate with one another they must agree on a data model over which they communicate. In order for such models to exist, they must be globally available to every service, assuming the service has access control rights to obtain and view the model. Making data models globally available and uniquely identifiable forms a conceptual global model consisting of the individual models. Therefore, given proper credentials, one model may reference previously defined entities in another model. For example, if the entity *Patient* is defined in the model *Hospital*, it could be referenced by the model *Clinic*. This means that the Clinic's definition of Patient is equivalent to the Hospital's definition of Patient. Since all entities are globally identifiable, entities within the schema must be unique. Therefore once the Patient entity is defined, it cannot be defined by another model. Although this promotes standardization of data definitions, this will not be an acceptable solution for the grid, as there will invariably be competing definitions for these entities. This problem can be solved by namespaces. Entities defined in models should be assigned to a namespace, where entities are unique within their namespace. The combination of the entities name and namespace makes the entity unique within the grid. Thus, entities can be referenced, systems and data integration become simpler, and standards can be promoted.

2.2 Data Instance Management

Services on the grid will need a method for storing and querying data and metadata. This will be integral for service communication on the grid. Given valid credentials, data should be able to be stored across a series of heterogeneous loosely-coupled machines. This will allow for data storage systems to be virtually any size and scale within the grid as well as become ad-hoc and dynamic in size and shape. The system should also provide the ability to allow trusted users to update, query and delete the data. These generic base storage service interfaces can be extended by application specific grid users in order to tailor them for specific needs.

2.3 Data Translation

It will often be the case that institutions will have different methods for modeling data which may be conceptually or semantically the same. It would be beneficial to both institutions to be able to translate between one another's data models such that each can leverage the other services. Where possible, the simplest way to share data and services is to use common models, but in practice, this is not always possible. This begs the existence of a registry architecture for managing and discovering services on the grid that translate between defined data models. Such an architecture would allow services to programmatically identify and leverage other computational services which would otherwise not be possible. This, of course, does not mean that the automatic structural or semantic data translation problem is solved. This simply gives a service architecture where data translation services, possibly written by domain experts, are published and can be trusted between the two domains in which the data is translated and potentially accepted by others as the standard.

3 Related Work

Due to the nature of a large componentized middleware architecture, covering all possible related work which may be leveraged by this architecture is beyond the scope of this paper. In this section we will cover the related work which deals specifically with grid middleware protocols and grid metadata management; the main contributing components of the mobius architecture. We will not cover related work pertaining to specific service implementation details. However, if there is related work which addresses a similar problem, as a Mobius service implementation, it can simply be wrapped with the Mobius service protocols or be used in coordination with the Mobius middleware.

There have been several projects which focus particularly on service infrastructure for metadata management, and a few which specifically target execution on the grid. Storage Resource Broker (SRB) [20] in coordination with the Meta Information Catalogue (MCAT) [15] is a grid-enabled middleware system for accessing heterogeneous data storage services in a client-server framework. In its current incarnation, the SRB system uses the term metadata to mean a predetermined set of key-value pairs which describe attributes of the data set. The user can add extra key-value entities which can be used to query and discover data sets in the SRB framework. The key defining difference between the SRB/MCAT solution and the Mobius framework is the way in which metadata is handled in the environment. The metadata in SRB is not a user-designed structure which is a published and managed entity. It is used to describe a specific data set or data sets and the metadata itself does not contain a particular unique name, or structure (other than key-value). The Mobius framework enables structured metadata of any size or shape to be user-defined, published, and managed. It also enables instance data conforming to this data to be created and validated. Although the two systems at a high level seem to support similar interactions, store and retrieve data which can be queried and discovered using metadata, they are quite different in scope and direction.

Another grid system for management of metadata is the Metadata Catalog Service (MCS) [16]. The MCS system was originally created to support metadata management in the Grid Physics Network (GryPhyN) [10]. MCS is closely related to SRB in that its current use and aim is to store metadata about logical files in a data grid. Although it does not provide standard data access interfaces that SRB or Mobius provides, it does store and allow querying of these logical files' metadata. Like MCAT, MCS models metadata as a set key-value pairs and not as a user-defined complex structure which can be comprised of other user-defined structures. Mobius addresses the concern that a generic platform for metadata in a grid should be capable of being structured and comprised of pre-existing structures from experts of their respective domains.

4 Global Model Exchange

The driving force behind the need for a global service architecture to manage user-defined data models can be laid out as a set of use-case driven requirements. A user in the grid would like to be able to create, publish, and version a data model, possibly defined using XML Schema. That model may be used by many other services at service runtime. This feature requires that those requesting services be able to request model definitions by namespace and name, and be guaranteed that they will receive the same model as any other service issuing the same request. The publishing grid user would like other grid services to be able to use that model and detect and use its changes, enabling possible programmatic updating of running services. A particular model may be made up of several sub-models as defined under different namespaces. The grid user would like to be assured that when a model is referenced in the grid, there is some assurance it is always available, as well as any of the sub-models that it may be referencing which are defined by other grid users. A service infrastructure that provides an implementation of these requirements would be the core building block for other services as model-to-model translation or mapping services, generic ad-hoc model instance storage services, and robust service-to-service data communication. These requirements could mostly be met with a combination of current technologies and with a large amount of new philosophic agreements with all users. For example, the current defacto standard for schema publishing uses web servers, HTTP, and DNS maintained namespaces, and schemas are downloaded via HTTP requests to a URL matching their fully qualified names. If a certain level of service and availability could be guaranteed, servers maintained all pervious versions, and authorities were put in place to allow users to publish schemas to appropriate namespaces, we could approach a solution to these problems. However, a new set of grid services with the sole purpose of providing an implementation of these requirements will be much easier to adopt in practice, and will be a fundamental building block of many future data driven grid services.

The Global Model Exchange (GME) is a service that responds to the requirements stated above. It is responsible for storing and linking data models as defined inside namespaces in the distributed

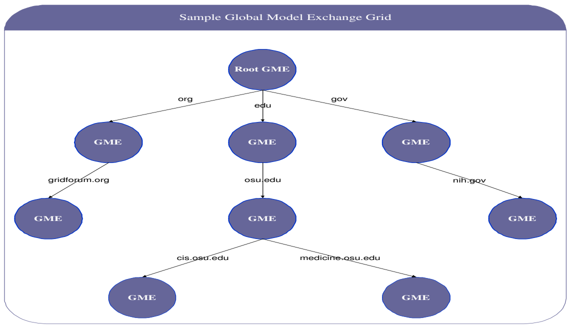


Fig. 1. GME Architecture

environment. The GME enables other services to publish, retrieve, discover, remove, and version metadata definitions. The GME services are composed together in a DNS-like architecture representing a parent-child namespace hierarchy as shown in the figure 1.

The *GMEAccess* interface is the user-level interface that exposes the functionality of these protocol messages through a basic set of service access methods. A user service can use the GMEAccess interface to publish a model, request an instance of a particular version of a model, and post a query against the models of a particular GME namespace.

The GME provides model version and model to model dependency management. For instance, if a user service publishes a model to the GME, and later the model is modified and republished, the model will automatically be versioned and both models can be used concurrently. Furthermore, if a suitable translation mapping is provided for changes between the two versions, they can be interchanged seamlessly. The protocol provides a mechanism for stating the exact version of a model for a given request. A model can also contain types defined by other models or references to types contained in other models. This reference integrity might be considered the largest requirement for a GME that the current use of a URL does not provide. The role of the GME in the greater picture is to ensure distributed model evolution and integrity while providing the ability for storage, retrieval, versioning, and discovery of models of all shape, complexity, and interconnectedness in a grid-like environment.

A future extension of the GME service architecture would be to support semantic model storage, versioning, and querying. Storing data models without semantics is a base case building block for being able to begin to store, search, and reason about semantic data models. We would like to extend the GME basic service by not only storing the model, but also adopting a semantic model definition language such as RDF [18] and provide higher level querying for those models. One could imagine being able to pose the question, "Are there any models published anywhere in the grid that have something to do with cancer research?". With higher level semantic model storage and querying, questions like this can be applied across all data models in all namespaces in the grid.

5 Mako

Mako is a service that exposes data resources as XML data services through a set of well-defined interfaces based on the Mako protocol. A data resource can be a relational database, an XML database, a file system, or any other data source. Data resources are exposed through a set of well-defined interfaces, thus exposing specific data resource operations as XML operations. For example, once exposed, a relational database would be queried through Mako using XPath as opposed to querying it directly with SQL. Mako provides a standard way of interacting with data resources, thus making it easy for applications to interact with heterogeneous data resources.

5.1 Mako Interfaces

The Mako client interfaces are similar to those of an XML database, however, since Mako is a distributed service, the interfaces themselves are motivated by the work of the Data Access and Integration (DAIS) [4, 1, 2] working group in the Global Grid Forum (GGF) [8]. Mako defines a set of access interfaces that define the data operations, and it also defines a set of handles that implement said operations on a given context. For example, the XPathAccess interface which defines XPath operations, is implemented by the XMLCollectionHandle and by the XMLDocumentHandle. In the context of the XMLCollectionHandle the XPath operations are implemented and applied across the entire collection, whereas in the context of the XMLDocumentHandle the XPath operations are performed only against the XML document that the XMLDocumentHandle represents. In all Mako provides three handle types, a XML Data Service Handle, a XML Collection Handle, and a XML Document Handle.

5.2 Mako Architecture

Clients interact with Mako over a network; the Mako architecture illustrated in Figure 2 contains a set of listeners, each using an implementation of the Mako communication interface. The Mako communication interface allows clients to communicate with a Mako using any communication protocol. For example, if the communication interface is implemented using Globus [9] security, clients may communicate with Mako using the Globus Security Infrastructure (GSI). Each Mako can be configured with a set of listeners, where each listener communicates using a specified communication interface.

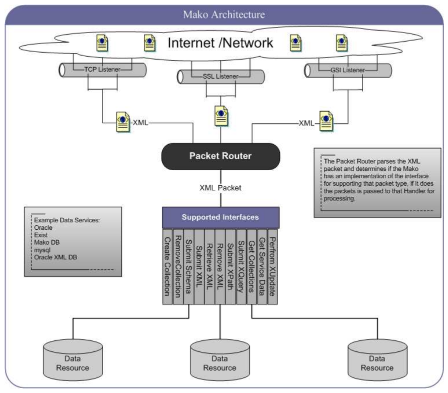


Fig. 2. Mako Architecture

When a listener receives a packet, the packet is materialized and passed to a packet router. The packet router determines the type of packet and decides if it has a handler, described below, for processing a packet of that type. If such a handler exists, the packet is passed onto the handler which processes the packet and sends a response to the client.

5.3 Mako Protocol

The Mako Protocol defines a set the packet types that a Mako may process. This section will give an overview of some of the operations that the Mako protocol supports.

Service Data Service Data, a term often used by the OGSI [17] working group in the GGF, refers to metadata about a given service. Mako service data contains information on when the Mako was started, information on the underlying data resource, a list of request types supported, etc.

Collection Management In the context of XML databases, the common nomenclature for referring to a group of related instance documents in a single storage service is a Collection. This is very similar to the relational database concept of a database. The concept diverges slightly, in that collections can have sub collections, and collections may not have a single schema (or any at all) associated with them. Mako has three request packets for managing collections of XML documents. The CreateCollectionRequest packet is used to create a new XML collection on a Mako. The RemoveCollectionRequest packet can be used to remove a collection from a Mako. Finally, the CollectionListRequest packet requests a list of collections that exists on a Mako or in a sub collections on a Mako.

Schema Management Each collection in Mako can be restricted to only accept XML documents from a set of certain types. This is accomplished by specifying a set of XML schemas in which a XML document must be validated against. The Mako protocol provides a method for adding and removing schemas to and from an XML collection within a Mako. The SubmitSchemaRequest packet is used to submit schemas to a collection in a Mako, and the RemoveSchemaRequest packet removes them. The SchemaListRequest packet can be used to get the list of schemas supported by an XML collection.

Document Management The Mako protocol defines methods for submitting, updating, removing, and retrieving XML documents. The SubmitXMLRequest packet is used to submit documents to an XML collection in a Mako. Upon submission, Mako assigns each entity a unique identifier; later we will see why this identifier is important. XML documents that reside in a Mako can be updated using XUpdate [14], the XUpdateRequest packet is used to accomplish this.

Mako provides two methods of removing documents. First, a list of documents can be removed by specifying their unique identifiers, this is done using the RemoveXMLRequest packet. Second, XML documents can be removed by specifying an element identifying XPath [3] expression. The XPathRemoveRequest packet is used to remove XML documents that meet an XPath expression.

The RetrieveXMLRequest packet is used to retrieve XML documents, or a subset of XML documents, from a Mako. Documents, or subsets of XML documents, can be retrieved by specifying their unique identifier. Recall that each element in an XML document is given an identifier, making any subset of a document uniquely addressable. The Mako protocol also allows the level of retrieval to be specified. For example, if you think of an XML document as a tree, then given a unique identifier, one would be able to specify how many levels of children should be included in the materialization of the document. Elements containing children that are below the height specified would not be included in the materialization, however references to their immediate children would be included. This feature becomes quite valuable when working with large datasets, in that full documents do not need to be materialized just to view partial results. It also allows one to build a demand driven Document Object Model (DOM) on top of the protocol. In general such a feature improves application performance by allowing the application to specify how data is materialized.

Querying Mako provides query support through the use of XPath [3], XPath queries are performed using the XPathRequest packet.

5.4 Mako Handlers

As specified in the architecture section, when a Mako receives a packet, the packet is given to the handler that processes that type of packet. Thus, for each packet type in the Mako protocol, there is a corresponding handler to process that packet type. In order to abstract the Mako infrastructure from the underlying data resource, there is an abstract handler for each packet type. Thus, for a given data resource, a handler extending the abstract handler would be created for each supported packet type. This isolates the processing logic for a given packet, allowing a Mako to expose any data resource under the Mako protocol. The initial Mako distribution contains a handler implementation to expose XML databases that support the XMLDB API [21]. It also contains handler implementations to expose MakoDB. MakoDB is an XML database optimized for interacting in the Mako framework. Implementations exposing other data services will be distributed as they become available.

5.5 Exposing a Data Service with Mako

Data services can easily be exposed through the Mako protocol by creating an implementation of the abstract handlers. Since there is an abstract handler for each packet type in the protocol, data services can expose all or a subset of the Mako protocol. Once handler implementations exist, Mako can be easily configured to use them. This is done in the Mako configuration file, which contains a section for associating each packet type with a handler.

5.6 Global Addressing

We mentioned earlier that Mako provides a method of uniquely identifying elements contained in XML documents. In actuality these elements are uniquely identified across the collection in which they reside. Mako also provides a method of globally addressing XML elements. This is done using the three tuple id, (Mako URI, collection, elementId). Being able to globally address entities within Mako provides several advantages. Most importantly it facilitates data federation across multiple Makos.

5.7 Virtual Inclusion

One example of how data may be federated across multiple Makos is by *virtual inclusion*. Virtual inclusion is a reference within an XML document to another XML document. This means that Mako allows XML documents to be created that may contain references to existing XML documents or elements, both local and remote. In this way, an XML document can be distributed and stored across multiple Mako servers by storing subsections of the document remotely and integrating them with references. This ability is critical in enabling large data documents to be partitioned across a cluster while still maintaining the single document semantics of a model.

5.8 VMako

Utilizing Mako's component architecture, alternate protocol handlers can be installed in a Mako server to enable it to utilize remote Mako instances. The Virtual Mako, illustrated in Figure 3, is a collection of protocol handler implementations that extend the operation of the data services to operate on an administrator-defined set of remote Makos. It maps a number of Virtual Collections to a set of remote collections. This simplifies the client-side complexity of interfacing with multiple Makos by presenting a single virtualized interface to a collection of federated Makos.

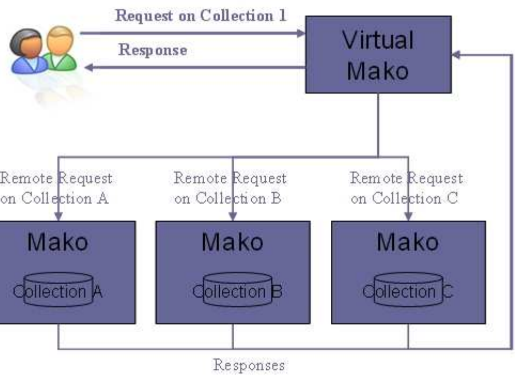


Fig. 3. Virtual Mako

For example, the SubmitSchemaHandler is extended to broadcast schema submission requests to all remote Makos, and return an aggregated response. The SubmitXMLHandler utilizes a configurable data ingestion algorithm to determine what to do with submitted instance data. It can be configured to distribute instance data to supporting Makos via a Round Robin scheduler, or any other desired distribution mechanism. Other handlers are implemented in a similar fashion. The Virtual Mako could also be utilized to decluster large data types. By utilizing virtual inclusion, a submitted XML instance document could be broken down into separate sub-documents, and distributed across remote Makos. A new document containing references to these sub-documents would then by created and stored in a selected remote Mako. A XMLDocumentHandle to this new document would be returned to the client. A request to materialize this document would then completely restore the original document.

While this virtualization eases the burden of interfacing with multiple Makos, the primary purpose for a Virtual Mako is to enable distributed query execution. In a Virtual Mako, the XPathHandler is implemented such that requests are broken down into sub-queries and sent to appropriate remote Makos. Responses are then aggregated at the Virtual Mako, and returned to the client. In the current implementation, the Virtual Mako acts as a simple mediator and aggregator. The architecture was designed such that this could be replaced by a sophisticated distributed join implementation without affecting any client-side code.

Future work for extending the Virtual Mako concept will focus on adding semantic information about remote data types such that the Virtual Mako can make informed decisions about where to store data, as well as utilizing this information to enable ad hoc queries. For example, with ontological knowledge of remote data types and their relationships, a query could be executed by utilizing data transformation services and by querying for semantically compatible data types instead of strict class equality.

6 Data Translation Service (DTS)

The Data Translation Service (DTS) is responsible for managing the translation between data types. It accomplishes this task by maintaining a registry of remote mapping services, which provide pairwise translation between namespace elements. As an example, consider the use case where a data type A is required to be translated to datatype *B*. There are two primary motivators for the DTS. The first driver is the need to maintain the link between data stored against a particular version of a schema and future versions, as the schema evolves. As data in Mobius is strongly typed against a particular version of at least one schema, it is important to be able to easily migrate data adhering to a schema when the schema changes.

Given a suitable mapping service from one version to the next, data could either be migrated to the new schema, or left unchanged but migrated on-demand when requests for data adhering to the new version is requested.

A second driver is the need to seamlessly translate data types that are semantically similar but syntactically different. As the number of distributed data sources and data types increase, the need for translation between common data types becomes extremely important. It is unreasonable to assume that in a Grid environment all services will use the same data types for services that are slightly related. It is also unreasonable to assume that even if the Grid started out using the same data types for services, that those data types would not slowly evolve and change over time. Inevitably there are subtle differences between services that work on related data that necessitate variations in the data's representation. Rather than try to impose a universal standard representation for all data types, Mobius takes the approach of encouraging organizations to represent their data as they see fit, while still enabling them to inter-operate with similar data types via a DTS published translation service. The hope is that by easing the data transition between separate schema versions, and even completely different schemas, the communities will be able to evolve standards organically where standards are appropriate.

While the DTS enables point to point translation of data types, it is expected that mapping services will be provided that map their data types to appropriate standards, and from said standards to their data types. This will enable a broader range of transitive mappings without requiring explicit mappings for every data type.

The basic DTSR (Data Translation Service Registry) is a simple registry style service architecture. A DTSR will contain the information that describes the individual DTSs that are running on the grid. Any user can build a DTS publishable service as long as their service implements the DTS service interface and registers itself with a DTSR who is responsible for one of the namespaces it is mapping to or from.

Once this service infrastructure is put in place you can envision the DTS being extended to be able to begin to do semantic translation. A user could begin to pose questions like "Can anyone map my 'car' to their 'car'?", where "car" is some model that has some meaning to the user to be the semantic model of a car. This will require an ontology registry which will store semantic information and relationships about registered namespace elements. It is expected that for many pairs of data types there will be many possible translations between the two, and a semantic model and query language will be required for users to specify translation preferences. Research in this area is ongoing, but it is our belief that a schema-based structural translation service is a necessary base case required prior to supporting semantic translation.

7 Applications

Although Mobius is a research effort still under active development, it has been successfully leveraged as a data management middleware for several applications. In a collaboration with Rescentris, a company focusing on information management solutions to enhance life sciences research and development, Mobius has been utilized to provide a framework for integrating and aggregating collections of disparate medical data sets. This work employed Mobius's ability to store, retrieve, and query distributed data sets modeled by XML schema. By using referenced model elements to describe common medical metadata, this application was able to interrogate biologically relevant portions of distinct medical data sets. Specifically, a Virtual Mako was used to query and aggregate distributed data sets representing single nucleotide polymorphisms, and molecular cytogenetic data, via XPath queries.

Mobius has also been leveraged in several applications which operate on large medical image data sets. The Distributed Image Archive System (DPACS), was designed to support generic image archival and management in a grid wide environment, and relied heavily on the Mobius framework's ability to perform on demand data loading using remote element referencing(*Virtual Inclusion*). The DPACS system consists of a front end client application and a collection of federated back end Mako servers. The Mako servers provide the image and metadata storage and retrieval, and the client application can be used by user to upload, query, and inspect the archived data. The DPACS client broadcasts data reference queries to the remote Makos and is able to quickly display the resultant data sets. The requested aspects of the data sets are then loaded on demand as needed by the client, such as the metadata, volumetric thumbnails, and ultimately the image data itself. In another work, Mobius was used to store and manage image processing workflows and image data sets in a system to support rapid implementation and distributed execution of image analysis applications [11], implemented using the Insight Segmentation and Registration Toolkit (ITK) [12] and Visualization Toolkit (VTK) [19]. In this work the processing pipeline discovered work by querying Makos, and processing components utilized Makos to retrieve and store their input and output data sets.

BlockMan, an application which is under current development, supports the distribution and indexing of extremely large data sets which may not fit within a single nodes storage limit. The BlockMan will ingest large binary data sets, partition and distribute the chunks based on a user specified model and distribution algorithm. It then providesa query interface into data and parallel retrieval of queried results. The uniqueness of this system is its ability of the user to add queryable meaning to the data set by allowing rich structured user-defined metadata to be attached to the data chunks. This enables the users to interactively query into datasets which before were not only hard to manage physically but extremely hard to locate and interrogate.

8 Conclusion

As the development of the concept of *the Grid* continues to evolve, we feel protocols and services for managing metadata, data descriptions, and datasets will play a critical role in the ultimate realization of the Grid. In this paper we have introduced the three core services of Mobius: Global Model Exchange (GME), Data Instance Management (Mako), and Data Translation Service (DTS), and shown how they can be leveraged both as a data integration framework in institutional research studies, and as key components of a global Grid.

References

1. M. Antonioletti, M. Atkinson, S. Malaika, S. Laws, NW Paton, D. Pearson, and G. Riccardi. Grid Data Service Specification. Technical report, Global Grid Forum, 2003.
2. M. Antonioletti, A. Krause, S. Hastings, S. Langella, S. Malaika, J. Magowan, S. Laws, and NW Paton. Grid Data Service Specification: The XML Realisation. Technical report, Global Grid Forum, 2003.

3. Anders Berglund, Scott Boag (XSL WG), Don Chamberlin, Mary F. Fernández, Michael Kay, Jonathan Robie, and Jérôme Siméon. *XML Path Language (XPath).* World Wide Web Consortium (W3C), 1st edition, August 2003.
4. Data Access and Integration Services. *https://forge.gridforum.org/projects/dais-wg.*
5. I. Foster, C. Kesselman, J.M. Nick, and S. Tuecke. Grid Services for Distributed System Integration. *IEEE*, 35(6):37–46, 2002.
6. I. Foster, C. Kesselman, J.M. Nick, and S. Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. *http://www.globus.org/ogsa*, 2002.
7. I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. 15(3), 2001.
8. Global Grid Forum. *http://www.gridforum.org/.*
9. The Globus Project. http://www.globus.org.
10. Grid Physics Network (GryPhyN). http://www.griphyn.org/index.php.
11. Shannon Hastings, Tahsin Kurc, Steve Langella, Umit Catalyurek, Tony Pan, and Joel Saltz. Image processing for the grid: A toolkit for building grid-enabled image processing applications. In *CCGrid: IEEE International Symposium on Cluster Computing and the Grid.* IEEE Press, May 2003.
12. National Library of Medicine. Insight Segmentation and Registration Toolkit (ITK). *http://www.itk.org/.*
13. S. Langella. Intellegent data management system. Master's thesis, Computer Science Department, Rensselear Polytechnic Institute, 2002.
14. Andreas Laux and Lars Martin. XUpdate Working Draft. Technical report, http://www.xmldb.org, 2000.
15. Meta Information Catalog (MCAT). http://www.npaci.edu/DICE/SRB/mcat.html.
16. Metadata Catalog Service (MCS). http://www.isi.edu/ deelman/MCS/.
17. Open Grid Services Infrastructure. *https://forge.gridforum.org/projects/ogsi-wg.*
18. Resource Description Framework (RDF). *http://www.w3.org/RDF/.*
19. Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit: An Object-Oriented Approach To 3D Graphics.* Prentice Hall, 2nd edition, 1997.
20. SRB: The Storage Resource Broker. *http://www.npaci.edu/DICE/SRB/index.html.*
21. XML:DB Initiative for XML Databases. *http://www.xmldb.org/.*

Interoperability and Transformability through Semantic Annotation of a Job Description Language

Jeffrey Hau, William Lee, and Steven Newhouse

180 Queens Gate, Department of Computing, Imperial College London,
London SW7 2BZ, UK
{jh398,wwhl,sjn5}@doc.ic.ac.uk

Abstract. The ability to submit jobs to foreign computational resources is a core requirement in all compute grids. Existing distributed resource management systems and job submission gateways require jobs to be described in propreitary languages that render transparent submission difficult. Standardisation effort to create a common language might resolve this problem. However, reliance on natural language documentation and the lack of a formal model to describe job attributes makes transformation error-prone and open to ambiguity. In this paper, we present the use of a generic transformation framework applicable to schematised language annotated with ontological metadata to job description documents. The paper demonstrates a meta job submission service utilising the framework to dynamically integrate and select heterogeneous resource management systems by transforming input job descriptions.

1 Introduction

Job Submission Service is one of the most important building block in a Service-Oriented Architecture based compute grid. There are many heterogeneous distributed resource management systems (DRM) providing propreitary interfaces to computational resources. Access to these systems can be characterised by the communication layer, notably command-line interface, socket-based approach, API with different language-bindings[8][9], HTTP web access and the recent adoption of Web Services. Apart from the means of communication, varieties of job description languages have been devised to describe the parameters of the job, such as input, output, arguments, system requirements, etc..

Although architectural differences between stakeholders can be accommodated by providing connectors or API to hide the communication detail, the diversity of job description however presents difficulty for clients to transparently access any computational resources without specific knowledge of the description syntax and semantics. It induces lock-in, hindering the network effect of Grid-wide job submission.

Recent effort of the Job Submission Description Language working group[7] in the Global Grid Forum aims to solve this problem by standardisation. The JSDL language presents a common extensible job description template absorbing many features accumulated from existing languages. The current form of the language centered around the notion of job attributes apparent in many languages (e.g. ClassAd[11], Globus RSL[10]). The standardisation effort aims to document a set of attributes and their meaning to achieve a common understanding across systems, therefore DRM vendors or third-parties can device transformations to their proprietary language without re-architecting their systems.

This paper motivates the use of ontology to compliment the descriptive nature of job submission language. Inferences on the relationship of attributes defined in heterogeneous job descriptions can provide insights to perform automatic transformation in some cases, or aid manual engineering in the under-specified ones. We argue that the current standard expressed in natural languages can be augmented with a formal model. We will provide an overview of a generic transformation architecture prototyped in ICENI[12]. The architecture exploits ontological metadata expressed in the Web Ontology Language (OWL)[3] attached to XSD schemas, in order to dynamically transform semantically compatible schematised XML. We will demonstrate how the transformation architecture is instantiated inside a meta job submission service to integrate and select at runtime a variety of compatible DRMs automatically. The exemplar also motivates application of this technique to other Grid services in a generic fashion.

2 ICENI Semantic Transformation Framework

The Semantic Transformation Service is a core element in ICENI to facilitate message exchanges between syntactically and architecturally incompatible services based on semantic and syntactic transformation augmented with architecture connectors. The transformation service extracts message semantics from ontology documents and reasons on their semantic compatibility. If message format is semantically compatible and transformable, the messages are transformed dynamically and passed onto the the architecture connectors capable of communicating with the selected implementations using the correct invocation method. The syntax translator performs syntactic transformation of the XML serialization to implementation native syntax (e.g. binary encoding) or performing additional actions (e.g. establishing encrypted session, etc..). In this section we will discuss each layers in more details.

2.1 Semantic Transformation Service

To transform semantically compatible but syntactically different message formats we need to able to reason about the structure of the messages. A typical Grid Service implemented using OGSI[13] or pure Web Service expresses its message structure through the XML Schema definition in its WSDL document. By linking Schema elements to ontology concepts we gain the semantic reasoning ability to transform messages. Once the compatibility is established, the messages are transformed using a backward chaining approach. Here we introduce the mapping and transformation process, for more details please refer to [6].

Semantic Mapping We take the approach of using a mapping for establishing the link between an XML Schema element and a Ontology concept. The mappings for a XML Schema document are expressed in RDF[4] and stored in RDF's XML serialization format[5] in a stand-alone *schemaMap* file independent of the ontology or the schema document. A distinct advantage of having separate schema mapping is that it allows the independent development of schema and ontology.

To link the XML Schema element with name *A* and type *A-Type* to an OWL class named *A-Concept*, we represent this link by a RDF graph. A *schemaMap* file contains at least one schema-to-ontology map. Each map has three properties, name, schemaType and semanticType. The property *name* represents the syntactic name given to the XML schema entity. *schemaType* is the XML Schema type given for the named entity in the XML Schema document. The *semantic-Type* points to a URI representing an OWL class or property. We envision graphical tools to help users in creating the schema map files. XML Schema and OWL documents can both be presented in graphical (trees and graphs) representations, this will allow users to drag and drop entities from both documents to establish the links between them.

Transformation After the semantic compatibility is deduced, we use an algorithm to create the transformation mapping functions[6] between between schema elements of source and target service. A skeleton[1] document is generated from the target service's schema. Applying the transformation mapping function, a pointer is placed in each of the value node that refers to the value in the source XML that needs to be transformed. Once this transformation document is generated, values are dynamically copied across during transformation.

2.2 Architecture Connector

The architecture connector acts as the bridge between different service layers. It is at the end of the transformation pipeline that takes messages off the request queue and invoke the backend service using secured socket, command-line integration with legacy systems or other pluggable means. Apart from being a mere message passing mechanism, it can also acts upon the message to perform actions, such as establishing session, file staging and other routines specific to the application using this framework.

3 A Generic Job Submission Service

The transformation framework (see Figure 1) is instantiated in the generic job submission service as the adaptation layer between the Web Service interface and the underlying distributed resource management systems. The Web Service interface provides a consistent architecture to submission clients advocated by the Grid community. It currently provides submission and monitoring capabilities. It encapsulates an extensible framework for plugging in architectural connectors to communicate with a variety of DRM systems. The connectors are associated with a set of XSD schemas describing the XML serialization of the DRM job description syntax annotated with

[1] A skeleton is an empty xml document conforming to the XML Schema definition

semantic link to a novel job terms ontology extracted from the JSDL language. By accepting input job description in a JSDL-like language also annotated with semantic link to the ontology, the transformation framework can deduce whether the input job description are transformable to any of the underlying DRM target language.

The technique has the advantage that new terms can be added at will without rewriting the generic job submission service. Only the ontological link needs to be appended or modified, which is more intuitive then a hard-wired transformation using XSLT or other declarative transformation language. Moreover, the semantic model applied to the job description can be used to deduce job description subsumption for scheduling (e.g. different scheduling policies apply to different "classes" of jobs) or monitoring purposes.

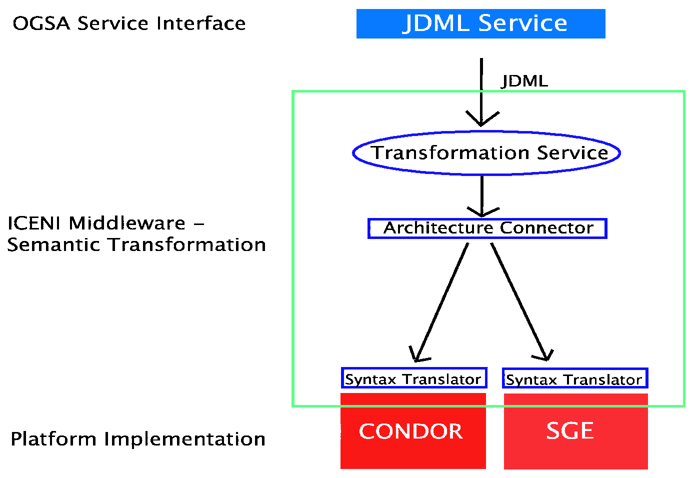


Fig. 1. Semantic Transfromation Architecture employed in the meta Job Submission Service

4 Future Work

The syntax translator needs a set of transformation rules to be predefined in order to perform the transformation from XML to implementation native syntax. This functionality is necessary due to the limitation that the transformation service can only automatically deduce and perform XML to XML transformation. We are currently investigating the process of XML to native syntax transformation. This could potentially further simplify our framework by render the syntax translator redundant.

The transformation framework is developed specifically to solve the problem of autonomously integrating heterogeneous job submission architectures with the Web Service interface. We believe the framework can be generalised to benefit a broader range of Grid Services (OGSA), such as resource usage service with diverse accounting attributes. Applying the current framework to different types of Grid Services as use cases, we hope to create a generic Grid Service transformation framework.

5 Conclusion

In this paper we discussed the difficulty of integrating different job submission languages and the advantages of using semantic metadata. We proposed a way of binding the metadata to the language schema by using an ontology. A framework is developed to utilise the ontologically annotated job description documents. These documents are then transformed dynamically to integrate and select heterogeneous resource management systems.

References

1. Foster I., Gannon D., Kishimoto H.: The Open Grid Service Architecture (Draft 0.14). https://forge.gridforum.org/projects/ogsa-wg March 2004
2. Christensen E., Curbera F., Meredith G., Weerawarana S., Web Service Description Language (WSDL), http://www.w3.org/TR/wsdl
3. Bechhofer S., van Harmelen, F., Hendler J., et al: OWL Web Ontology Language Reference. http://www.w3.org/TR/2004/REC-owl-ref-20040210 (10 February 2004)
4. Klyne G., Carroll J. J.: Resource Description Framework (RDF): Concepts and Abstract Syntax (W3C Recommendation). http://www.w3.org/TR/rdf-concepts/ 10 February 2004

5. Beckett D.: RDF/XMLSyntax Sepcification (Revised, W3C Recommendation). http://www.w3.org/TR/rdf-syntax-grammar/ 10 February 2004
6. Hau J., Lee W., Newhouse S.: Web Service Interoperability by Semantic Transformation. Submitted to 3rd International Semantic Web Conference
7. Job Submission Description Language Working Group (JSDL-WG): http://www.epcc.ed.ac.uk/ ali/WORK/GGF/JSDL-WG/
8. Distributed Resource Management Application API Working Group (DRMAA-WG): http://www.drmaa.org
9. Globus Commodity Grid Toolkits: http://www.globus.org/cog
10. The Globus Project: http://www.globus.org
11. Rajesh Raman, Miron Livny, and Marvin Solomon: Matchmaking: Distributed Resource Management for High Throughput Computing, Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, July 28-31, 1998, Chicago, IL
12. Imperial College e-Science Networked Infrastructure (ICENI): http://www.lesc.ic.ac.uk/iceni/
13. Tuecke S., Czajkowski K., Foster I., Frey J., Graham S., Kesselman C., Snelling D., Vanderbilt P.: Open Grid Service Infrastructure (OGSI) Specification Version 1.0

Using the Semantic Grid to Build Bridges between Museums and Indigenous Communities

Jane Hunter[1], Ronald Schroeter[1], Bevan Koopman[1], and Michael Henderson[1]

DSTC, University of Queensland, Brisbane, Australia 4072
{jane, ronalds, bevank, mjh}@dstc.edu.au

Abstract. In this paper we describe a Semantic Grid application designed to enable museums and indigenous communities in distributed locations, to collaboratively discuss, describe and annotate digital objects and documents in museums that originally belonged to or are of cultural or historical significance to indigenous groups. By extending and refining an existing application, Vannotea, we enable users on access grid nodes to collaboratively attach descriptive, rights and tribal care metadata and annotations to digital images, video or 3D representations. The aim is to deploy the software within museums to enable the traditional owners to describe and contextualize museum content in their own words and from their own perspectives. This sharing and exchange of knowledge will hopefully revitalize cultures eroded through colonization and globalization and repair and strengthen relationships between museums and indigenous communities.

1 Introduction

Many museums, archives, libraries and cultural institutions throughout the world hold large collections of objects that are of cultural or historical significance to indigenous communities. Because many of these objects were collected without the consent of the traditional owners, the custodial organizations are now facing the challenges of determining ownership, seeking direction from the traditional owners on the future of such objects and repatriating them, storing them or exhibiting them appropriately as requested. This process is made more difficult because colonization has caused many indigenous communities to become dispossessed of their lands and widely dispersed geographically. New collaborative interactive software tools, high-speed networks and emerging Grid technologies that facilitate communication and the sharing of resources and knowledge between geographically dispersed groups, appear to offer an infrastructure that is ideally suited to the implementation of such digital and physical repatriation programs.

Within this paper we describe the software that we are developing specifically for such an application, within the Smithsonian's National Museum of the American Indian (NMAI). In the United States, the Native American Graves Protection and Repatriation Act (NAGPRA) specifies the types of objects and sites to be protected and/or repatriated. Going beyond the requirements of NAGPRA, the NMAI has established a Culturally Sensitive Collections Care Program to respond to areas of concern of Native peoples with regard to the maintenance, presentation, and disposition of sensitive materials and information in the collections of the museum. Past experience has indicated that many tribal communities want access to the records of all objects in museum collections associated with their community and that after reviewing these, some will be satisfied with digital surrogates and access to physical objects when requested. The objective of the application described here is to provide the cyber-infrastructure to support such a program.

Within the FilmEd project [1], we developed the Vannotea system to collaboratively index, annotate and discuss digital film and video (MPEG-2) over high bandwidth networks. Vannotea has been extended to support the sharing, indexing and annotation of high-quality images (JPEG2000) and 3D objects (Direct 3D). Within the Indigenous Knowledge Management (IKM) project [2], we developed software tools to enable non-collaborative indexing, annotation and rights management of indigenous collections. In this paper we explain how we have amalgamated software developed within these two projects to produce a system that enables collaborative indexing, discussion, annotation and rights management of indigenous collections via access grid nodes. We have done this by extending the Vannotea software through the addition of:

– Fine-grained rights management components specifically required for Indigenous Knowledge;
– Support for the sharing, indexing and annotation of 3D digital objects.

The remainder of this paper is structured as follows. The next section describes related work and the background and objectives to the work described here. Section 3 describes the architectural design of the system and the motivation for design decisions that were made. Section 4 describes the issues involved in integrating Indigenous knowledge management requirements and adding support for 3D objects, respectively. Section 5 provides the conclusion and describes plans for future work.

2 Related Work and Objectives

Developing a system to collaboratively index, annotate and discuss high quality images, videos and 3-D objects within the context of indigenous collections involves research across a range of disciplines:

- Multimedia indexing, search and retrieval;
- Annotation tools for digital multimedia documents;
- Collaborative application sharing and document sharing tools;
- Authentication, authorization and digital rights management for Indigenous collections.

In the next four sub-sections we describe relevant, related work in these areas.

2.1 Multimedia Indexing, Search and Retrieval

There has been considerable work on the indexing, searching and retrieval of images and video content. Many automatic tools have been developed to extract low level features [3-6] and various sophisticated content-based retrieval methods have been developed (e.g. Query-by-Example, Sketching interfaces, etc.) [7-9]. A number of tools have also been developed to enable semantic descriptions to be manually attached to video [10-13] and image [14, 15] content using free text, controlled vocabularies or ontologies. Other research groups are attempting to "bridge the semantic gap" [3, 16, 17] by automatically generating semantic descriptions using machine-learning techniques.

Although the majority of multimedia indexing, search and retrieval systems target images and video, Rowe et. al [18] recently created a tool to capture, model, analyze, and query complex 3D data within an archeological museum context. Their focus was on feature extraction of 3D objects rather than on user annotations and only asynchronous, non- collaborative user access is provided.

None of the work carried out to date allows distributed groups to index and define the access rights policies associated with multimedia (images, video and 3D) objects collaboratively. Decision making within many indigenous groups is a group process, carried out by a council of elders. When describing cultural or historical objects and defining access rights, it's important that geographically dispersed community leaders can do this in collaboration with museum staff through real-time group discussions that will generate a legitimate consensus for future generations.

2.2 Annotating Digital Multimedia Documents

Existing annotation tools (which enable users to attach personal notes, questions, explanations, etc. to documents) can be categorized according to the media types which can be annotated (text, web pages, images, audio or video, 3D) and the extent of collaboration supported. This matrix in Table 1, gives an overview of the different products, tools, systems and projects according to these categories.

	Non-Collaborative	**Collaborative**	
	private annotations	shared annotations (assynchronous)	logged, live discussions (synchronous)
Text or webpages	text processors like MS Word, Adobe Acrobat, etc.	Annotea [19], Cadiz et al. [20]	Churchill et al. [21], Collaborative Information Browser [22]
Image	Adobe PhotoShop, QTVR	PAIS [23], Photo Annotator (Annotea + SVG) [24]	mimio classroom [25]
Audio/Video		MRAS [26]	DTVI [27]
3D		Jung et al. [28]	

Table 1. Annotation Tools

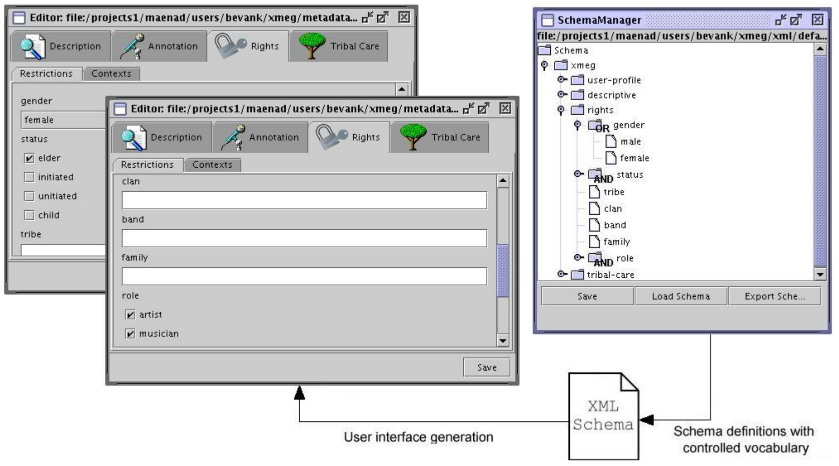
Microsoft's Distributed Tutored Video Instruction (DTVI) [27] is the only system that enables students to replay and discuss videos of lectures collaboratively. However it does not support real-time synchronous annotations. It is based on a combination of Windows Media Player and Microsoft's NetMeeting [29], which uses the T.120 protocol [30] for application sharing.

The approach adopted by application sharing protocols such as T.120 (NetMeeting) or VNC-Protocol [31] makes them unsuitable for our application. In such protocols, the shared application runs on a master client or server, which receives the keyboard and mouse events from the participants and sends captured screen/window updates back to the participants. Although this framework could potentially be used to transfer high frame rates of MPEG-2 video - with sufficient bandwidth and computing power - it clearly trades efficiency for generality.The protocols are also restrictive in terms of the collaborative environment, e.g. multiple users with multiple simultaneous cursors are not supported. Therefore we have had to build a collaborative environment from scratch, using .NET Remoting. This is described in detail in Section 4.4.

2.3 Indigenous Rights Management

One of the challenges that museums face when managing Indigenous collections is providing support for traditional laws related to the protection and preservation of sacred or secret resources. Attributes including a user's gender, tribal affiliation and status within the tribe are examples of the fine-grained access control required by organisations such as the National Museum for the American Indian (NMAI). Physical artefacts (such as those stored at the NMAI) also have special storage, orientation or preservation needs - known as tribal care constraints. Within the Indigenous Knowledge Management (IKM) project [2, 32] tools have been developed to enable traditional owners to define the specific rights requirements associated with digital objects and to match them against user profiles to control access. A number of other projects [33-35] have been developing software to support the management of indigenous multimedia collections but none provide the same level of granularity, flexibility, scalability and interoperability or are designed for real-time collaborative use.

Figure 1 shows the rights definition interface which is generated from backend XML Schema files which define the metadata schemes. A graphical user interface is also provided so users can customize the schema files to suit their particular community's descriptive and rights metadata requirements. A keyword web-based search interface is also provided so users may search, browse and retrieve resources that they are permitted to access. Our objective is to incorporate the fine-grained rights management components of the IKM software within the metadata schemas and search interface of the Vannotea system, essentially to enable the IKM software to be used by distributed groups of users, collaborating in real-time.



gender: female

status: [✓ elder] initiated unitiated child

tribe

clan

band

family

role: [✓ artist] [✓ musician]

Fig. 1. Rights definitions using Indigenous Knowledge Management Software

3 System Architecture and Components

Figure 2 illustrates a hypothetical usage scenario of the Vannotea system - a live discussion between museum curators and traditional owners, communicating with each other using Access Grid Nodes and the Vannotea system over the GrangeNet broadband network.

The architecture also reflects the assumption of two separate metadata stores:

– One (or more) databases for the search and retrieval of multimedia content based on objective metadata about the multimedia content. These databases are typically maintained by the custodial organization (museum or archive) or owner of the content.
– A separate metadata store that logs the shared personal annotations.

Figure 2 also illustrates the four major components of the system which are described in more detail below: User Interface; Indexing, Search and Retrieval; Annotation and Discussion Server; Collaborative Environment.

3.1 User Interface

Figure 3 shows the three different components of the user interface. The Content Description component enables the objective and au-

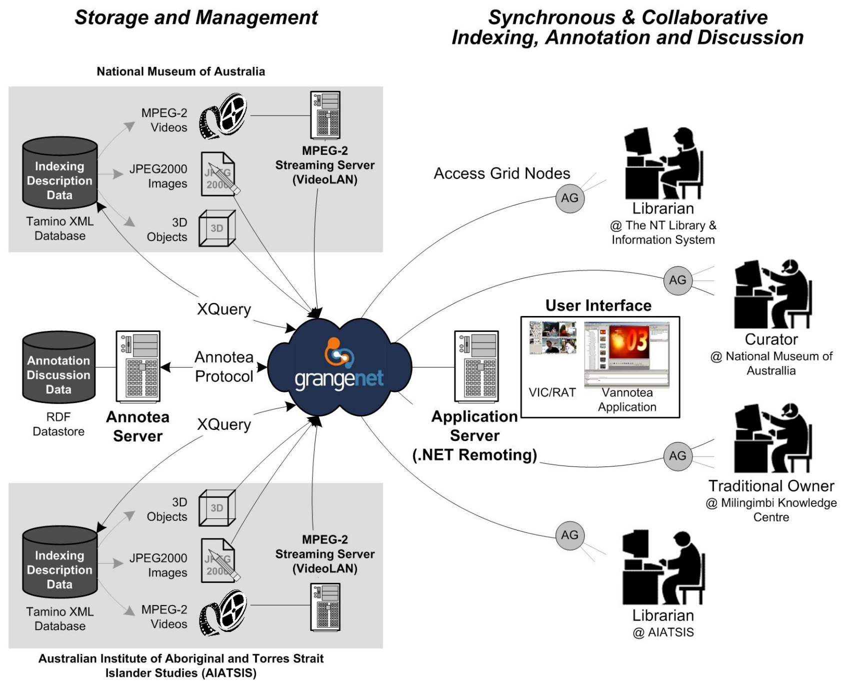

Storage and Management
Synchronous & Collaborative Indexing, Annotation and Discussion
National Museum of Australia
Indexing Description Data
Tamino XML Database
MPEG-2 Videos
JPEG2000 Images
3D Objects
MPEG-2 Streaming Server (VideoLAN)
Access Grid Nodes
AG
Librarian
@ The NT Library & Information System
Curator
@ National Museum of Australia
User Interface
VIC/RAT
Vannotea Application
XQuery
Annotea Protocol
Annotation Discussion Data
RDF Datastore
Annotea Server
grangenet
Application Server (.NET Remoting)
Traditional Owner
@ Milingimbi Knowledge Centre
Librarian
@ AIATSIS
Australian Institute of Aboriginal and Torres Strait Islander Studies (AIATSIS)

Fig. 2. System Architecture

thorized segmentation and indexing of the content, as well as search, browsing and retrieval.

In order to streamline the indexing and segmentation process of video content the Mediaware SDK [36] is used to perform automatic shot- detection. The resulting shot-list is used for further hierarchical segmentation of shots to frames or aggregation of shots to higher-level segments (scenes). This hierarchical temporal structuring enables easy navigation and browsing through the video. The entire multimedia object, selected segments, or individual frames, can be described either by entering free text values or using controlled vocabulary/terms available through pull-down menus.

Different Content Viewers/Players were required to support the different high quality media formats: MPEG-2 for videos, JPEG2000 for images and Direct3D for mesh files. Microsoft Direct3D was chosen for its generic mesh file format and native C# API. Direct3D mesh files describe the model as a series of interconnected polygons. Utilities such as AccuTrans 3D [37] can convert a variety of popular 3D languages, including VRML, into this format.

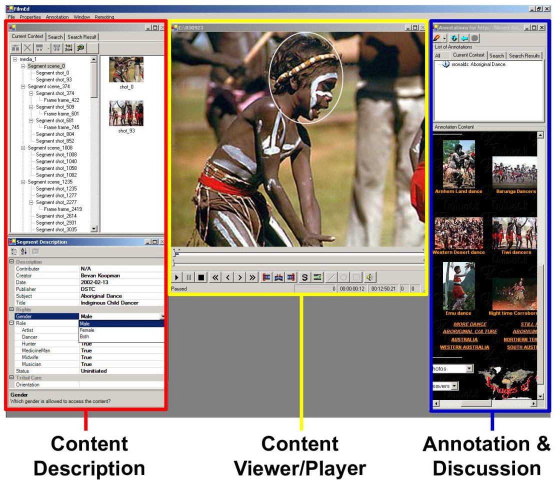


Fig. 3. The three key components of the UI

The controls of each Content Viewer/Player vary depending on the format of the currently shared object. The Video Player features common video playback functionalities (play, pause, seek, stop). The Image Viewer provides tools such as panning, zooming and tilting and the 3D Object Viewer (Figure 4) provides controls to zoom, pan and rotate or change the current view.

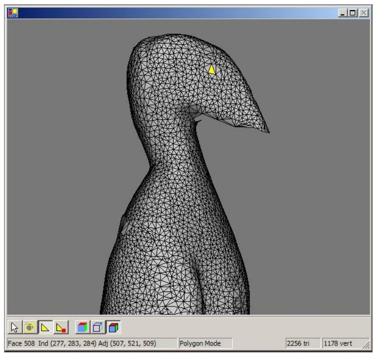


Fig. 4. 3D Object Viewer

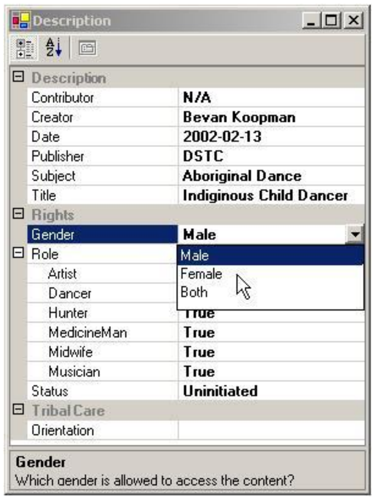
Users can attach annotations to selected regions within images, selected segments, frames or frame regions within videos, or areas within 3D objects. For 2D this is done through drawing simple shapes (line, rectangle, ellipse) on top of the image or frame. The frame number (for video), shape and the coordinates are then stored as context fields the annotation refers to. For 3D objects, annotations can be attached to a specific polygon or group of polygons in the mesh. A 3D technique called picking is used to identify a polygon based on a mouse click. Once annotated, the polygon(s) is/are highlighted to visually inform users that an annotation exists on that surface region of the model. Currently annotations can be either plain text or a URL. Within the Annotation & Discussion window, we not only list the annotations (details of who and when) for a multimedia document, but also provide a search and browsing interface. Consequently, users can not only retrieve content based on the custodial metadata, but also based on the community annotations.

3.2 Indexing & Search and Retrieval Database

A key objective of the system was to provide simplicity and flexibility for users in their choice of metadata descriptions, whilst still supporting standards, interoperability and different media types. This required a design which could easily adapt to the different application profiles required by different communities.

We did this by providing a tool which enables users to define and edit XML Description Templates - simplified versions of XML Schemas. The templates are directly mapped to the UI for entering the metadata (Figure 5). This flexible description architecture allows fast and easy customization of the system to support different indigenous community needs, as well as different media formats.

The actual metadata for each media file is represented as a Description DOM (Document Object Model) similar to the structure of the template, which makes it simple to transform to different standards like Dublin Core [38] and MPEG-7 [39, 40] or the IKM metadata format using XSL-Stylesheets. The metadata is stored in a Tamino native XML database and XQuery used to query the repository. However, the flexible description architecture and search and retrieval interface allows the integration of third party web-based search and retrieval tools such as provided by the IKM project described in Section 2.3.



Description

Description	
Contributor	N/A
Creator	Bevan Koopman
Date	2002-02-13
Publisher	DSTC
Subject	Aboriginal Dance
Title	Indiginous Child Dancer
Rights	
Gender	Male
Role	
Artist	
Dancer	
Hunter	True
MedicineMan	True
Midwife	True
Musician	True
Status	Uninitiated
Tribal Care	
Orientation	

Male
Female
Both

Gender
Which gender is allowed to access the content?

Fig. 5. A metadata capture form generated from a Description Template

3.3 Annotation Server

The annotation database stores the annotations (which may be associated with regions of 2D images or 3D objects, video segments, keyframes, or regions within frames), as well as the source of the annotations (who, when, where). Currently either textual or hyperlink annotations are possible. We based the annotation component of our software on Annotea [19], an open source system developed by the W3C which enables shared RDF annotations to be attached to any Web document or a part of the document. We extended Annotea to support the annotation of other media types, such as audiovisual content or 3D objects through spatio-temporal fragment identifiers generated by extending URI's. For the annotation of regions on 3D objects we used unique polygon IDs. Although this was the simplest approach, it may be problematic when the regions do not exactly match polygon boundaries.

Use of Annotea also allowed us to test prototypical annotation server implementations such as Zope [41] or the W3C Perllib [42] server. We experienced problems with current implementations of Annotea servers which also don't support fine-grained access control to annotations based on user profiles. Therefore we are currently implementing our own annotation server based on the same principles but with more flexibility enabling us to satisfy our needs.

The flexible architecture of Annotea will also allow us to easily attach and store audiovisual annotations - small audio or video clips captured during the video conferencing discussion. However the current access grid node videoconferencing tools (vic and rat) prevent easy capture and synchronization of audio and video streams. Researchers from the ANU's Internet Futures Program are currently working on alternative videoconferencing tools [43] which will support easy capture of audiovisual streams from AGN sessions.

3.4 Collaborative Environment

Because Vannotea is implemented in C# within the .NET development framework, the most flexible, modular and integrated approach to application sharing was to develop it using .NET Remoting. .NET Remoting provides a framework that allows objects to interact with each other across application domains or on different servers through events. Figure 6 illustrates the event-handling architecture of our application. In this example, the client-master is in control of the application, the remote clients are joining the session by connecting to the same server-application.

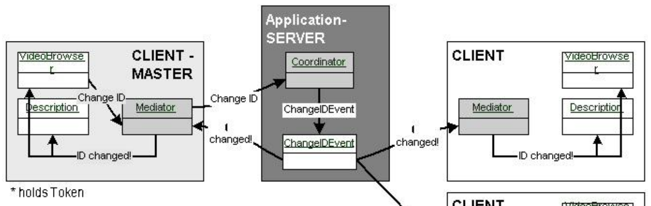


Fig. 6. Event handling using .NET Remoting

The Mediator objects handle the communication between the clients and the server. They can call methods on the remote object (Coordinator). In return, the Coordinator can call methods on the Mediator by raising events that the Mediator has subscribed and listens to. To achieve a form of collaboration, selected events are simulated on all clients. Even mouse movement events can be handled in this way, resulting in several colour-coded mouse pointers within one application that can all be in control simultaneously.

The MPEG-2 videos are streamed using multicast over VideoLan [44]. VideoLan is controlled by the application server to ensure all clients are watching precisely the same content at the same time. Collaborative viewing of 3D objects can be implemented by:

- Either transferring the whole file and rendering it at the client. This is easier to implement; thin server; each client can have a different view. The major disadvantage is that computationally expensive rendering must be performed by each client.
- Or rendering it on the server and streaming the rendered result to each client. This is more efficient because rendering of complex 3D objects is computationally expensive. But it is more difficult to implement and to ensure that each client sees the same view.

One objective of the project is to evaluate users' behavior and obtain user feedback on the different levels of collaboration available during the image / video / 3D object analysis and discussion and annotation processes. We may want to restrict access to shared application controls based on user profiles. Access management of content during collaborative sessions also presents a difficult problem. Participants may have different access rights to content, that will be shared during a collaborative session. If a user chooses to open a new multimedia object, the access rights of each user must be compared with the access rights of the object. If one or more participants are not allowed to see the object, the initiating participant is warned and must choose if the object should be opened or if restricted participants should be excluded from the collaborative session.

4 Future Work and Conclusions

In this paper we have described a system which combines high-speed networks, access grid nodes and collaborative document- and application- sharing software to facilitate the communication and exchange of knowledge between dispersed indigenous communities and museum staff. The aim is to deploy and test the software in collaborative projects between museums, archives and indigenous communities, to facilitate cultural repatriation programs. We are currently discussing a collaborative project between the Smithsonian

National Museum of the American Indian (NMAI), the American Indian Higher Education Consortium (AIHEC) Tribal Colleges and American Indian communities which will use this system to facilitate the implementation of the NMAI's Culturally Sensitive Collections Care Program. It will provide a means for Native people to express and document their concerns with regard to the maintenance, presentation, disposition and repatriation of sensitive materials and information in the museum's collections.

In the immediate future we plan to complete the integration of the IKM software within the Vannotea system and carry out further testing, evaluation and usability studies using real groups communicating via access grid nodes. We also intend to investigate the following:

- Audiovisual annotations - capture of video/audio streams from access grid node sessions;
- Rights management of annotations;
- The use of Shibboleth and OpenSAML to implement user authentication and access controls;
- More intuitive search and browse interfaces, e.g. GIS/map interfaces.

Acknowledgements

The work described in this paper has been funded by the Cooperative Research Centre for Enterprise Distributed Systems Technology (DSTC) through the Australian Federal Government's CRC Programme (Department of Industry, Science and Resources). It has also been funded by the GrangeNet program which is a component of the Australian Federal Government's BITS Advanced Network Initiative (Department of Communications, Information Technology and the Arts). The IKM project is a collaboration with the Smithsonian Institution's National Museum for the American Indian.

References

1. R. Schroeter, J. Hunter, and D. Kosovic, "FilmEd - Collaborative Video Indexing, Annotation and Discussion Tools Over Broadband Networks," in International Conference on Multi-Media Modeling, Brisbane, Australia,, 2004.
2. J. Hunter, B. Koopman, and J. Sledge, "Software Tools for Indigenous Knowledge Management," in Museums and the Web 2003, Charlotte, 2003.

3. A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 1349-1380, 2000.
4. C. G. M. Snoek and M. Worring, "A review on multimodal video indexing," in Multimedia and Expo, 2002. ICME '02. Proceedings. 2002 IEEE International Conference on, 2002, pp. 21-24 vol.2
5. A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 4-37, 2000.
6. C. C. Chibelushi, F. Deravi, and J. S. D. Mason, "A review of speech-based bimodal recognition," IEEE Transactions on Multimedia, vol. 4, pp. 23-37, 2002.
7. J. Fan, A. K. Elmagarmid, X. Zhu, W. G. Aref, and L. Wu, "ClassView : Hierarchical Video Shot Classification, Indexing, and Accessing," IEEE Transactions on Multimedia, vol. 6, pp. 70-86, 2004.
8. M. Worring, A. Bagdanov, J. v. Gemert, J.-M. Geusebroek, M. Hoang, A. T. Schreiber, C. G. M. Snoek, J. Vendrig, J. Wielemaker, and A. W. M. Smeulders, "Interactive Indexing and Retrieval of Multimedia Content," in 29th Annual Conference on Current Trends in Theory and Practice of Informatics (SOFSEM), Milovy, Czech Republic, 2002.
9. S. Marchand-Maillet, "Content-Based Video Retrieval: An Overview," University of Geneva, Geneva, Switzerland Technical Report No. 00.06, CUI 2000.
10. Ricoh, "Movie Tool," June 2003, Available: http://www.ricoh.co.jp/src/multimedia/MovieTool/
11. B. Adams, et al., "Semantic Indexing of Multimedia Content Using Visual, Audio and Text Cues," EURASIP Journal on Applied Signal Processing, 2003.
12. C. Fuller, "Virage Developer Tools Overview," Virage, San Mateo, CA, March 2002.
13. IBM, "MPEG-7 Annotation Tool," April 2003, Available: http://www.alphaworks.ibm.com/tech/videoannex
14. L. Hollink, A. T. Schreiber, J. Wielemaker, and B. J. Wielinga., "Semantic Annotation of Image Collections.," in Workshop on Knowledge Markup and Semantic Annotation, KCAP'03, Florida, 2003.
15. B. B. Yves Lafon, "Describing and retrieving photos using RDF and HTTP." Available: http://www.w3.org/TR/photo-rdf
16. R. a. G. Zhao, W. I., "Negotiating The Semantic Gap: From Feature Maps to Semantic Landscapes," Pattern Recognition, vol. 35, pp. 51-58, 2002.
17. O. a. B. Marques, N, "Semi-automatic Semantic Annotation of Images Using Machine Learning Techniques," in International Semantic Web Conference, Florida, 2003.
18. J. Rowe and A. Razdan, "A Prototype Digital Library For 3D Collections: Tools To Capture, Model, Analyze, And Query Complex 3D Data," in Museums and the Web 2003, Charlotte, 2003.
19. J. Kahan, M.-R. Koivunen, E. Prud'Hommeaux, and R. R. Swick, "Annotea: An Open RDF Infrastructure for Shared Web Annotations," in Proceedings of the WWW10 International Conference, Hong Kong, 2001.
20. J. J. Cadiz, A. Gupta, and J. Grudin, "Using Web annotations for asynchronous collaboration around documents," in Proceedings of the 2000 ACM conference on Computer supported cooperative work. Philadelphia, Pennsylvania, United States: ACM Press, 2000, pp. 309-318.

21. E. F. Churchill, J. Trevor, S. Bly, L. Nelson, and D. Cubranic, "Anchored conversations: chatting in the context of a document," in Proceedings of the SIGCHI conference on Human factors in computing systems: ACM Press, 2000, pp. 454-461.
22. Q. Jin, N. Furugori, K. Hanasaki, N. Asahi, and Y. Ooi, "Collaborative information browser: collecting and organizing information through group collaboration," in 2002 IEEE International Conference on Systems, Man and Cybernetics, 2002, pp. 5 pp. vol.6.
23. W. B. Lober and J. F. Brinkley, "A Portable Image Annotation Tool for Web-based Anatomy Atlases," in Proceedings AMIA Symposium 99, 1999.
24. Jim Ley, "Photo Annotator - Annotating Images with SVG," Available: http://jibbering.com/svg/AnnotateImage.html
25. Virtual Ink Corporation, "mimio classRoom," Available: http://www.mimio.com/meet/classroom/
26. D. Bargeron, A. Gupta, J. Grudin, and E. Sanocki, "Annotations for Streaming Video on the Web: System Design and Usage Studies," Computer Networks, vol. 31, pp. 1139-1153, 1999.
27. M. J. Sipusic, R. L. Pannoni, R. B. Smith, J. Dutra, J. F. Gibbons, and W. R. Sutherland, "Virtual Collaborative Learning: A Comparison between Face-to-Face Tutored Video Instruction (TVI) and Distributed Tutored Video Instruction (DTVI)," Sun Microsystems Laboratories TR-99-72 1999.
28. T. Jung, M. D. Gross, and E. Y.-L. Do, "Annotating and sketching on 3D web models," in 7th International Conference on Intelligent User Interfaces, San Francisco, California, USA, 2002, pp. 95-102.
29. Microsoft, "NetMeeting," January 2002, Available: http://www.microsoft.com/windows/netmeeting/
30. Microsoft Developer Networks, "NetMeeting COM T.120 Application API," Available: http://msdn.microsoft.com/library/default.asp? url=/library/enus/ netmeet/nm3tcom_7z6x.asp
31. T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper, "Virtual network computing," Internet Computing, IEEE, vol. 2, pp. 33-38, 1998.
32. J. Hunter, "Rights Markup Extensions for the Protection of Indigenous Knowledge," in The 11th International World Wide Web Conference - Global Community Track, Honolulu, 2002.
33. "Ara Irititja Archival Project." South Australia. Available: http://waru.org/arairititja
34. "First Voices Project." Available: http://www.firstvoices.com
35. "Special Issue on Digital Technology and Indigenous Communities," in D-Lib Magazine. Available: http://www.dlib.org/dlib/march02/03contents.html
36. "MediaWare Solutions," 2003, Available: http://www.mediaware.com.au/
37. "AccuTrans 3D," MicroMouse Productions. Available: http://www.micromouse.ca/
38. The Dublin Core Metadata Initiative, "Dublin Core Metadata Element Set, Version 1.1: Reference Description," 2003, Available: http://dublincore.org/documents/dces/
39. F. Nack and A. T. Lindsay, "Everything you wanted to know about MPEG-7. 1," Multimedia, IEEE, vol. 6, pp. 65-77, 1999.
40. F. Nack and A. T. Lindsay, "Everything you wanted to know about MPEG-7. 2," Multimedia, IEEE, vol. 6, pp. 64-73, 1999.
41. Zope, "Zope Annotation Server," 2003, Available: http://www.zope.org/Members/Crouton/ZAnnot/

42. W3C, "Perllib Annotations Server HOWTO," 2003, Available: http://www.w3.org/1999/02/26-modules/User/Annotations-HOWTO
43. "VP (Video Presence), Internet Futures group, Australian National University." Available: http://if.anu.edu.au/SW/VP.html
44. VideoLAN, "Overview of the VideoLAN streaming solution," March 2004, Available: http://www.videolan.org/streaming/

eBank UK – Linking Research Data, Scholarly Communication and Learning

Dr Liz Lyon[1], Dr Simon Coles[2], Dr Les Carr[3], Rachel Heery[1], Prof Mike Hursthouse[2], Christopher Gutteridge[3], Monica Duke[1], Dr. Jeremy Frey[2], and Prof Dave De Roure[3]

[1] UKOLN, University of Bath, UK
[2] School of Chemistry, University of Southampton, UK
[3] School of Electonics and Computer Science, University of Southampton, UK

Abstract. This paper presents an overview of the changing landscape of scholarly communication and describes outcomes from the innovative eBank UK project, which seeks to build links from e-research through to e-learning. As introduction, the scholarly knowledge cycle is described and the role of digital repositories and aggregator services in linking datasets from Grid-enabled projects to e-prints through to peer-reviewed articles as resources in portals and Learning Management Systems, are assessed. The development outcomes from the eBank UK project are presented including the distributed information architecture, requirements for common ontologies, data models, metadata schema, open linking technologies, provenance and workflows. Some emerging challenges for the future are presented in conclusion.

1 Introduction and context the scholarly knowledge cycle

The eBank project is predicated on the concept that research and learning processes are cyclical in nature, and that subsequent outputs which contribute to knowledge, are based on the continuous use and reuse of data and information [1]. We can start by examining the creation of original data, (which may be, for example, numerical data generated by an experiment or a survey, or alternatively images captured as part of a clinical study). This initial process is usually followed by one or more additional processes which might include aggregation of experimental data, selection of a particular data subset, repetition of a laboratory experiment, statistical analysis or modelling of a set of data, manipulation of a molecular structure, annotation of a diagram or editing of a digital image, and which in turn generate modified datasets. This newly-derived data is related to the original data and can be re-purposed through publication in a database, in a pre-print or in a peer-reviewed article. These secondary items may themselves be reused through a citation in a related paper, by a reference in a reading list or as an element within modular materials which form part of an undergraduate or postgraduate course. Clearly it will not always be appropriate to re-purpose the original data from an experiment or study, but it is evident that much research activity is derivative in nature.

The impact of Grid technologies and the huge amounts of data generated by Grid-enabled applications, suggest that in the future, (e-)science will be increasingly data-intensive and collaborative. This is exemplified in the biosciences where the growing outputs from genome sequencing work are stored in databases such as GenBank but require advanced computing tools for data mining and analysis. The UK Biotechnology and Biological Sciences Research Council (BBSRC) recently published a Ten Year Vision which describes this trend as "Towards predictive biology" and proposes that in the 21st Century, biology is becoming a more data-rich and quantitative science. The trend has clear implications for data/information management and curation procedures, and we can examine these further by returning to the concept of a scholarly knowledge cycle, figure 1.

A complete cycle may be implemented in either direction so for example, discrete research data could (ultimately) be explicitly referenced in some form of electronic learning and teaching materials. Alternatively, a student might wish to "rollback" to the original research data from a secondary information resource such as a published article or from an element within an online course delivered via a Learning Management System. In order to achieve this, a number of assumptions must be made which relate largely to the discovery process but are also closely linked to the requirement for essential data curation procedures. The assumptions are:

- The integrity of the original data is maintained
- There is a shared understanding of the concept of provenance
- The original dataset is adequately described using a metadata description framework based on agreed standards
- A common ontology for the domain is understood
- Each dataset and derived data and information are uniquely identified (fig. 2)
- Open linking technology is applied to the original dataset and the derived data and information

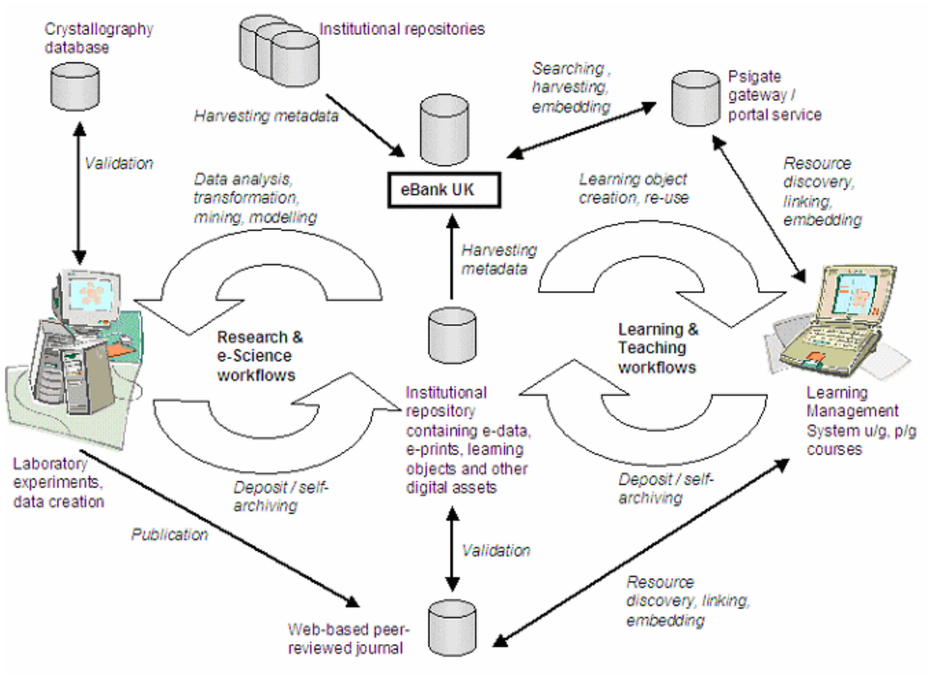

Crystallography database
Institutional repositories
Searching, harvesting, embedding
Psigate gateway / portal service
Harvesting metadata
Validation
Data analysis, transformation, mining, modelling
eBank UK
Learning object creation, re-use
Resource discovery, linking, embedding
Harvesting metadata
Research & e-Science workflows
Learning & Teaching workflows
Institutional repository containing e-data, e-prints, learning objects and other digital assets
Learning Management System u/g, p/g courses
Laboratory experiments, data creation
Deposit / self-archiving
Deposit / self-archiving
Publication
Validation
Resource discovery, linking, embedding
Web-based peer-reviewed journal

Fig. 1. Illustration of the scholarly knowledge cycle for research and teaching

In addition to the Grid computing context, these requirements relate directly to the vision of the Semantic Web, and Semantic Web technologies can be used to express the necessary relationships between objects. The application of Semantic Web technologies within the e-Science and Grid computing context places thiss research in the arena of the Semantic Grid [2].

2 The eBank UK Project - outcomes to date

The eBank UK project is addressing this challenge by investigating the role of aggregator services in linking data-sets from Grid-enabled projects to open data archives contained in digital repositories through to peer-reviewed articles as resources in portals. This innovative JISC-funded project which is led by UKOLN in partnership with the Universities of Southampton and Manchester, is seeking to build the links between e-research data, scholarly communication and other on-line sources. It is working in the chemistry domain with the EPSRC funded eScience testbed CombeChem [3], which is a pilot project that seeks to apply the Grid philosophy to integrate existing structure and property data sources into an information and knowledge environment.

The specific examplar chosen from this subject area is that of crystallography as it has a strict workflow and produces data that is rigidly formatted to an internationally accepted standard. The EPSRC National Crystallography Service (NCS) is housed in the School of Chemistry, University of Southampton, and is an ideal case study due to its high sample throughput, state of the art instrumentation, expert personnel and profile in the academic chemistry community. Moreover, recent advances in crystallographic technology and computational resources have caused an explosion of crystallographic data, as shown by the recent exponential growth of the Crystal Structure Database (CSD) see Cambridge Crystallographic Data Centre. However, despite this rise it is commonly recognized that approximately only 20reaching the public domain. This situation is even worse in the high throughput NCS scenario where approximately 15disseminated, despite producing ¿60 peer reviewed journal articles per annum. With the imminent advent of the eScience environment, this problem can only get more severe.

A schema for the crystallographic experiment has been devised that details crystallographic metadata items and is built on a generic schema for scientific experiments, figure 2. During the deposition of data in a Crystallographic EPrint metadata items are seamlessly extracted and indexed for searching at the local archive level. The top level document includes 'Dublin Core bibliographic' and 'chemical identifier' metadata elements in an Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) compliant form, which allow access to a secondary level of searchable crystallographic metadata items, which are in turn directly linked to the associated archived data. In this manner the output from a crystallographic experiment may be disseminated as 'data' in such a way that aggregator services and researchers may add value to it and transform it into knowledge and the publication 'bottleneck' problem can be addressed.

The metadata about datasets made available in the repository will be harvested into a central database using the OAI-PMH. This metadata will then be indexed together with any other available metadata about research publications. A searchable interface will enable users to discover datasets as well as related literature. The metadata contains links back to the datasets which the user will be able to follow in order to obtain access to the original data, when this is available. Harvested metadata from different repositories not only provides a common entry point to potentially disparate resources (such as datasets in dataset repositories and published literature which may reside elsewhere) but also offers the potential of enhancement of the metadata such as the addition of subject keywords to research datasets based on the knowledge of subject classification terms assigned to related publications. A further area of work investigates the embedding of the search interface within web sites, adopting their look-and-feel. PSIgate (http://www.psigate.ac.uk/) will be used to pilot these embedding techniques based on CGI-mechanisms and portal-related standards.

The concept we have implemented within the Southampton e-print archive system is Data Publication@Source [4]. Crystallographic EPrints use the OAI concept to make available ALL the data generated during the course of a structure determination experiment. That is the publishable output is constructed from all the raw, results and derived data that is generated during the course of the experiment. This presents the data in a searchable and hierarchical system that relates to the workflow of the experiment. This metadata includes bibliographic and chemical identifier items which are above a secondary level of searchable crystallographic items which are directly linked to the associated archived data. The table below depicts the schema and shows the hierarchical manner in which the open archive report is constructed in the figure below.

Hence the results of a crystal structure determination may be disseminated in a manner that anyone wishing to utilise the information may access the entire archive of data related to it and assess its validity and worth. In the future a notification, or bulletin board type system, (much like Amazons reviewers comments) could be added to create in effect a new type of peer reviewed publication.

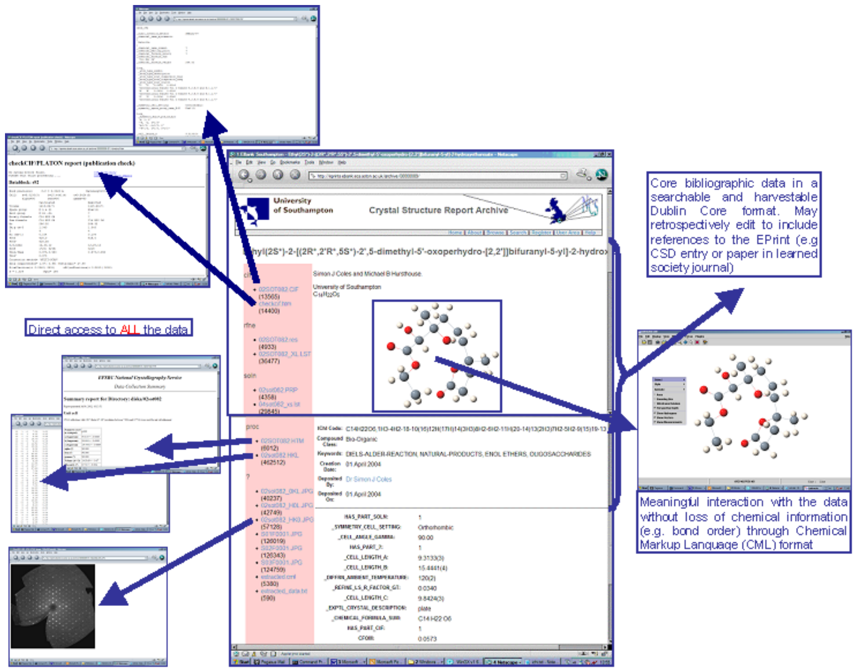
3 Conclusions - issues and challenges for the future

While only test versions of the e-crystal-data-report have been produced so far they have been populated with real crystallographic data sets produced at the National Crystallography Service (NCS).

Name	Files associated with this stage			Metadata associated with this stage	
	File	Type	Description	Name	Data Type
Initialisation	.htm	HTML	Metadata for crystallography expt	Morphology	*STRING (SET)
	i*.kcd	BINARY	Unit cell determination images	Solvent	*STRING
			Unit cell	Sample_image	.JPG
Collection	s*.kcd	BINARY	Diffraction images	Instrument_Type	*STRING
	scan.jpg	JPG	Visual version of .kcd file	Temperature	*INTEGER
				Crystal_image	.JPG
				Software_Name	STRING
				Software_Version	INTEGER
Processing	scale_all.in	ASCII	Result of processing	Cell_a	*NUMBER
	scale_all.out	ASCII	Result of correction on processed data	Cell_b	*NUMBER
	.hkl	ASCII	Derived data set	Cell_c	*NUMBER
	.htm	HTML	Report file	Cell_alpha	*NUMBER
				Cell_beta	*NUMBER
				Cell_gamma	*NUMBER
				Crystal_system	*STRING (SET)
				Completeness	*INTEGER (%)
				Software_Name	STRING
				Software_Version	INTEGER
Solution	.prp	ASCII	Symmetry file, log of process	Space_group	*STRING (SET)
	xs.lst	ASCII	Solution log file	Figure_of_merit	*NUMBER
				Software_Name	STRING
				Software_Version	INTEGER
Refinement	xl.lst	ASCII	Final refinement listing	R1_obs	*NUMBER
	.res	ASCII	Output coordinates	wR2_obs	*NUMBER
				R1_all	*NUMBER
				wR2_all	*NUMBER
				Software_Name	STRING
				Software_Version	INTEGER
CIF	.cif	ASCII	Final results	Formula_moiety	*STRING
	checkcif.htm	HTML	Automatic validation results	CIF_check	*STRING
Report	.html	.HTML	Publication format (HTML/XHTML)	EPrint_type	*CRYSTAL STRUCTURE
				Authors	*STRING
				Affiliations	*STRING
				Formula_empirical	*STRING
				Compound_name	*STRING
				CCDC_Code	*STRING
				Compound_class	*STRING (SET)
				Keywords	*STRING (SET)
				Available_data	*STRING (SET)
				Related_publications	STRING

Fig. 2. The draft version of the schema details for the crystallographic data showing the main data types and associated file names generated in a crystal structure determination.

The lessons learnt so far fall in to three main areas. First, the ease of use in inputting the data to the system and ensuring that sufficient information about the materials, the experiment and the nature of the data is being entered. Some of the input styling is taken over directly from the current e-print system which of course allows for files of different types to be up loaded. However the issue of file types becomes more complex with the advent of data. In this regard the conformity of the crystallography community is an advantage, with common file types with well-understood extensions. Never the less some standardization is imposed following the NCS practice at this stage. Conversion between different file types for the higher-level data elements (e.g. the molecular structure files) can be done automatically, with a high degree of fidelity, and this is necessary to provide suitable data for the visualization programs built in to the viewing interface.



Core bibliographic data in a searchable and harvestable Dublin Core format. May retrospectively edit to include references to the EPrint (e.g CSD entry or paper in learned society journal)

Direct access to ALL the data

Meaningful interaction with the data without loss of chemical information (e.g. bond order) through Chemical Markup Language (CML) format

University of Southampton — Crystal Structure Report Archive

Home | About | Browse | Search | Register | User Area | Help

...hyl(2S*)-2-[(2R*,2'R*,5S*)-2',5-dimethyl-5'-oxoperhydro-[2,2']]bifuranyl-5-yl]-2-hydrox...

Simon J Coles and Michael B Hursthouse

University of Southampton

$C_{14}H_{22}O_6$

cif
- 02SOT082.CIF (13565)
- checkcif.htm (14400)

rfne
- 02SOT082.res (4933)
- 02SOT082_XL.LST (36477)

soln
- 02sot082.PRP (4358)
- 04sot082_xs.lst (29845)

proc
- 02SOT082.HTM (6012)
- 02sot082.HKL (462512)

?
- 02sot082_0KL.JPG (40237)
- 02sot082_H0L.JPG (42749)
- 02sot082_HK0.JPG (57128)
- S01F0001.JPG (126019)
- S02F0001.JPG (126343)
- S03F0001.JPG (124759)
- extracted.cml (5380)
- extracted_data.txt (590)

ICM Code:	C14H22O6.1H3-4H2-18-10(16)12H(17H)14(3H3)8H2-6H2-11H(20-14)13(2H3)7H2-5H2-9(15)19-13
Compound Class:	Bio-Organic
Keywords:	DIELS-ALDER-REACTION, NATURAL-PRODUCTS, ENOL ETHERS, OLIGOSACCHARIDES
Creation Date:	01 April 2004
Deposited By:	Dr Simon J Coles
Deposited On:	01 April 2004

HAS_PART_SOLN:	1
_SYMMETRY_CELL_SETTING:	Orthorhombic
_CELL_ANGLE_GAMMA:	90.00
HAS_PART_?:	1
_CELL_LENGTH_A:	9.3133(3)
_CELL_LENGTH_B:	15.4441(4)
_DIFFRN_AMBIENT_TEMPERATURE:	120(2)
_REFINE_LS_R_FACTOR_GT:	0.0340
_CELL_LENGTH_C:	9.8424(3)
_EXPTL_CRYSTAL_DESCRIPTION:	plate
_CHEMICAL_FORMULA_SUM:	C14 H22 O6
HAS_PART_CIF:	1
CFOM:	0.0573

Fig. 3. A summary of the different screens available from the basic e-data report showing how the data and metadata described in the basic schema of figure 2 are displayed.

This brings us on to the second main area, the viewing of this information directly from the e-print system. Once an entry is located then the information is presented on a web interface with the major details about the molecule available, including a visual, rotateable image. The conversion of the crystallographic CIF file to other formats (e.g. MOL file) is necessary for this. Similarly the newly define unique chemical identifier (INCHI) is also calculated by converting the CIF file via these stages to a CML file. The required conversions are implemented (or soon will be) as web services to allow updates independently of the main e-print software system.

The final issues revolve around finding the existence of the data on the e-print system. As indicated above the data schema that provides the outline for the arrangement of the data highlights the significant data that a chemist would wish to search on. This is made available via an OIA interface to harvesting programs, which can then provide additional functionality to enable a multi-parameter search.

This side of the system is less tested and does present some problems for us as many of the current search systems allow for searches on chemical structures (drawn in 2 or 3D) using proprietary algorithms, especially for the sub-structure search. We are thus not able to implement this type of search but can demonstrate that the data needed for such searches can be made available to data aggregators.

The test experiments have proved very successful and have engaged the interest of several of the traditional suppliers of crystallographic information, who wish in effect to move up market in the information supply chain. In the academic community there is a growing word wide interest. The system of separating the archiving of the data, including all the raw and background information, from the aggregated higher level structural data which is to be curated over a much longer term, looks to be very successful.

We should stress that the e-experiment-data-reports are not restricted to crystallographic data. These data sets were chosen as an ideal test case due to their relative uniformity, community agreement and availability of diagnostics to enable the user to assess the quality of the data. We are now extending the system to cover spectroscopic data, for which again the issue of the lack of extensive libraries of even common molecules is a major hindrance to efficient research. We are applying the principle to Raman spectra are working with one of the spectrometer supplies from the start to ensure that the links to the e-print system can be built in to the spectrometer software, further simplifying the task of disseminating the data.

One aspect that may need further consideration if the automated processes are enhanced, is finer control over access to the data by different groups at different times. This will of course overlap with security concerns.

References

1. Lyon, L.: Developing Information Architectures - to support Research, Learning and Teaching. In: UCISA Conference. (2002)
2. De Roure, D., Jennings, N.R., Shadbolt, N.R.: The Semantic Grid: A future e-Science infrastructure. In: Grid Computing - Making the Global Infrastructure a Reality. John Wiley and Sons Ltd (2003) 437–470
3. Frey, J.G., Bradley, M., Essex, J.W., Hursthouse, M.B., Lewis, S.M., Luck, M.M., Moreau, L., De Roure, D.C., Surridge, M., Welsh, A.: Combinatorial chemistry and the Grid. In: Grid Computing - Making the Global Infrastructure a Reality. John Wiley and Sons Ltd (2003) 945–962

4. Frey, J., De Roure, D., Carr, L.: Publishing at Source: Scientific Publication from a Web to a Data Grid. In: EuroWeb 2002 Conference, Oxford (2002)
5. Bearman, D., Lytle, R.: The Power of the Principle of Provenance. Archivaria (**21**) 14–27
6. Hitchcock, S., Brody, T., Gutteridge, C., Carr, L., Hall, W., Harnad, S., Bergmark, D., Lagoze, C.: Open Citation Linking: The Way Forward. D-Lib Magazine **8** (2002)

The Integration of Peer-to-peer and the Grid to Support Scientific Collaboration

Tran Vu Pham, Lydia M. S. Lau, and Peter M. Dew

{tranp,llau and dew}@comp.leeds.ac.uk
School of Computing, University of Leeds, Leeds, UK

Abstract. There have been a number of e-Science projects which address the issues of collaboration within and between scientific communities. Most effort to date focussed on the building of the Grid infrastructure to enable the sharing of huge volume of computational and data resources. The "portal" approach has been used by some to bring the power of grid computing to the desk top of individual researchers. However, collaborative activities within a scientific community are not only confined to the sharing of data or computational intensive resources. There are other forms of sharing which can be better supported by other forms of architecture. In order to provide a more holistic support to a scientific community, this paper proposes a hybrid architecture, which integrates Grid and peer-to-peer technologies using Service Oriented Architecture. This platform will then be used for a semantic architecture which captures characteristics of the data, functional and process requirements for a range of collaborative activities. A combustion chemistry research community is being used as a case study.

1 Introduction

Research on infrastructure to support scientific collaboration has attracted the attention of many institutions and organisations worldwide. The UK e-Science programme and its research centres are amongst the most active participants[1]. E-Science projects, such as myGrid[2] or DAME[3], have mainly focused on developing middleware to support scientific collaboration commonly known as the Grid. The main purpose of the Grid is for *coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organisations*[4]. Grid developer community recently has adopted a Service Oriented Architecture, OGSA[5], to enhance the interoperability of services on the Grid and the flexibility when using the services. However, the challenge is how to bring the benefits of the Grid to the wider scientific communities (i.e. those which may need the power of the Grid only occasionally). The current portal approach provides a gateway to the Grid power from a desktop machine, e.g. in [3], but its support for interactive collaboration amongst end users is rather limited.

Peer-to-peer, on the other hand, is a computing model, which has a more lightweight approach to the sharing of computing resources. This model has been proved to be successful in many commercial desktop file-sharing applications such as Napster[1] and currently Kazza[2]. The advantage of peer-to-peer is that its application is closer to end users, and when using the system, they have the sense of ownership over their shared resources. In addition, a peer-to-peer application often provides online means of communication to support collaborative work, therefore, not only computing resources but also scientific knowledge could be shared. It is anticipated that the world of scientific computing will be more and more decentralised into peer-to-peer model, where scientists' desktops will be edges of the network[6].

In the effort to improve the support for scientific collaboration, an integrated architecture combining the Grid and peer-to-peer concepts is proposed. The goal of this integration is to bring resources on the Grid more widely available to the whole scientific community. In order to increase automation and quality of collaboration within the new architecture, the use of Semantic Web technology is planned.

The domain of experimentation for this study comes from the Combustion Chemistry Research Community. Requirements capture has been an on-going activity. The next section of this paper will provide the background of collaborative requirements within the Combustion Chemistry Research Community. The proposed hybrid architecture will be presented in the third section. The potential offered by the Semantic Web technology and its challenges will be assessed.

2 Context: The Combustion Chemistry Research Community

The centre of Combustion Chemistry research is building models of chemical reactions. This activity is time-consuming and requires expert knowledge. However, because the input data necessary for the building process, experimental data and reaction rate coefficients, is

[1] Napster has been put out of service
[2] http://www.kazza.com

scattered around in the community and improperly evaluated, the model builders do not have access to all of these data, and hence, the accuracy of resulted models are limited to only particular conditions. Consequently, subsequent combustion processes that use these models are also limited to certain level of accuracy. In order to overcome these shortcomings and to get model users involved in building process, the combustion chemistry community is looking for a computing infrastructure, which has:

– A storage for storing knowledge relevant to compiling reaction models, such as experiment data, as well as reaction models.
– A collection of tools for enabling scientific collaboration amongst all distributed participants and tools for enabling processing, analysis and validation of data as well as assembly the data into models.

The ultimate goal of the Combustion Chemistry Research Community is a paradigm that enables building reaction models in a consistent and systematic way by incorporating all available data and expertise of all members of the community.
(The problems and requirements above identified by chemists in[7], [8] & [9])

3 The Integration of Peer-to-peer and the Grid

Integration between Grid and peer-to-peer has been considered in Peer-to-peer Grids[10], which mixes Grid and peer-to-peer concepts into a democratic architecture mediated by Web Services. This paper proposes a different way of integrating the Grid and peer-to-peer, which attaches peer-to-peer and the Grid together (Fig. 1) so that they can mutually support each other. This method of integration reduces the cost of implementation and management while still maintaining flexibility as characteristics of the Grid and of peer-to-peer will remain the same.

As shown in Fig. 1, the integrated architecture separates heavy computation and storage into the Grid side, named computation layer, and lightweight collaboration into peer-to-peer side, named collaboration layer. For example, in the context of Combustion Chemistry research, heavy computation can be simulations of reaction mechanism; sharing of experimental data and mechanisms amongst

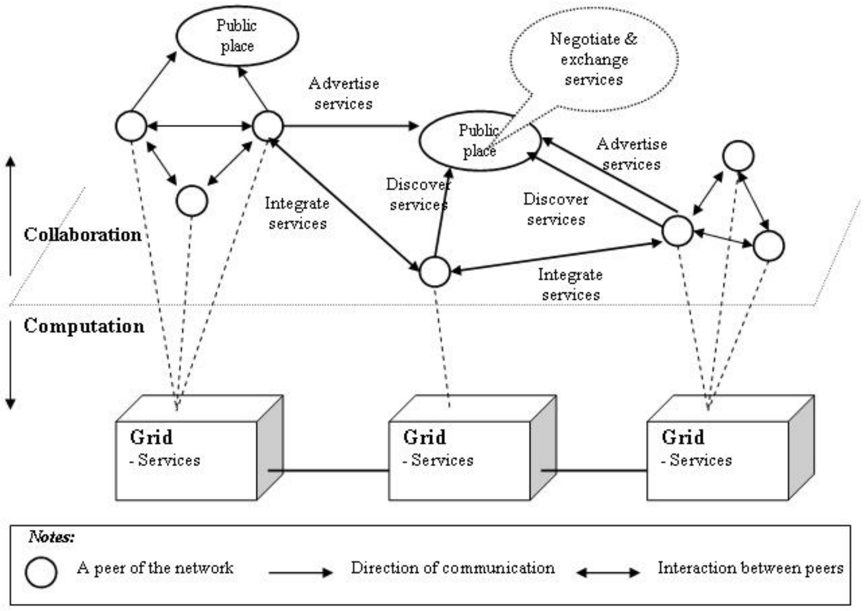


Fig. 1. The integration of Peer-to-peer and the Grid

scientists are lightweight collaboration. The two layers can be connected by either Web or Grid Services that depends on the choice of implementation.

Information about resource on the traditional Grid such as experimental data, simulation programs, is only known and accessible to authorised users. However, research communities are widely distributed, and not all members are granted access to Grid. Therefore, there might be a situation in which some scientists are desperate for resource, whereas the resource is readily available on the Grid with minimal use. The proposed integration is addressing this problem in the following way: (i) resources on the Grid are transformed into Web (Grid) services; (ii) the owners of these services publish service advertisements into their peer-to-peer communities; (iii) on receiving information about necessary services, scientists, who do not have access to Grid, can request service owners to execute the services on their behalf. In this model, the role of resource owners is to bridge the need of peer-to-peer communities and resources on the Grid. This role will be automated and can be extended to authorised Grid users, who are also members of peer-to-peer communities.

In addition to the above advantage, the separation of computation to the Grid and lightweight collaboration into peer-to-peer also reduces the management complexity of trivial collaboration on the Grid. Other features of peer-to-peer computing such as instant messaging, file sharing also add great values to the integrated architecture in supporting scientific collaboration.

4 Potential Application of the Semantic Web

The above section has described in principle the interaction of Grid and peer-to-peer in the collaborative architecture. A semantic architecture would also be necessary to bring the architecture to its full potential as well as to satisfy strict requirements of modern scientific communities. For instance, Combustion Chemistry Research Community requires of higher level of automation in building chemical mechanism and locating validated source of scientific data. In order to achieve this, characteristics of scientific collaborative processes and data need to be captured and categorised in a proper format that is understandable to computer programs. This requires the use of ontology from the Semantic Web technology. One example from the architecture, the discovery of available services on the Grid in peer-to-peer environment will clearly benefit from the use of ontology.

The use of ontology in a decentralised peer-to-peer environment leads to a challenging problem. As ontology is a means of capturing scientific knowledge and knowledge in a scientific community evolves overtime, it will not be feasible to have only a common static ontology for a community. Ontology building has to be a continuous process. It will also not appropriate for some people to develop the ontology, and the other people using it. The one who uses the ontology is the one who has most knowledge of it, then that one should contribute to development of the ontology. In addition, with one piece of ontology, communities from different backgrounds may understand differently. Therefore, ontologies should be local to communities that it is built to support.

For all the above reasons, this paper is proposing an approach to be used within the integrated architecture, in which ontology within a community will be used and contributed by its members. In order to achieve this, the peer-to-peer environment will be organised as communities and sub-communities. Each community or sub-community will promote a leader or a chairperson, who leads the community, and agree on a common ontology to be used. A member of a community can make change to the ontology of the community by advertising definition of new terms and concepts to all other members of that community. Other members will give feedback if appropriate. When all the agreements are made on the new ontology, the chairperson will make the change official to the community and notify other members about the decision.

5 Conclusion and Work to Be Done

This paper has introduced an integrated architecture between Grid and peer-to-peer to support scientific collaboration, focusing more on the collaboration aspect with the aim to bring large-scale resources a step closer to end scientist users. The use and the management of ontologies have also been considered to exploit full potential of the collaboration within and amongst communities in peer-to-peer environment.

A prototype implementation of the integration in the context of Combustion Chemistry Research Community without ontology has been developed and proved to be successful. This implementation is using Grid Services from Globus Toolkit 3[11] and JXTA[12] on Grid and peer-to-peer side respectively. The concept Peer Group and messaging mechanism of JXTA seems to match well with the integrated architecture. The next implementation will be incorporating ontology into collaboration layer to support resource discovery, focusing particularly on discovering services, and managing ontologies within communities. This implementation promises to bring a novel architecture into reality.

References

1. National e-Science centre, http://www.nesc.ac.uk/ (2004) *(NeSC's website).*
2. myGrid, http://www.mygrid.org.uk (2004) *(myGrid project's website).*
3. DAME: Distributed aircraft maintenance environment, http://www.informatics.leeds.ac.uk/pages/03_research/rp_dame.htm (2002) *(DAME project's website).*
4. Foster, I., Kesselman, C., Tuecke, S.: The anatomy of the grid: Enabling scalable virtual organisations. International J. Supercomputer Applications **15(3)** (2001)

5. Foster, I., Kesselmana, C., Nick, J., Tuecke, S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum. (2002)
6. Foster, I., Iamnitchi, A.: On death, taxes, and the convergence of peer-to-peer and grid computing. In: 2nd International Workshop on Peer-to-Peer Systems(IPTPS'03), Berkeley, CA (2003)
7. Dixon, D.A., Rahn, L.A.: Advancing chemical science: Future networking requirements. In: High Performance Network Planning Workshop, Office of Science, US Department of Energy (2002)
8. Bair, R., et al.: Report of the high-performance network planning workshop. In: High Performance Network Planning Workshop, Office of Science, US Department of Energy (2002)
9. Process Informatics Model, http://www.me.berkeley.edu/prime/ (2004) *(PrIMe information website)*.
10. Fox, G., Gannon, D., Ko, S., Lee, S., Pallickara, S., Pierce, M., Qiu, X., Rao, X., Uyar, A., Wang, M., Wu, W.: Peer-to-peer Grids. In Berman, F., Fox, G., Hey, T., eds.: Grid Computing - Making the Global Infrastructure a Reality. John Wiley & Sons Ltd (2003) 471–490
11. The Globus Toolkit, http://www-unix.globus.org/toolkit/ (2004) *(The Globus Alliance website)*.
12. Project JXTA, http://www.jxta.org (2003) *(Project JXTA website)*.

Semantic Annotation of Computational Components

Peter Vanderbilt[1] and Piyush Mehrotra[2]

[1] AMTI*, NASA Ames Research Center, M/S T27A-2, Moffett Field, CA 94035
pv@nas.nasa.gov
[2] NASA Ames Research Center, MS 258-5, Moffett Field, CA 94035
Piyush.Mehrotra@nasa.gov

Abstract. This paper describes a methodology to specify machine-processable semantic descriptions of computational components to enable them to be shared and reused. A particular focus of this scheme is to enable automatic composition of such components into simple workflows.

1 Introduction

Part of the vision of the Semantic Grid is to enable "an infrastructure where all resources, including services, are adequately described in a form that is machine-processable"[1]. This paper describes a methodology to specify machine-processable semantic descriptions of computational components.

The focus of the CRADLE project is to represent the semantics of workflow components, or so called "tools," with the ideal goal of enabling automatic generation of semantically correct workflows. A prototype of CRADLE has been implemented including a repository for tool descriptions, a plan (or workflow) generation program and a prototype plan execution system. While the focus of CRADLE is on tools, we think that similar techniques apply to data collections and web/grid services.

In general, a *tool* is a program, component or service that computes one or more outputs from one or more inputs. Some tools require significant computation such as a simulation that computes a flow field around some object or one that computes thermodynamic properties given a flow. However, tools also include format translators, data extraction functions, data analysis and visualization programs.

* This work is supported by the NASA Advanced Supercomputing Division under Task Order A61812D (ITOP Contract DTTS59-99-D-00437/TO #A61812D) with Advanced Management Technology Incorporated (AMTI).

In order to manage the growing number of these tools and to share them, it is desirable to put tool descriptions in web-accessible, searchable *repositories*. Associated with each tool would be information about what it does, how to run it, who created it, its properties and so on.

The main focus of our research is to find ways to express the *semantics* of a tool in a machine-readable way. Semantic tool descriptions could be used to narrow the set of tools that a user must select from to solve a particular problem. Further, we are interested in *plans*, collections of interconnected tools, and how to use semantic descriptions to ensure the well-formedness of a plan and to determine its semantics. Plans are a kind of *workflow* that have been restricted to DAGs (directed acyclic graphs).

Our goal is to facilitate *plan generation*, which is a process that creates a plan whose execution produces a specified result given specified inputs. Plan generation uses tool descriptions to determine feasible combinations of tools. In order to push the research, we take as the ideal goal the automatic ("hands-off") generation of plans. However, we recognize that, for various reasons, user input may be needed to select among several combinations of tools.

This paper describes CRADLE's *dataspace* approach. A dataspace is an abstraction that allows one to provide meaningful, compound names to various properties of real or logical entities. We present how this approach can be used to specify descriptions of computational components and also to chain together these components in semantically consistent ways to yield simplified workflows.

2 Tool and Problem Specifications

Consider the situation where a scientist is computationally investigating the flow characteristics of a class of aerodynamic structures. For a given system of a body and external characteristics, there may be several potential properties that could be calculated. Assume there are tools that take various properties (radius, angle, velocity) and generate datasets (such as meshes), tools that take datasets to other datasets (such as other meshes and flow simulations), tools that analyze datasets (yielding floats, integers and booleans) and tools that convert between the different dataset formats used by these tools.

The scientist needs a way to compose these tools depending on what is given and what is to be calculated. In order to do this, there must be ways to describe tools and the problems that users want to solve.

Consider a simplified example of a "body" whose properties include its geometry file, volume, mass, (average) density, velocity and momentum. A "problem" specifies which properties will be provided (the "givens") and which will be calculated (the "goals"). At different times, there will be different sets of givens and goals. Assume that the scientist has at his disposal the following "tools":

1. Three tools for calculating each of momentum, velocity or mass of a body, given the other two. (There is one tool for each unknown).
2. Three tools for calculating each of average density, mass or volume, given the other two.
3. A tool that takes a body's geometry file and yields its volume.
4. A tool that calculates the mass and volume of a system of bodies, each with mass and momentum.

So how does one specify these tools and how would the scientist specify what data he wants (given what data he has)?

In a programming language, a tool would be a method or function. In WSDL[5], such a tool would be a web service[4] operation. In these two systems, the machine-readable aspects of a "tool" description are the types of its inputs and outputs. A type system ensures that a composition of tools is consistent at the type level.

While type correctness is required, it is not sufficient for our purposes. For instance, the first three tools mentioned in item #1 above, take two floats and return a float (assuming velocity and momentum are scalars) and are thus indistinguishable from the point of view of the types. Of course, the problem is that the semantics of each tool is not taken into account. Generally, the semantics of a tool is expressed as a comment and/or is implied by the name of the function and the names of its inputs. It is difficult to effectively reason about information in this form.

Another possible method is to require that the function name indicate the quantity being produced. Unfortunately, there may be tools with more than one output. Item #4, above, is an example of such a tool that generates both mass and momentum.

One can associate one or more output names with each tool. However, there may be more than one tool producing the same output.

For instance, there are three tools yielding mass in our example. While the types are different, one could imagine examples where the types are the same. A way to solve this problem is to treat the input names as significant (as well as the output names).

While this approach goes a long way, there are still a few problems. One is that a name alone may not adequately specify a quantity and its representation. For instance, if a velocity-producing tool and a velocity-using tool use different units or formats, the tools will not interoperate. Basically, this is a human issue – the system can compare names but only humans can ensure that the names are used consistently. Thus it is important that each name have an associated natural language description that, together with its type, unambiguously describes the associated quantity and its format.

There is also a problem when two different users unintentionally use the same term with different meanings. While it is possible for a system to detect conflicting definitions, it is better to have a name qualification scheme such that two independent people use different qualifiers for the names they define. Examples of such schemes include UUIDs, URIs, XML QNames and Java class names.

A related problem arises, for example, when one tool uses "width" while another uses "breadth" for the same concept. In this case, the two tools will be deemed incompatible when really they are compatible. What would help is a way to equate two names.

There is also a problem with the "flatness" of names. Consider a simulation of several interacting physical bodies, each with velocity, a mass and momentum. In this case, a simple name, like "velocity," is ambiguous, since each body has a velocity. One could use names like "b1_velocity" and "b2_velocity" but then a tool taking "velocity" as an input will not apply. What is needed is a notion of compound names, or *paths*. In the system above, the bodies could be named "b1" and "b2" and their velocities would be named by "b1.velocity" and "b2.velocity" (where the period combines two names). Tools apply as if universally quantified over paths, so the tool described above also can take inputs "b1.velocity" and "b1.mass" and yield "b1.momentum."

A final problem is illustrated by considering the computation of the area of rectangles and ellipses. Both have "height," "width" and "area" properties, all floats, but the tools used to compute their areas must be different. The issue is that property names alone do not determine the semantics of the object of which they are a part. Thus there needs to be a way to associate a tool with a class of objects.

3 The Dataspace Model

In this section, we describe an abstract model called the *dataspace model* that addresses the issues of the previous section. Briefly, a dataspace is a tree-like structure with named slots as leaves. A slot can be thought of as a place where a data value can be deposited. Dataspaces have types that imply a vocabulary of slot names and their semantic interdependencies. Each tool is associated, via a relation called "appliesTo," with one type of dataspace. Logically when a tool runs, it is passed a dataspace of the appropriate type; the tool retrieves its inputs from certain slots and places its outputs in other slots. Roughly, two tools can be composed only if they both have the same "appliesTo" type and the names of the inputs of one are among the names of the outputs of the other. We now discuss the model in more detail.

A dataspace is used to model some real world or logical *entity*, such as a physical body (of various shapes), a surface, a flow field, a particle or a galaxy. An entity can also be a composite thing like a system of several bodies or a flow interacting with the surface of some structure.

A dataspace is made up of a collection of named *slots* each capable of holding one piece of information, like a float, an array of floats or a filename. Each slot is either empty or filled and, when filled, its content denotes one *aspect* of the entity being modeled. An aspect is some parameter, attribute, property or view of an entity. Example of aspects are a body's mass, velocity or a reference to a file containing its geometry. Different aspects can be used for different representations or units for the same property.

It is possible that an aspect of an entity is itself a composite entity, in which case the aspect is represented by another dataspace, refered to as a *subdataspace*. In general a dataspace forms a tree with named edges and slots at the leaves. For example, a velocity aspect might be a vector modeled by a subdataspace with x, y and z aspects. Similarly a system with two bodies could be modeled as two subdataspaces named “b1” and “b2.” In this case, “b1.velocity.x” names a slot that contains the x component of b1’s velocity.

Each slot or subdataspace is considered one aspect and is given an *aspect name*. A compound name, like “b1.velocity.x,” is called an *aspect path*.

Aspects are typically interdependent and, so, the values of certain slots can be computed from others. For example, a physical body might have aspects mass, geometry, velocity, volume, average density and momentum. Given any two of mass, velocity or momentum, the third can be calculated. Volume, mass and density are in a similar relationship and, presumably, volume can be computed from geometry.

A *dataspace type* denotes a set of dataspaces and is typically associated with some class of entities. A dataspace type defines a vocabulary of aspect names and their associated types and interpretation. The type also denotes a set of constraints on the values of aspects and their interdependencies. These semantic properties are given explicitly by an associated description string or are implied by the names of the type and its aspects. Essentially, the type name becomes a proxy for these human-understood semantics.

Consider the following example definition

```
dataspace Body {
    aspect URL geometryFile;
    aspect Float volume;
    aspect Float mass;
    aspect Float density;
    aspect Float velocity;
    aspect Float momentum;
}
```

While CRADLE actually uses an XML syntax, a more convenient syntax like this is better for explanatory purposes. This definition defines a type named “Body” with six aspects. The first aspect is named “geometryFile” and is of type “URL.” The remaining five aspects are of type Float with names “volume,” “mass,” “density,” “velocity” and “momentum.” We assume “URL” and “Float” are defined elsewhere. The interdependencies between aspects are implied by the aspect names. A description string could be added to the definition if further explanation was needed.

Each dataspace type definition defines a new, independent type. Even if the aspects are identical, it is assumed that their interdependencies are different, as implied by the name of the type or its description. For instance, there could be two type definitions with identical aspects, "height," "weight" and "area," but having different names, "Rectangle" and "Ellipse." They would denote different types.

The CRADLE type system supports inheritance where inheritance implies an "isa" or subset relation – instances of a derived type are instances of the base type. A derived type has all the aspects of the supertype and can add new aspects, refine existing aspects and add additional constraints (between aspects). For example, Square could be a subtype of Rectangle, adding the constraint that the "height" and "width" aspects are the same. An aspect is *refined* if the derived aspect's type is a subtype of the base aspect's type and if any description-implied constraints of the derived aspect imply the corresponding constraints of the base aspect.

Now that the dataspace model has been described, we turn to CRADLE tool descriptions, which use the dataspace model as a basis for defining their inputs and outputs. Each tool description has an *appliesTo* attribute, a set of input aspect paths and a set of output aspect paths. Consider the following.

```
tool momentum_calc {
    appliesTo Body;
    input mass;
    input velocity;
    output momentum;
    ...
}
```

This definition describes a tool that yields the momentum of a body, given its mass and velocity. The ellipsis is to indicate that there may be other attributes for the tool, such as execution information.

The "appliesTo" attribute identifies a dataspace type and specifies that the tool is capable of computing aspect values relative to the associated kind of entity. Thus the "appliesTo" type scopes the tool's inputs and outputs.

The "appliesTo" type also determines the relative semantics of the inputs and outputs. Recall from section 2 the example of two area-computing tools with the same inputs and output, one for rectangles and one for ellipses. In this case, the two tool descriptions would be the same except one would have "appliesTo Rectangle" and the other "appliesTo Ellipse."

4 Plan Generation and Execution

As discussed above, a dataspace type defines a vocabulary of slot names and the semantics of their interdependencies. A tool is specified with respect to some dataspace type and, so, its semantics is determined by the relative semantics of its inputs to outputs.

When a user uses CRADLE, he presents a *problem* which consists of a *problem type*, a set of *givens* and a set of *goals*. The problem type is a dataspace type and each given and goal is an aspect path relative to the problem type. During this process, the CRADLE repository may be used to browse the set of types and their aspects. An example of a problem is as follows.

```
problem {
    problemType Body;
    given velocity;
    given geometryFile;
    given density;
    goal momentum;
}
```

Given a problem, the CRADLE *plan generator* attempts to find a a *plan*, which is a directed acyclic graph of *steps*. Each step contains the name of a tool and the set of *prerequisite* steps that it must follow. The plan also indicates which steps yield one or more of the goals. The plan must be such that each tool's "appliesTo" type is a supertype of (or possibly equal to) the problem type. Each input of each tool must be among the problem's givens or among the outputs of a previous tool. Each of the problem's goals must be among the outputs of some tool.

The full plan generation algorithm is too complex to present here, so we give a quick summary. The algorithm uses backward chaining and works back from the goals. At each point it has a list, *neededAspects*, of aspects (really aspect paths) that need to be computed and a list, *producedAspects*, of aspects that are given or computed by some tool. Iteratively, it selects a needed aspect, *subgoal*, and finds a tool, *tool*, whose “appliesTo” is a supertype of (or equal to) the problem type and whose outputs contain *subgoal*. If there is no such tool, it backtracks if possible. If there is more than one tool, it tries each in turn. To handle the “flatness” problem mentioned in section 2, the algorithm also considers applying a tool to certain subdataspaces of the original problem, in addition to applying it at the root. The outputs of *tool* are added to *producedAspects* and its inputs are added to *neededAspects*. Also a step is allocated and added to the plan. The iteration terminates when all *neededAspects* are in *producedAspects*.

Plan execution is the process by which a plan is executed. It follows the usual rules for executing a DAG. The dataspace model is used to link outputs of one tool to the inputs of the next. As mentioned earlier, the dataspace concept is logical and it is not necessarily the case that any real data structure directly implements the dataspace, although some implementations may. The purpose of the dataspace concept is to provide a conceptual model interrelating types, tool descriptions, problem specification, plan generation and plan executions.

As examples, one implementation may directly implement the dataspace as a centralized hash table from which the tools extract their inputs and into which they place their outputs. Another implementation may instantiate a software component for each step and use the fully qualified input and output names to hook together ports. A third implementation might generate a script using a “mangled” form of the aspect paths to name files or script variables that carry data produced by one step to later ones.

5 Status and Future Work

A CRADLE prototype has been implemented using a client-server model with a protocol similar to web services. The server is in Java and accesses a MySQL database containing tables for tool and dataspace type descriptions. The type descriptions can be used by tool specifiers and by users posing “problems” to CRADLE. Type descriptions are also used during plan generation to reason about inheritance and the types of subdataspaces. Tool descriptions are used during plan generation and may also be used during plan execution to obtain execution-related information.

For future research, we will look at applying a similar methodology to data collections and the tools that operate on them. As RDF[3] is a popular standard for expressing and sharing machine processable information on the web, we will investigate using RDF/-XML[2] in the client-server protocol. Also, we plan to look at various extensions to the dataspace model, including arrays, parameterized aspects (similar to methods) and parameterized types. Adding machine-processable constraint expressions to dataspace types is another potential avenue of investigation.

Acknowledgments: We gratefully acknowledge the contributions of Ken Gee and Karen McCann in the early discussions regarding the direction and design of the project.

References

1. Global Grid Forum, Semantic Grid Working Group: The Semantic Grid Vision. http://www.semanticgrid.org/vision
2. W3C: RDF/XML Syntax Specification (Revised). http://www.w3.org/TR/rdf-syntax-grammar/
3. W3C: Resource Description Framework (RDF): Concepts and Abstract. Syntax. http://www.w3.org/TR/rdf-concepts/
4. W3C: Web Services Architecture. http://www.w3.org/TR/ws-arch/
5. W3C: Web Services Description Language (WSDL). http://www.w3.org/TR/-wsdl20/

OWL-Based Resource Discovery for Inter-Cluster Resource Borrowing

Hideki YOSHIDA, Nobuo SAKIYAMA, Toshibumi SEKI,
Tatsunori KANAI, and Tetsuro KIMURA

Toshiba Corporation, Corporate Research & Development Center,
Komukai-Toshiba-Cho 1, Saiwai-ku, Kawasaki 212-8582, Japan

Abstract. To improve the resource adjustment capability of business application cluster systems, we have devised a new grid system in which clusters borrow lower-level computing resources from each other. Heterogeneous resources that coexist in these clusters are described using OWL Web Ontology Language (OWL) to realize a flexible resource discovery. In this paper we present its system structure and OWL-based resource discovery.

1 Current Business Application Cluster Systems and Their Problem

In contrast to traditional grid systems whose main target is scientific and engineering computation, our research focuses on cluster systems used for business applications such as web application servers, content management systems, or groupware systems.

These applications often show high load fluctuations according to date and time, and at the same time have severe service level requirements such as response time, calling for an alternative approach to conventional grids.

In these systems, jobs are executed as shown in Figure 1.

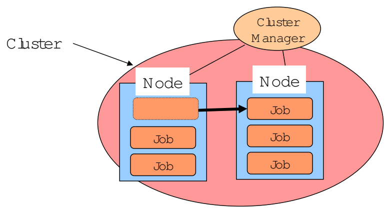


Fig. 1. Job execution in current cluster systems

The Cluster Manager controls the execution of jobs in a cluster and assures failover of jobs in case of a hardware or software failure. The resource assignment in the cluster is managed centrally to reflect the resource allocation policy of that cluster.

If there is an overloaded node in the cluster, jobs are moved from one node to another (as shown with an arrow in Figure 1) to adjust the load.

Such adjustments limited to a single cluster cannot cope with an extremely high load, causing performance degradations such as a delay in response time.

Furthermore, if nodes reserved for provisioning purposes are used up to cope with a fault or an overload, redundancy against further faults or further overloads cannot be secured.

To solve these problems, resource adjustment that spans multiple clusters is demanded. We want to realize this resource adjustment by connecting multiple clusters in a grid-like way.

The traditional approach for connecting cluster systems into a grid is shown in Figure 2. Jobs are either submitted to the grid and allocated to a cluster, or the execution of a job is delegated from one cluster to another cluster.

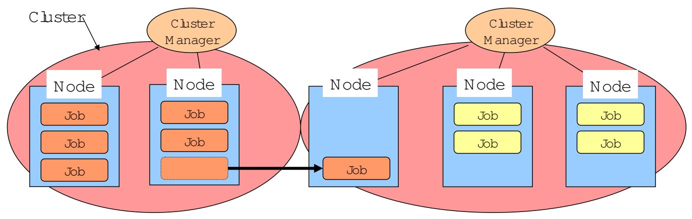


Fig. 2. Conventional method to form a grid by connecting clusters

A problem of this approach is that the execution of jobs cannot easily be assured at the level realized in cluster systems. In a cluster system, the cluster manager typically uses a heartbeat and a quorum mechanism for a reliable fault detection and job failover. Such mechanisms are usually not available to jobs submitted to a grid.

Another problem is the lack of practical mechanisms to reflect a resource allocation policy to another cluster. Setting priorities to jobs or allocating a job to a specific node is not too difficult within a cluster, but in order to apply such allocation policy to jobs submitted to a grid, an elaborate mechanism and a new protocol are needed.

In these respects, this conventional method of grid deployment has a wide discrepancy with the job management in current business application cluster systems.

2 Inter-Cluster Resource Borrowing

As an alternative to the conventional method of grid deployment discussed above, we have devised an approach that we call *inter-cluster resource borrowing* and is more suited to the resource and job management model in current cluster systems. Its overall structure is shown in Figure 3.

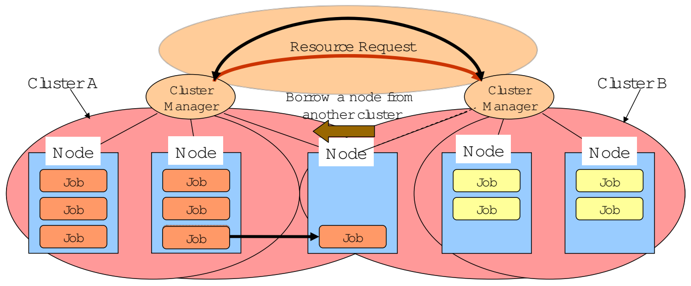


Fig. 3. Abstract structure of inter-cluster resource borrowing

In this approach, each cluster operates as a natural extension of the current cluster. The resource borrowing between clusters is implemented at the dedicated layer that we call *grid layer*.

If Cluster A in Figure 3 is suffering an overload or a fault that cannot be made up for by an adjustment within Cluster A, the cluster manager of Cluster A issues a resource request to the grid layer. The cluster manager of Cluster B is notified of that request in order to transfer some node from Cluster B to Cluster A.

After the node transfer, jobs are moved to the acquired node to accomplish the load adjustment within Cluster A.

The node transfer can be implemented in various ways. *Network boot* is the node transfer mechanism that we think most promising. As an instance of network boot mechanisms, we are investigating

SAN boot. SAN boot is a mechanism to change the correspondence of a node to a boot image on a disk by placing boot images on a disk on a storage area network (SAN). By using the SAN boot mechanism, a node borrowed from another cluster can be booted with the boot image of the borrowing cluster, making it easier to place the node under the control of the borrowing cluster.

In addition to the network boot, virtualization methods such as virtual machines can also be employed to realize node transfer.

In conventional grids, a resource was managed by a fixed cluster and jobs were moving across clusters. In an inter-cluster resource borrowing system, the execution of a job is always control by a fixed cluster, and it is the resources that move across clusters.

A major problem of implementing this system is the discovery of resources to be borrowed. We will discuss it in the next section.

3 Resource Discovery from Heterogeneous Resources

A business application cluster, which is the target of the inter-cluster resource borrowing, has usually multiple application programs running on it. These applications often run on different operating systems and/or on processor architectures. In addition to this internal heterogeneity of a single cluster, clusters are usually constructed with their own goals in mind, so there is a high heterogeneity among clusters.

Furthermore, resources that are physically equivalent can differ in the area or domain in which they can be used. For example, a boot image on a SAN disk can only be used from a node that is connected to the SAN.

In these regards, there is a great deal of diversity among available or required resources.

Inter-cluster resource borrowing calls for a technology to discover adequate resources efficiently from these diversified resources.

In order to perform this discovery, both a description mechanism and a matching mechanism are needed. We will discuss them below.

3.1 Description of Heterogeneous Resources

Resources that are target of this discovery should be described in some way. In addition, a mechanism to specify requirements toward desired resources is necessary.

In OGSA (Open Grid Services Architecture) [1] and Globus Toolkit [3], which is its reference implementation, a resource can be described by a service data. However, requirements can be handled only in a very limited way. For example, GLUE Computing Element (CE) [4] used in Globus can describe hardware resources, but dependency between resources (such as a server and a disk that can be connected to it) cannot be described. Descriptions to handle network topology or bandwidth are also lacking. Resource Specification Language (RSL) [5] has a limited capability to express requirements of an application program, but the RSL description is not currently matched against formal resource descriptions.

It is possible to make a new fixed specification (or a fixed extension of these existing specifications) for the resource description, but in order to describe diversified resources including those yet to come, a more flexible method of resource description is preferred.

3.2 Use of OWL

To describe these wanted and available resources, we have introduced the OWL Web Ontology Language (OWL)[6]. In the sense that we are using a Semantic Web technology for implementing a kind of grid infrastructure, our study can be classified to a Semantic Grid[7].

By using OWL, off-the-shelf inference engines that support OWL can be used for matching of resource descriptions. In addition, a flexible resource description is realized as follows.

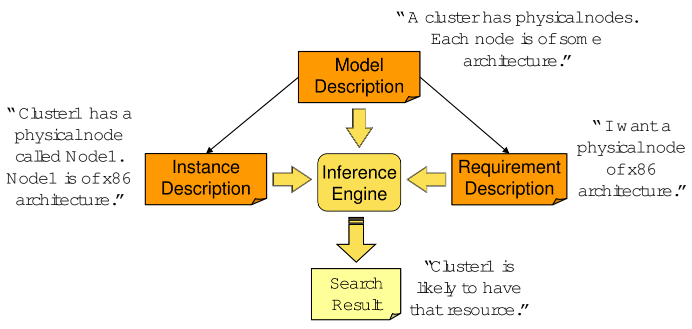

"A cluster has physical nodes. Each node is of some architecture."
Model Description
"Cluster1 has a physical node called Node1. Node1 is of x86 architecture."
Instance Description
Inference Engine
Requirement Description
"I want a physical node of x86 architecture."
Search Result
"Cluster1 is likely to have that resource."

Fig. 4. Matching of OWL descriptions

As shown in Figure 4, three kinds of OWL descriptions are involved:

1. **Model Description.**
   General concepts and their relations are defined here.
2. **Instance Description.**
   Actual resources in given clusters are described here based on the above model.
3. **Requirement Description.**
   Requirements toward resources that a cluster wants to borrow are stated here also based on the same model.

As model itself is also written in OWL and loaded at runtime, a new model to describe additional resources can be incorporated without modifying the source code of the grid layer (though the cluster manager itself should be able to process the new resource descriptions).

Details of the OWL usage are stated in Section 6.

4 Overall System Structure

The overall structure of our implementation is shown in Figure 5.

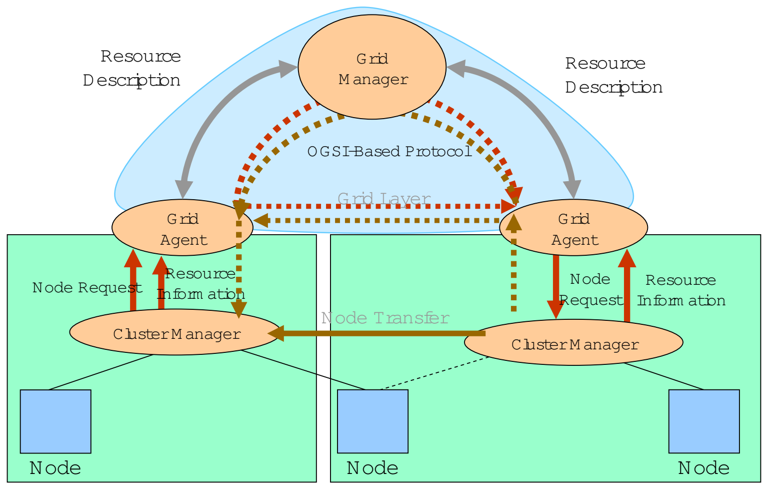


Fig. 5. Overall structure of inter-cluster resource borrowing system

This system is composed of two layers: the cluster layer and the grid layer. They have following responsibilities:

- **Cluster layer.**
  - Execution control of jobs
  - Resource management and allocation within the cluster
- **Grid layer.**
  - Global adjustment of resource allocation
  - Sharing of resource information
  - Processing of resource request
  - Aggregation and resolution of resource requirements
  - Determination of candidate clusters to provide resource
  - Processing of node transfer

Of these, the grid layer is further divided into following two components:

- Grid agent.
  A grid agent resides in each cluster and provides the interface between the grid layer and the cluster.
  In addition to the mutual transfer of resource information, the aggregation of resource information is also performed here.
- Grid manager.
  A grid service to perform the grid-wide processing. Part of the resource information is stored here.

A prototype of this system is currently being implemented as services on Globus Toolkit 3 (GT3).

5 Two-Phase Resource Discovery

The resource information consulted at the resource discovery span from static to dynamic. If we can manage all the information centrally on the grid manager, the resource discovery can be efficiently contained to the grid manager. However, if all the resource information is to be registered frequently to the grid manager, its load may rise excessively and ruin the scalability of the grid. Furthermore, a dynamic information that change without intervention of the grid agent or the cluster manager cannot be registered to the grid manager consistently.

Therefore, we separated the resource discovery into two phases:

Phase 1: Candidate cluster discovery at the grid manager
Phase 2: Resource discovery at the grid agent

Some of the resource information is consulted in both phases, and others are referenced only in the second phase. The criteria of this classification are discussed below.

We assume that resource information can be classified into three categories:

Category 1: Static resource information.
Changes itself rarely and can be registered or cached to the grid manager. The operating system or the processor architecture of a server are examples of information of this category.

Category 2: Dynamic resource information determined by the cluster manager.
The resource information that changes dynamically under the control of the cluster manager. These changes can be notified from the cluster manager to the grid manager to keep the information at the grid manager consistent. The kind and the number of jobs running on a node can be classified here.

Category 3: Dynamic resource information determined by components other than the cluster manager.
The cluster manager does not directly affect changes of this resource information. The network topology information including connectivity, bandwidth or delay is an example of the information of this category.

In this system, the resource information of Category 1 and 2 is transferred from the grid agent to the grid manager and registered there. Phase 1 of the resource discovery depends only on the information of these two categories. In Phase 2, the resource information of Category 3 is measured dynamically, and the information of all three categories is used to determine the resource to be borrowed.

6 Resource Description and Constraint Resolution Using OWL

As stated in Section 3.2, we are using OWL to describe resources and resource requirements. There are three sublanguages of OWL: OWL Lite, OWL DL (description logics), and OWL Full. Of these, OWL

Full allows most flexible descriptions, but there is no guarantee that OWL Full descriptions can be processed in finite time, so it is not feasible here. On the other hand, while OWL Lite descriptions are most simple and can be processed most efficiently, some functionalities not included in OWL Lite (such as `owl:oneOf`) were necessary. Therefore, we have selected OWL DL, whose description capability and computation complexity appears appropriate, for resource descriptions.

Currently we are using Jena 2.1[8], which has a full support of OWL Lite and a partial support of OWL DL, as the inference engine.

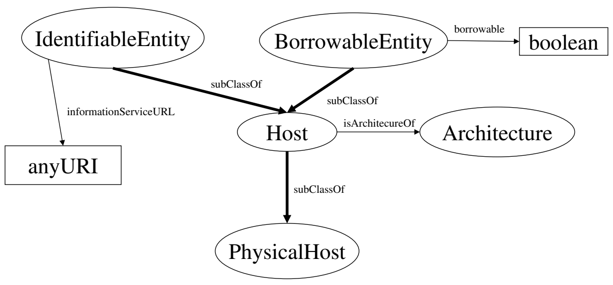


Fig. 6. Partial Example of Model Description

In the current prototype, we are using a handwritten OWL model with many elements derived from GLUE CE. An example model is shown in Appendix A and its partial structure is shown in Figure 6. `PhysicalHost` is a resource that share properties of `Host` including the processor architecture, but has additional properties including connectivity to a network.

Appendix B shows an requirement description based on this model. This description is used in a request to find a cluster which has a physical node that can boot a specific logical node (`lnode1`). The first class in this description defines a cluster which has a desired physical host. Three restrictions to this physical host are described in the definition of the second class: It should be borrowable, should be of the same architecture as `lnode1`, and should share a storage area network with the network storage that hosts the boot image of `lnode1`.

We are planning to develop tools to automatically convert descriptions written in extended GLUE CE or in cluster-native formats to OWL. CIM(Common Information Model)-based standards [9] are also candidates of resource information sources.

7 Implementation Status and Future Directions

A stand-alone prototype (i.e. without connection to a real cluster manager) of the grid layer is already runnable. The resource discovery using the requirement description in OWL and the resource request is realized. The node transfer is currently just simulated.

Following extensions are planned:

- Interface to real cluster managers.
  In order to realize an actual node transfer, at least some functionality of a real cluster manager is needed. We are examining feasibility of several clustering software (including our in-house product cluster manager) to connect to our grid layer.
- Policy handling in resource discovery.
  Besides hardware and software property of resources, policies such as the security or service level policy should be respected in the resource discovery. We are investigating ways to incorporate policy descriptions into the OWL-based resource discovery.
- Aggregation of resource information.
  Registering resource information of individual resources to the grid manager entails a high processing time at the registration and the first phase of resource discovery. It would be desirable if we can make a summarized version of the resource information and register it to the grid manager for a faster resource discovery. Depending on applications, a certain degree of false positives of resource information can be allowed, as the second phase realizes an accurate discovery of resources. If overlooking available resources is acceptable, false negatives are also allowed. [11] mentions to an aggregation function that is related to our idea.

8 Conclusion

In this paper, we have illustrated a problem of current business application cluster systems, and how the inter-cluster resource borrowing can be applied to cope with that problem. By using OWL for the description of resources to be borrowed and using an OWL engine as the matching mechanism, a flexible resource discovery is realized.

We are going to further implement this architecture. Resource borrowing between clusters in a data center is targeted first, and we aim to support borrowing between clusters that span wide-area networks as future prospects.

References

[1] The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Ian Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke, OGSI-WG document, June 2002.

[2] Open Grid Services Infrastructure (OGSI) Version 1.0, S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maguire, T. Sandholm, P. Vanderbilt, and D. Snelling, OGSI-WG document, June 2003.

[3] Globus Toolkit 3 Core — A Grid Service Container Framework, Thomas Sandholm and Jarek Gawor, whitepaper, July 2003.

[4] Sharing a Conceptual Model of Grid Resources and Services, Sergio Andreozzi and Cristina Vistoli, Conference for Computing in High Energy and Nuclear Physics, March 2003.

[5] A Resource Management Architecture for Metacomputing Systems, K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith and S. Tuecke, in 4th Workshop on Job Scheduling Strategies for Parallel Processing, Springer-Verlag, 1998.

[6] OWL Web Ontology Language Reference, Mike Dean and Guus Schreiber, Editors, W3C Recommendation, February 2004.

[7] Semantic Grid and Pervasive Computing, David De Roure, GGF9 Semantic Grid Workshop, October 2003.

[8] Jena: Implementing the Semantic Web Recommendations, Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Dave Reynolds, Andy Seaborne, Kevin Wilkinson, HP Technical Report HPL-2003-146, December 2003.

[9] DMTF Announces New Working Group for Utility Computing, Press Release, Distributed Management Task Force, February 2004.

[10] Interoperability of Grid Resource Descriptions: A Semantic Approach, John Brooke, Kevin Garwood and Carole Goble, GGF9 Semantic Grid Workshop, October 2003.

[11] Knowledge Discovery and Ontology-based services on the Grid, Mario Cannataro, GGF9 Semantic Grid Workshop, October 2003.

A Example OWL Model Description

```
<owl:Class rdf:ID="BorrowableEntity" />
<owl:Class rdf:ID="IdentifiableEntity" />
<owl:Class rdf:ID="BootableEntity" />
<owl:Class rdf:ID="Cluster">
  <rdfs:subClassOf
   rdf:resource="#IdentifiableEntity" />
</owl:Class>
<owl:Class rdf:ID="Host">
  <rdfs:subClassOf
   rdf:resource="#BorrowableEntity" />
  <rdfs:subClassOf
   rdf:resource="#IdentifiableEntity" />
</owl:Class>
<owl:Class rdf:ID="PhysicalHost">
  <rdfs:subClassOf rdf:resource="#Host" />
</owl:Class>
<owl:Class rdf:ID="LogicalHost">
  <rdfs:subClassOf rdf:resource="#Host" />
  <rdfs:subClassOf
   rdf:resource="#BootableEntity" />
</owl:Class>
<owl:Class rdf:ID="Network" />
<owl:Class rdf:ID="StorageNetwork">
  <rdfs:subClassOf rdf:resource="#Network" />
</owl:Class>
<owl:Class rdf:ID="SAN">
  <rdfs:subClassOf
   rdf:resource="#StorageNetwork" />
</owl:Class>
<owl:Class rdf:ID="Storage" />
<owl:Class rdf:ID="NetworkStorage">
  <rdfs:subClassOf rdf:resource="#Storage" />
  <rdfs:subClassOf rdf:resource="#Host" />
</owl:Class>
<owl:Class rdf:ID="StoragePartition" />
<owl:Class rdf:ID="Architecture">
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#i386" />
    <owl:Thing rdf:about="#powerpc" />
  </owl:oneOf>
</owl:Class>
<Architecture rdf:ID="i386" />
<Architecture rdf:ID="powerpc" />
<owl:DatatypeProperty rdf:ID="borrowable">
  <rdfs:domain rdf:resource="#BorrowableEntity" />
  <rdfs:range rdf:resource="&xsd;boolean" />
</owl:DatatypeProperty>
<owl:DatatypeProperty
 rdf:ID="informationServiceURL">
  <rdfs:domain rdf:resource="#IdentifiableEntity" />
  <rdfs:range rdf:resource="&xsd;anyURI" />
</owl:DatatypeProperty>
<owl:ObjectProperty rdf:ID="controls">
  <rdfs:domain rdf:resource="#Cluster" />
  <rdfs:range rdf:resource="#Host" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isOfArchitecture">
  <rdfs:domain rdf:resource="#Host" />
  <rdfs:range rdf:resource="#Architecture" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isArchitectureOf">
  <owl:inverseOf rdf:resource="#isOfArchitecture" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="connects">
  <rdfs:domain rdf:resource="#Network" />
  <rdfs:range rdf:resource="#Host" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isConnectedTo">
  <owl:inverseOf rdf:resource="#connects" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="connectsPhysicalHost">
  <rdfs:subPropertyOf rdf:resource="#connects" />
  <rdfs:domain rdf:resource="#Network" />
  <rdfs:range rdf:resource="#PhysicalHost" />
</owl:ObjectProperty>
<owl:ObjectProperty
 rdf:ID="isConnectedPhysicalHostTo">
  <owl:inverseOf
   rdf:resource="#connectsPhysicalHost" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="connectsNetworkStorage">
  <rdfs:domain rdf:resource="#Network" />
  <rdfs:range rdf:resource="#NetworkStorage" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="includes">
  <rdfs:domain rdf:resource="#Storage" />
  <rdfs:range rdf:resource="#StoragePartition" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hosts">
  <rdfs:domain rdf:resource="#StoragePartition" />
  <rdfs:range rdf:resource="#LogicalHost" />
</owl:ObjectProperty>
```

B Example OWL Requirement Description

```
<owl:Class rdf:ID="ClustersIWantNow">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="&cst;Cluster" />
    <owl:Restriction>
      <owl:onProperty rdf:resource="&cst;controls" />
      <owl:someValuesFrom
       rdf:resource="#PhysicalHostsIWantNow" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
<owl:Class rdf:ID="PhysicalHostsIWantNow">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="&cst;PhysicalHost" />
    <owl:Restriction>
      <owl:onProperty
       rdf:resource="&cst;borrowable" />
      <owl:hasValue rdf:datatype="&xsd;boolean">
        true
      </owl:hasValue>
    </owl:Restriction>
    <owl:Restriction>
      <owl:onProperty
       rdf:resource="&cst;isOfArchitecture" />
      <owl:someValuesFrom>
        <owl:Restriction>
          <owl:onProperty
           rdf:resource="&cst;isArchitectureOf" />
          <owl:hasValue rdf:resource="#lnode1" />
        </owl:Restriction>
      </owl:someValuesFrom>
    </owl:Restriction>
    <owl:Restriction>
      <owl:onProperty rdf:resource=
       "&cst;isConnectedPhysicalHostTo" />
      <owl:someValuesFrom>
        <owl:Restriction>
          <owl:onProperty rdf:resource=
           "&cst;connectsNetworkStorage" />
          <owl:someValuesFrom>
            <owl:Restriction>
              <owl:onProperty
               rdf:resource="&cst;includes" />
              <owl:someValuesFrom>
                <owl:Restriction>
                  <owl:onProperty
                   rdf:resource="&cst;hosts" />
                  <owl:hasValue
                   rdf:resource="#lnode1" />
                </owl:Restriction>
              </owl:someValuesFrom>
            </owl:Restriction>
          </owl:someValuesFrom>
        </owl:Restriction>
      </owl:someValuesFrom>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```