



**Annotation and Assessment Work Package 2
(AA2)**

Secure Annotation Services

Final Report

Version 1.0, MARCH 2007

Lead Investigator: Imran Khan – imrank@itee.uq.edu.au
Chief Investigator: Prof Jane Hunter – jane@itee.uq.edu.au
Prepared by: Imran Khan – imrank@itee.uq.edu.au

Executive Summary

This is a report for the AA2 – “development of secure authenticated access to annotation servers by trusted members through the development of Shibboleth-based interface to the W3C’s open source Annotea server”. The work package is part of the DART (Dataset Acquisition, Accessibility and Annotation e-Research Technologies) project.

To implement the objective of AA2, AA2 is has been separated into two areas of research:

- Development of a modified annotation server which enables authentication and definition of access constraints over annotations using Shibboleth and XACML
- Tools for the creation of access constrained annotations as well as tools for the searching, browsing and retrieval of secured annotations.

Initial development focused on the implementation of a secured annotation server. Its development was based upon the open Annotea protocol defined by the W3C. Along with this initial development, we carried out the deployment of the Shibboleth infrastructure to support authentication over the server. Once these tasks were complete, we proceeded to incorporate Sun’s XACML implementation to provide access control.

The second area of development centred upon the development of client-side tools to interact with the annotation server. For this component of the project we implemented a plug-in for the Internet Explorer sidebar which provided all of the necessary functionality for the creation, browsing and search over annotations deposited on the annotation server.

This report covers:

- A brief description of the background, available research, the motivation behind this work package and how it relates to other DART packages,
- The milestones delivered in order to achieve the aims of this work package including discussions of any issues discovered, and
- A description of the architecture of the current system delivered and the interaction with external systems.

Table of Contents

1	Introduction	4
2	Project Milestones	5
3	Project Outcomes.....	6
3.1	Q1 March 2006 - Literature and Technology Review of Annotea, Shibboleth and XACML.6	
3.1.1	Summary of Work.....	6
3.1.2	Issues Encountered.....	7
3.2	Q2 June 2006 – Development of Annotea Annotation Server	8
3.2.1	Summary of Work.....	8
3.2.2	Issues Encountered.....	9
3.3	Q3 September 2006 – Deployment and Integration of Shibboleth Infrastructure.....	9
3.3.1	Summary of Work.....	9
3.3.2	Issues Encountered.....	10
3.4	Q4 December 2006 – Development of End-user Software	10
3.4.1	Summary of Work.....	10
3.4.2	Issues Encountered.....	11
3.5	Q1 February 2007 – Testing and Evaluation of Software.....	11
3.5.1	Summary of Work.....	11
3.5.2	Issues Encountered.....	13
3.6	Technical Requirements.....	13
3.6.1	Annotation Server	13
3.6.2	Annotea Sidebar.....	13
4	Archival Storage of Project Deliverables	15
5	Recommendations	16
6	Publications	17
7	Terms of Reference	18
7.1	Glossary	18
7.2	References.....	18
8	Report Signoff.....	19
9	Appendix A: Publication: Implementing a Secure Annotation Service.....	20

1 Introduction

The purpose of this document is to outline the objectives of the work package AA2, the work that was performed and the outcomes and conclusions from the work.

The aim of the AA2 is to develop a secure annotation server that can be used by end-users to create, browse, search and share annotations in secure manner. It is also important the tools be provided for end-users and administrators to easily define access control over annotations which they create, using a simple and intuitive user interface.

The objectives of the work package are to:

- Improve annotation and deposit rates,
- By allowing end-user control over
- Who can annotate what and who can access the annotations.

2 Project Milestones

The initial milestones for AA2 included:

- Prototype software
- Tested and Working software

Prototype software included the secure annotation server, developed as Tomcat web archive and also an Internet Explorer sidebar plugin which includes all of the necessary functionality for the end-user to interact with the server. It should be noted that this sidebar has also been successfully integrated into the tools developed as part of Work Package AA3.

The development of the work packages has been developed largely by Imran Khan, who has collaborated with Jonathon Guerin and Ronald Schroeter from Work Package AA3. Initial testing of the prototype software was done in conjunction with Ron Chernich.

3 Project Outcomes

The outcomes of the project include:

- Secure annotation servers
- Secure, access controlled search, browse and retrieval tools for annotations.

The project comprises of 5 milestones:

1. Literature and Technology Review of Annotea, Shibboleth and XACML,
2. Development of Annotea Annotation Server,
3. Deployment and Integration of Shibboleth Infrastructure,
4. Development of End-user Software, and
5. Testing and Evaluation of Software.

3.1 Q1 March 2006 - Literature and Technology Review of Annotea, Shibboleth and XACML

3.1.1 Summary of Work

The initial milestone consisted of reviewing the three core technologies; Annotea, Shibboleth and XACML.

Initial research focused upon the W3Cs Annotea protocol. The protocol itself was relatively simple, however had limited existing support in the form of 2 different software packages; Zope [1] and W3C Perllib Annotea server [2]. Although it may have been possible to build upon either of these two existing implementations, we identified a number of shortcomings with this approach:

- lack of thorough documentation
- developer's unfamiliarity with programming languages
- unsuitable language API for interfacing with other key technologies
- lack of support for integrating alternative RDF storage
- lack of support for integrating inferencing engine or reasoner
- lack of SPARQL support

After identifying the shortcomings of the existing Annotea compliant servers, in particular a lack of a programming API in either Perl or Python to the XACML API, we deemed it necessary to develop the annotation server from scratch using the Java programming language.

We also conducted some initial testing of Annotea compliant client-side software. We identified 2 working examples; Annozilla [3] a Mozilla browser plugin and Amaya [4] W3C's browser which included Annotea support. Amaya was thoroughly documented and included most of the features outlined in the Annotea protocol. Amaya's lack of adoption among general Web users was considered a key issue. Annozilla was much less mature and included only a fraction of features mentioned in Annotea.

The key feature missing with the Annotea protocol and the key reason for the work carrying out this project is the lack of any access control or authentication. This is the reason for integrating the other key technologies. Shibboleth provides a method in which to authenticate users, while XACML is used to define access control over resources (annotations).

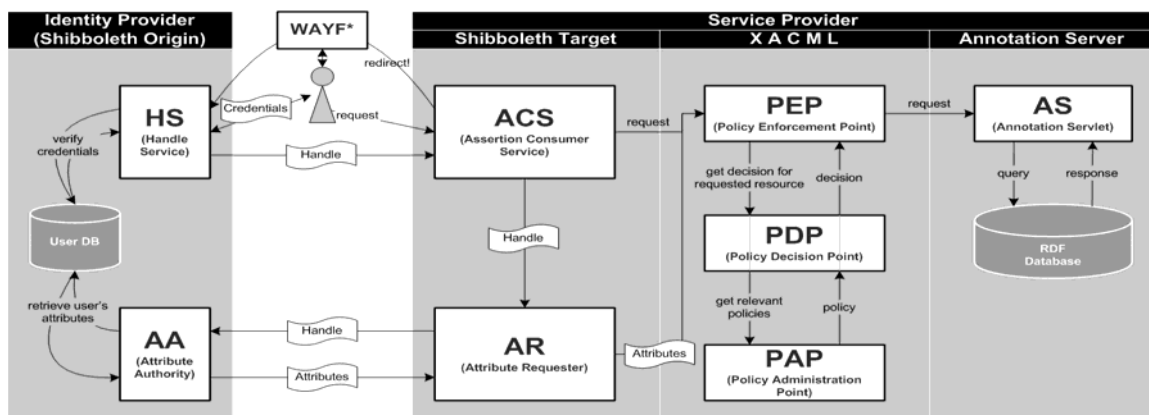


Figure 1: Annotation Server System Design

The result of this initial milestone resulted in the production of the following artefacts:

- modified Annotea protocol and schema which incorporates XACML policies
- design of the annotation server's security infrastructure which encompasses Shibboleth architecture
- identification of technologies to use in the development of the annotation server
- identification of potential client-side applications

3.1.2 Issues Encountered

This initial stage of the project did not provide any substantial issues apart from the early steep learning curve.

3.2 Q2 June 2006 – Development of Annotea Annotation Server

3.2.1 Summary of Work

Based upon the design carried out in the previous quarter, it became possible to begin development of the annotation server. This initial phase of development focused upon a non-secure implementation of the annotation server which was able to communicate with an Annotea compliant client.

In the ongoing design and development of the server, we identified a number of design decisions as well as possible extensions:

- it should be possible to easily incorporate other RDF data stores as the need might arise
- it should be possible to easily switch between secure/non-secure mode
- it should be possible to easily search over the annotations using SPARQL, instead of Algae language
- it should be possible to conduct standard text-based search over the annotations
- it should be possible to subscribe to a subset of annotations using RSS
- it should be possible to upload local files as annotations
- it should be possible to incorporate controlled vocabularies from secondary sources, to perform semantic searches across annotations

After completing non-secure annotation server, we were able to test the server using both the Amaya and Annozilla clients. Using this approach we were able to conduct a limited set of testing over the annotation server for the standard Annotea protocol.

<p>Example Policy</p> <table border="1"><tr><td>Staff (eduPersonAffiliation = "staff")</td></tr><tr><td>LIST = "Permit"</td></tr><tr><td>READ = "Permit"</td></tr><tr><td>READ_POLICY = "Deny"</td></tr><tr><td>Student (eduPersonAffiliation = "student")</td></tr><tr><td>LIST = "Deny"</td></tr><tr><td>READ = "Deny"</td></tr><tr><td>READ_POLICY = "Deny"</td></tr></table>	Staff (eduPersonAffiliation = "staff")	LIST = "Permit"	READ = "Permit"	READ_POLICY = "Deny"	Student (eduPersonAffiliation = "student")	LIST = "Deny"	READ = "Deny"	READ_POLICY = "Deny"	<p>Subject</p> <p>eduPersonAffiliation = "student"</p> <p>Resource</p> <p>Resource-id = "https://localhost/Annotea/AnnoteaServlet/policy/123"</p> <p>Action</p> <p>Action-id = "READ"</p>
Staff (eduPersonAffiliation = "staff")									
LIST = "Permit"									
READ = "Permit"									
READ_POLICY = "Deny"									
Student (eduPersonAffiliation = "student")									
LIST = "Deny"									
READ = "Deny"									
READ_POLICY = "Deny"									

Figure 2: (a) Example Policy and (b) Example Request

Artefacts produced as a result of this milestone include the following:

- a non-secure Annotea compliant annotation server
- a template for simplified XACML policies to be used for defining access constraints

- identification of other possible extensions to the server

3.2.2 Issues Encountered

Issues encountered during this stage were mainly due to two problems:

- there is a limited number of XACML editing tools available to develop and test XACML policy definitions

3.3 Q3 September 2006 – Deployment and Integration of Shibboleth Infrastructure

3.3.1 Summary of Work

After completing the development of the non-secure annotation server, we began extending the server to incorporate Shibboleth and XACML. This milestone included the deployment of Shibboleth and the integration of Sun's XACML engine.

Shibboleth is a federated approach to authentication, centred primarily in the educational realm. It would therefore obviously run distributed across multiple organizational domains and part of the process involves registering to become part of a federation. However initially for testing purposes and to minimize the learning curve, Shibboleth infrastructure was deployed locally.

This task was a lengthy and complicated task as a result of the developer's unfamiliarity with a number of the underlying packages and software which Shibboleth relies upon. However the task of Shibboleth deployment was made easier after Imran Khan, attended the MAMS workshop. This workshop provided a forum to discuss a number of ongoing issues and also helped in joining the Level 1 MAMS federation.

In parallel with the Shibboleth deployment we were able to continue the development of the annotation server. During this process we were able to mock some of the expected inputs from Shibboleth and use this as a way to test the server. Due to the modular design of the server we were also able to easily integrate the XACML engine to perform authorization decisions.

Although not part of this milestone, it became necessary to have some client-side tools available to test the underlying server. So during this milestone there was some preliminary development of client-side tools to supplement the server-side modules.

The culmination of this milestone produced the following artefacts:

- a working deployment of Shibboleth infrastructure
- extended annotation server incorporating XACML access control and Shibboleth
- extensions to the Annotea protocol to incorporate the creation and querying of XACML policies

3.3.2 Issues Encountered

The issues encountered:

- the length of time taken to deploy Shibboleth delayed the development
- lack of a standard format to transfer eduPerson attributes, meant that we simply adopted

3.4 Q4 December 2006 – Development of End-user Software

3.4.1 Summary of Work

With the culmination of server-side development, focus moved towards creating client-side tools which would enable users to manage control over annotations. As already discussed after analysing the existing Annotea client tools, we came to the agreement that it would be better to develop our client side tools from scratch.

One of the key reason's for developing the client-side tool as an Active-X component was so that it could be used within the AA3 Work Package. The working codebase for AA3 was developed using the .NET Framework. Another advantage in developing the client-side tool as an Active-X component is that it became possible to use it as a plug-in for Internet Explorer. This in turn made it possible to use the Internet Explorer plug-in to be used by Work Package AA1.

Development of the client-side software was undertaken in two stages:

- Browsing, creation and searching annotations
- Sharing and management of annotations

3.4.1.1 Browsing, creation and searching annotations

In the first stage of development we analysed existing client-side tools as a basis for our preliminary user interface design. We were able to quickly and easily develop mock user interfaces similar to Annozilla. This interface was then modified according to what we deemed to be deficiencies in the Annozilla user interface design.

In particular, changes were made to the:

- creation of annotations
- searching of annotations
- display of annotations

We then created a set of libraries to support the HTTP based Annotea protocol. Using the annotation server developed in the previous quarter we were then able to progressively incorporate and test each of the core functionalities specified in Annotea:

- Posting new annotations
- Replying to existing annotations

- Updating existing annotations
- Querying using a variety of queries both defined within and outside of Annotea

During this stage we were also able to identify some inconsistent behaviour within the server. Therefore there was parallel debugging of the server as bugs became apparent during client side development.

3.4.1.2 Sharing and management of annotations

This part of the milestone is related to creating end-user tools for the creation and management of policies. One of the key requirements of this Work Package was to enable end users to manage access to their annotations easily. It was therefore necessary to develop a user interface which abstracts the complexity of the XACML language as far as possible.

We used the template XACML policy developed specifically for our requirements as a guide for the user interface for the tool. Once the UI an appropriate UI was agreed upon we went about implement the extended Annotea protocol for the creation, updating and querying of policies.

The final stage of this development centred on generating a of list test-case scenarios to evaluate the correct operation of the annotation server's authorization logic. This included testing the retrieval of annotations with a predefined policy as well as testing the retrieval after the policy was updated.

This milestone culminated in the production of the following artefacts:

- user interface for the creation, browsing and querying of annotations
- user interface for the creation, updating and browsing of annotation policies and their association with annotations

3.4.2 Issues Encountered

- inability to display multiple contexts
- no suitable RDF parser for C# could be found
- not platform independent
- policy editor created, is only specific to the annotation server and cannot be used to make more generic queries

3.5 Q1 February 2007 – Testing and Evaluation of Software

3.5.1 Summary of Work

This milestone focused upon the testing and evaluation of the server and sidebar. There were some additional features and refactoring as well. We will discuss the testing and evaluation in each in turn.

3.5.1.1 Annotation Server Testing and Evaluation

During this milestone, with the assistance of Ron Chernich we were able to refactor much of the existing codebase. This refactoring over a period of 4 weeks resulted in a number of design and architectural changes. This included storing uploaded files within the underlying database as well as providing support for “Association Annotations” in collaboration with AA3 [5].

The other major development during this milestone was the incorporation of OAI-PMH interface to the annotation server. This provides an interface which OAI enabled harvesters can use to harvest annotation information contained within annotation servers. This interface was created by using the libraries provided developed by OAICat.

This milestone also included creating unit and application tests for the Annotation server. Using this approach we were able to test most of the core functionality outlined in Annotea protocol as well as the additional features that have been added.

3.5.1.2 Annotation Sidebar Testing and Evaluation

One of the major shortcomings of this Work Package is the lack of testing for the Annotation sidebar. This has resulted in sidebar which is not completely stable, which has in turn made evaluation of the software outside of the group difficult.

Despite this we have been able to evaluate the use of the Annotea sidebar in 3 major use cases:

- Annotation of Web Pages:
 - University of Queensland’s ePrints archives
- Annotation of Video (AA3):
 - Motion picture content
 - Medical content
 - AIS Swimming content
 - Ethnographic content
- Annotation of Crystallographic data (AA1):
 - 3-D Crystallographic models

In addition to the evaluations outlined above there were some additional simplifications to the UI during this milestone. Also modifications needed to be made to incorporate “Association Annotations”.

This milestone resulted in the creation of the following artefacts:

- installer for the Annotea sidebar
- install script for Annotation server
- test scripts for the Annotation server
- initial user evaluation results of Annotation sidebar

- back up scripts for the Annotation server data

3.5.2 Issues Encountered

- scalability testing not very impressive
- some initial feedback would indicate that the user interface not simple enough
- there was not adequate time or resources to conduct thorough unit testing of the sidebar

3.6 Technical Requirements

The system comprises of server and client sides. Each of the technical requirements for both sides is outlined below.

3.6.1 Annotation Server

The annotation server is a Tomcat Web application. It has been developed using the following environment configurations:

- Windows XP SP 2 and Fedora Core 5.2
- Java Runtime Environment (1.5.06)
- Java Software Development Kit (1.5.06)
- Apache Tomcat 5.0 or 5.5
- MySQL 4.1

In addition to the specified system configuration, the annotation server also needs to be protected by the Shibboleth Service Provider. The deployment of the Shibboleth infrastructure is complicated process and is outside the scope of this document.

It makes use of the following third party libraries:

- Jena 2.3
- Sun XACML 1.0
- Apache Logging Library
- MySQL J Connector
- Apache Lucene 1.1
- COS 1.0
- OAICat 1.5.51

3.6.2 Annotea Sidebar

The Annotea Sidebar is a plug-in for the Internet Explorer browser. It has been tested and developed in the following environment:

- Windows XP SP 2
- Microsoft .NET Framework v2.0.50727
- Internet Explorer 6.0 and 7.0

It makes use of the following third party libraries:

- DevComponents DotNetBar 6.4.0.1

4 **Archival Storage of Project Deliverables**

The software and installation instruction for the AA2 work package is available from the eResearch AA2 home page:

<http://www.itee.uq.edu.au/~eresearch/projects/dart/outcomes/annotationserver.php>

This links to the download page for the work package which is at:

<http://www.itee.uq.edu.au/~eresearch/projects/dart/outcomes/serverDownload.php>

It is also stored on DVD for archival purposes.

5 Recommendations

The objective of this Work Package according to the original DART bid document is to “improve annotation and deposit rates by allowing end-user control over who can annotate what and who can access the annotations” [6]. The artefacts created from this Work Package were:

- secure annotation server using Shibboleth authentication and XACML authorization
- secure access controlled client tool for the search, browsing and retrieval of annotations

These two key artefacts provide the necessary tools for researchers to quickly and easily annotate each others work with the knowledge that the valuable thoughts and ideas contained within the annotations are secured

However it is important to note that the software produced from this Work Package are purely prototypes and require thorough revisions to make them suitable for enterprise use. Some major issues which have yet to be resolved include:

- dealing with scalability of RDF datastore
- inadequate testing of client-side software
- non-platform independent client-side software (Internet Explorer specific)
- lack of aggregate functions in the SPARQL language prevents queries such as MAX, AVG and COUNT
- possible scalability problems with Sun’s XACML API

The following are recommendations of possible improvements and enhancements to this Work Package:

- support for distributed annotation server queries
- more research into rich inferencing and semantic searches over annotations
- replace underlying RDF datastore with a more scalable datastore such as Kowari
- ability to define fine-grained policies on a triple level
- a cross platform implementation of the Annotea sidebar
- better administrative tools for managing annotation server
- “Association Annotations” within browser to allow associations across browser tabs
- Consider future impact of OpenId [7] on Shibboleth and provision of support for OpenId identity management [8]

6 Publications

The work carried out in this Work Package has resulted in the production of the following Conference paper:

- I. Khan, R. Schroeter and J. Hunter. ["Implementing a Secure Annotation Service"](#), International Provenance and Annotation Workshop, Chicago, USA. 3 - 5 May 2006. pp 212-221.
- R. Schroeter, J. Hunter, J. Guerin, I. Khan and M. Henderson. ["A Synchronous Multimedia Annotation System for Secure Collaboratories"](#) 2nd IEEE International Conference on E-Science and Grid Computing (eScience 2006). Amsterdam, Netherlands. December 2006. p 41.

7 Terms of Reference

7.1 Glossary

Acronym	Definition
Shibboleth	Internet2 Middleware layer for Single-SignOn
HTML	HyperText Markup Language
XACML	eXtensible Access Control Markup Language
OWL	Web Ontology Language
RDF	Resource Description Framework
SPARQL	SPARQL Protocol And RDF Query Language
SAML	Security Assertion Markup Language
OAI	Open Archive Initiative
OAI-PMH	OAI Protocol for Metadata Harvesting

7.2 References

- [1] Zope Annotation Server, Retrieved on 20 Feb 2005, from <http://www.zope.org/Members/Crouton/ZAnnot/>
- [2] PerlLib Annotations Server HOWTO, Retrieved on 19 December 2003, from <http://www.w3.org/1999/02/26-modules/User/Annotations-HOWTO>
- [3] M. Wilson, Annozilla (Annotea on Mozilla), Retrieved on 7 Aug 2005, from <http://annozilla.mozdev.org/>
- [4] Amaya Home Page, Retrieved on 5 Feb 2005, from <http://www.w3.org/Amaya/>
- [5] R. Schroeter and J. Hunter, "Annotating Relationships between Multiple Mixed-media Digital Objects by Extending Annotea," presented at European Semantic Web Conference (ESWC), Innsbruck, 2007.
- [6] A. Treloar, The Dataset Acquisition, Accessibility, and Annotation e-Research Technologies Project (DART), Retrieved on from http://dart.edu.au/DART_Bid_Document.pdf
- [7] OpenID: an actually distributed identity system, Retrieved on 1 April 2007, from <http://openid.net/>
- [8] A. Powell and D. Recordon, "OpenID: Decentralised Single Sign-on for the Web," *Ariadne*, 2007. Available at <http://www.ariadne.ac.uk/issue51/powell-recordon/>

8 Report Signoff

It is agreed between

Imran Khan

and

Prof Jane Hunter


and

Andrew Treloar

That the **Final Report Document** for the AA2 – Secure Annotation Tools gives a full account of the work undertaken for the DART Project.

- has been read and reviewed by all parties,
- shows that the work package AA2 has been completed satisfactorily,
- clearly outlines the functionality that was delivered.

Dated this 1st day of May 2007


Signed by Prof Jane Hunter for
and on behalf of the Chief
Investigator

Signed for and on behalf of DART by
the Project Director Andrew Treloar

9 Appendix A: Publication: Implementing a Secure Annotation Service

Paper begins on the next page.

Implementing a Secure Annotation Service

Imran Khan, Ronald Schroeter, Jane Hunter

The School of ITEE
The University of Queensland,
St Lucia, Queensland, Australia
{imrank,ronalds,jane}@itee.uq.edu.au

Abstract: Annotation systems enable “value-adding” to digital resources by the attachment of additional data in the form of comments, explanations, references, reviews and other types of external, subjective remarks. They facilitate group discourse and capture collective intelligence by enabling communities to attach and share their views on particular data and documents accessible over the Web. Annotation systems vary greatly with regard to the types of content they can annotate, the extent of collaboration and sharing they allow and the communities which they serve. However many applications share the need to authenticate the source of annotations and restrict access to them - in order to protect intellectual property rights or personal privacy. This paper describes a secure, open source annotation system that we have developed that uses *Shibboleth* [1] and *XACML* [2] to identify and authenticate users and restrict access to annotations stored on an *Annotea* [3] server.

1 Introduction

Annotations have long been used as a tool to facilitate collaborative scholarly discourse. They enable users to attach additional material such as comments, notes, queries, assessments, references to resources such as documents, images or datasets. When applied to digital resources shared via the Web, they provide a very powerful collaborative tool - enabling the easy capture and wide dissemination of individuals’ and group opinions of particular digital resources.

Currently available annotation systems vary widely with respect to the types of content they annotate, the extent of collaboration and sharing they allow and the communities which they serve [4]. Although they have been successfully applied to domains including education [5], research, medicine [6] and neuroscience [7] in order to capture and exchange metadata, ideas, opinions and interpretations, evaluation of these applications indicates limitations in existing commercial and prototype systems. Current systems are limited by: lack of responsiveness, use of non-standard proprietary technologies; lack of authentication of the annotations’ creator; limited search capabilities; lack of security mechanisms; inability to reply to/stagger annotations; asynchronous sharing only; support for limited media types; coarse granularity and unstructured annotations (single field, free text only).

The main focus of the work described in this paper is to provide annotation tools for collaborators within eResearch environments – and particularly higher education environments. A critical requirement for such a domain is the need to be able to restrict access to annotations attached to a particular collection of digital resources - to a particular group of trusted colleagues - for reasons of privacy, confidentiality or protection of intellectual property. This is particularly important within eScience, where the annotation or interpretation of the raw document or data, is often more valuable than the target of the annotation. Also by providing researchers with a robust, reliable security infrastructure, they may be more willing to engage in the exchange of views and ideas – a key to successful inter-organizational collaboration.

The security requirements for annotations involve two levels of protection:

- protecting the annotation server on which the annotations are stored, through some form of identity management and authentication;
- authenticating and protecting the individual annotations through the specification of access policies that define permissible types of access (e.g. list, create, read, edit, delete) by individual users or user groups based on user attributes.

Within this paper we describe an open source implementation of a secure annotation service that we have developed. Our implementation involves the combination and extension of a number of existing open source technologies that are based on open standards:

- Annotea – a Web-based annotation server developed by the W3C as part of the Semantic Web initiative [8];
- Shibboleth – an Internet2 middleware initiative that enables identity management and secure access to Web resources shared amongst multiple organizations [1];
- XACML (eXtensible Access Control Markup Language) – XML-based language for defining and enforcing access control policies [9].

The remainder of this paper describes in detail the secure annotation system that we have built. Section 2 describes previous related activities in the development of annotation systems and security mechanisms. Section 3 describes the overall architecture of our system and its main components. Section 4 illustrates the user interface and system functionality. Section 5 provides an evaluation of the system and describes future work. Section 6 provides a brief conclusion.

2 Background and Previous Work

Significant prior work has been carried out on both web-based annotation systems and on identity management and role-based access control. Rather than re-invent the wheel, we carried out an analysis of existing systems to determine if any currently available solutions satisfied our needs and hence could be integrated, refined or extended.

2.1 Existing Annotation systems and Annotea

A survey of current Web-based annotation systems [4] reveals that they vary in the way in which annotations may be attached, the way in which they are presented and in the access control mechanisms. Some systems are designed for private use only whilst others permit sharing amongst groups and/or public access. None of the surveyed systems provide the kinds of fine-grained access control mechanisms that is required by collaborative teams of scientists engaging in eResearch.

Through earlier work [10] we identified Annotea [8] as an ideal approach for implementing an annotation server. Annotea is a Web-based annotation system that uses Resource Description Framework (RDF) [11] to model annotations as a set of statements or assertions made by the author. These annotations are then stored in a HTTP enabled server, which enables clients such as Annotilla [12] and Amaya [13] to query, update, post, delete and reply to annotations. Currently there are two publicly available implementations of annotation servers which use Annotea: Zope [10] and W3C PerlLib[14]. Figure 1 illustrates the RDF annotation schema used to describe various

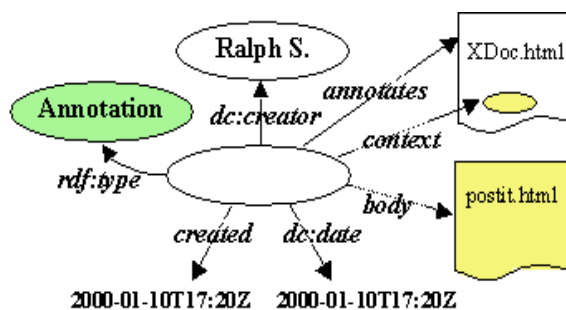


Figure 1: Annotea's Annotation Schema [5]

properties of an annotation including its author, title, date of creation, body and context. A key strength of the Annotea protocol is that it uses open W3C standards such as RDF, XPointer, XLink and HTTP. By choosing to use RDF, Annotea makes it possible to easily adapt or extend the existing scheme to incorporate additional information (e.g. what type of annotation it is, its language, the type of resource it annotates, structured annotations). The use of machine-processable RDF descriptions also enables easier search, retrieval and

linking of the annotations to related resources and services using semantic web technologies (e.g., OWL, SWRL). Annotea can also be easily extended to allow for the annotation of media types other than text e.g., images through the use of SVG [15]. Vannotea [16] uses a similar approach to extend Annotea to enable the annotation of videos. These applications and in particular the application of Vannotea to Indigenous Knowledge [17] clearly identified the need to further extend the Annotea server to enable fine-grained access control to annotations.

2.2 Identity Management and Shibboleth

Harris et al. generated a comprehensive report describing access management (AM) systems used in the UK Higher Education sector [18]. The most prominent systems identified included: Microsoft's Passport, Liberty Alliance, WS-Security, PAPI, Athens and Shibboleth. Of the six systems, only three are targeted at the higher education domain, while the other three (Passport, Liberty Alliance and WS-Security) are primarily focused on business-centric solutions. At present in the UK, the major AM solution in higher education is Athens. Its key distinction is that it uses a single centralized database which maintains a list of Athens username/passwords for all users with accounts at participating institutions. Although Athens uses distributed administration and physical database replication, it does not exhibit true *Single Sign-on* capability. This limitation of Athens led us to consider Shibboleth. Morgan et al [19] describe Shibboleth as "an open-source system that extends Web-based applications and identity management for secure access to resources among multiple organizations." Shibboleth is based upon a number of open standards including HTTP, XML, XML Schema, XML Signature, SOAP (Simple Object Access Protocol). In particular it is dependent on:

- SAML [20] Security Assertion Markup Language for the exchange of assertions between the Identity Providers and Service Providers
- eduPerson [21] – an EDUCAUSE initiative to define a standard set of person attributes in higher education environments

2.3 XACML

As Lorch et al [9] explains, Shibboleth does not provide a dynamic and distributed approach to access control. XACML enables us to address these issues. XACML (eXtensible Access Control Markup language) is an XML-based language used to describe general purpose access control policies as well as an access control decision request/response language [20]. XACML policies are expressed in XML and must conform to an XML schema that defines the language semantics. Policies are defined in a tree-like structure as a set of rules pertaining to a particular resource and subject. XACML also specifies the structure and syntax of the requests and the responses to these requests. Each request is composed of attributes associated with the requesting user, the resource being acted upon, the action being performed and environmental information. The response can be one of four specific types: Permit, Deny, Not Applicable, and Intermediate.

3 System Architecture and Implementation

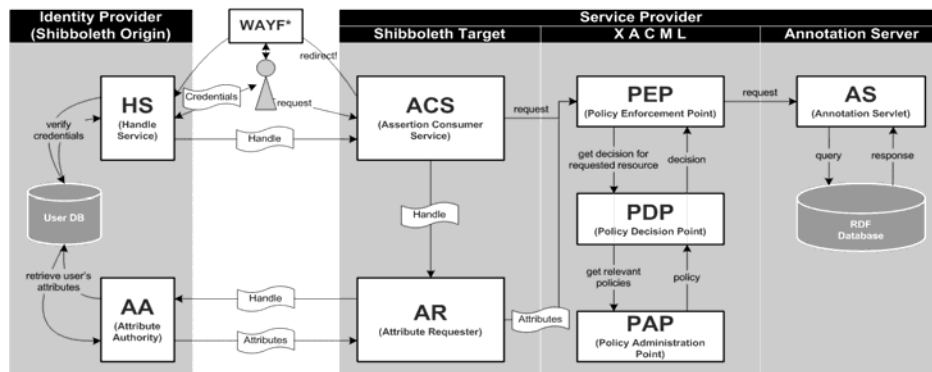


Figure 2: System Architecture

Figure 2 illustrates the overall architecture of the system. The diagram highlights the two key components of the Shibboleth architecture: Identity Provider (IDP) and Service Provider (SP).

The annotation server within Figure 2 is part of the Shibboleth SP, and may be located on any of the organizations/universities that are part of the federation. Figure 2 illustrates a simplified view of a shibbolized annotea transaction. Firstly Shibboleth is responsible for authenticating a user and retrieving the users' attributes from the requestors IDP. These attributes are then passed on to the XACML module which generates an XACML request and retrieves the relevant policy from the repository to generate an XACML response. Finally this response is passed to the annotation server which based on the XACML response either proceeds with or denies the request.

3.1 Server-side

The Server side consists of four main architectural components: the Annotation and Policy server, XACML module; Shibboleth attributes and the Jena database.

3.1.1 The Annotation and Policy Server

The Annotea server (implemented using Java Servlets, hosted on a Tomcat server) has been extended to support the fine-grained access policies in addition to the operations defined by Annotea (posting, querying, downloading, updating, replying and deletion of annotations). Figure 3 illustrates the extensions made to support policies (in red). The first extension is the unique *creatorID* property – which applies to the annotation as well as to the body and policy objects. The *creatorID* property is used when making decisions on *delete* and *update* operations - only the creator of a resource is permitted to modify or delete that resource.

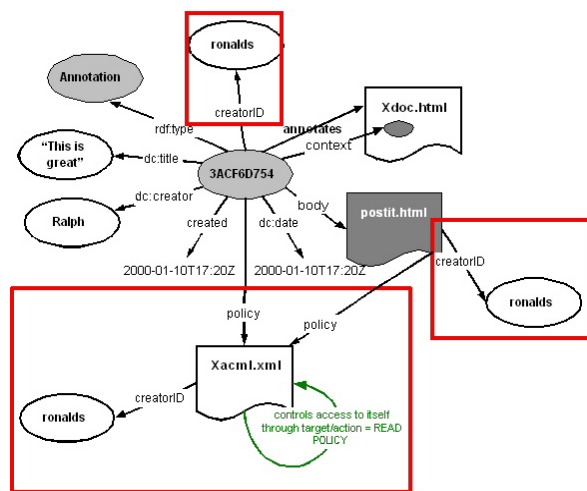


Figure 3: Extended Annotea Model

The other key extension is the policy object. Uniquely identified XACML policies are stored within the RDF repository, along with annotation bodies. Objects are linked to particular policies through their *policy* property – which is specified by a URL. This approach has the benefit of enabling multiple annotations to use the same policy. If a policy is modified, the changes will effect all those annotations associated with the policy.

3.1.2 XACML Module and Policies

This module is responsible for implementing the *Role Based Access Control* functionality and is based upon Sun's XACML implementation [2]. It makes decisions on whether a particular request is permitted based upon the role/attributes of the person making the request. The creation and reply of annotations is open to all users that are permitted to access the annotation server. Updating and deleting of objects are operations which are only available to the creator of the object. There are three types of actions permissible on annotations by users other than the creator:

- **LIST** – viewing of annotation metadata (e.g., author, creation date, language, etc.)
- **READ** – viewing of the annotation body.
- **READ_POLICY** – viewing of the annotation policy.

Figure 4 illustrates an example policy and request. Each **Policy** consists of a set of **Rules** related to whether a specific operation is permitted by a particular **Subject**. The **Subject** is described by a set of attributes which identify the credentials of a particular user e.g. affiliation, role, etc. In Figure 4a, members of the Staff "Group" have an attribute *eduPersonAffiliation* equal to "staff". This is a relatively simple example. It is possible to define more complex groups based on multiple

attributes. In the example, staff are permitted to perform all three operations on annotations whilst students are denied access to all three. Given the example policy in Figure 4a, the example request (in Figure 4b) - that a student to be permitted to read policy 123, will be denied. It is important to note that although XACML policies provide much more expressiveness than we provide, we have deliberately kept the user interface (Figure 6) simple so that end users can create policies themselves.

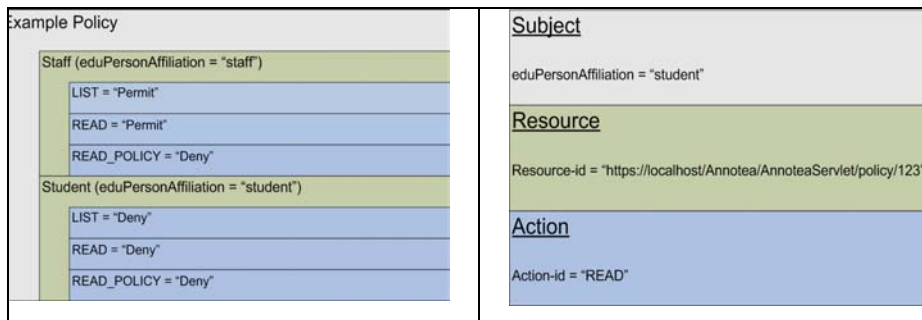


Figure 4: (a) Example Policy and (b) Example Request

The XACML module is implemented through three steps. The first step involves gathering attributes about the requester from the requester's IDP through Shibboleth's SAML assertions. Using these attributes, an XACML request is created – it specifies the action to be performed, the resource requested and the attributes of the requester. The second step involves locating the policy associated with the resource being requested by querying the RDF repository for a policy with a given URL. Thirdly the retrieved policy is compared with the request and Sun's XACML implementation generates an XACML response specifying whether the request is permitted or denied.

3.1.3 Shibboleth (SAML) Attribute Assertions

The annotation server depends on Shibboleth to provide the necessary eduPerson attributes about a requester, as provided by its origin site (Identity Provider). These attributes are used by the XACML module to make an access control decision. Shibboleth itself is a complex architecture and details are available from [22]. Each site which takes part in a Shibboleth federation may consist of either/both an origin (identity provider) and target (service provider). In terms of our annotation system, a user's attributes are provided by their origin (Identity Provider), which stores them in a directory service such as an LDAP server. The annotation server is hosted on the target site. The server is available to members of organizations that are part of the federation and have sufficient access privileges to the annotation server as defined by the host organization.

3.1.4 Jena Database

Jena [23] provides an API to an RDF repository and in the context of this system is responsible for enabling the storage and interfacing of data – including annotations, policies and annotation bodies, which are all stored in the repository as RDF instances. Jena itself sits on top of MySQL database which stores the RDF data as a relational database. The Jena API also enables us to search the annotations - via the creator, date, language and in_reply_to fields.

3.2 Client-side

The user's client side application is responsible for the user interface that enables: the retrieval and display of annotated web resources; display, search and browsing of annotations the creation, editing, deletion and attachment of annotations to Web resources; the creation, editing and attachment of access policies to annotations.

The client also has to process and exchange information in compliance with the Annotea protocol. Although a number of client-side annotation tools exist for annotating Web resources (Amaya [13], Annotzilla [12], and Vannotea [16]) none of these provide an interface suitable for also specifying XACML access policies and attaching them to the annotations. Consequently we developed our own client-side application using .NET to allow the display and editing of annotations and policies.

4 The User Interface

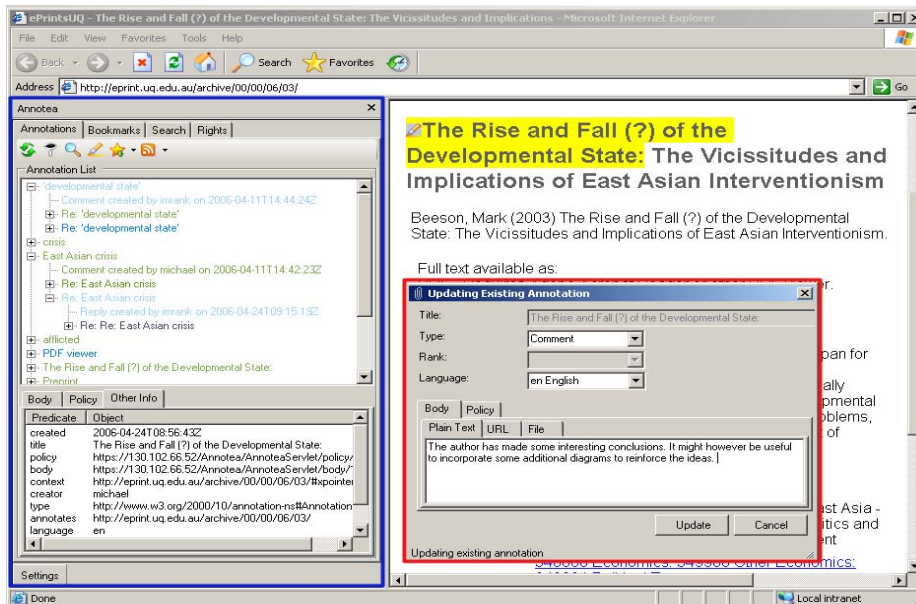


Figure 5: User Interface showing Sidebar with threaded replies and dialog box for creating annotations

For testing and illustrative purposes, we used the ePrints archive at the University of Queensland. Figure 5 shows the user interface after a user with authenticated access to the annotation server logs onto the system and retrieves a particular annotated publication. The annotations are displayed in the top left-hand frame, the details of a selected annotation are in the bottom left-hand frame and the publication is displayed in the right hand frame.

Figure 5 also illustrates the user interface for creating and attaching an annotation. We have extended Annotea to support structured annotations that contain a number of fields including hyperlinks, files, free text or controlled vocabularies.

Figure 6 shows the user interface we developed for defining groups and policies. Firstly it enables 'Groups' to be defined by sets of common attribute values. In Figure 6a, the group *uq_members* is defined as users with (*eduPersonAffiliation* = *staff*, *eduPersonOrgUnitDN* = *dke* and *eduPersonOrgDN* = *itee*) where the attributes are issued by *uq.edu.au*. The policy in Figure 6a has three groups - *uq_members*, *monash_members* and *jcu_members*. The second part of policy definition involves defining access rules for each of the groups. In Figure 6b, the Group *jcu_members* are permitted to *List* and *Read* annotations, but not *Read Policy*. This interface makes it easy for users to define new groups, modify/remove existing groups and define/modify policies.

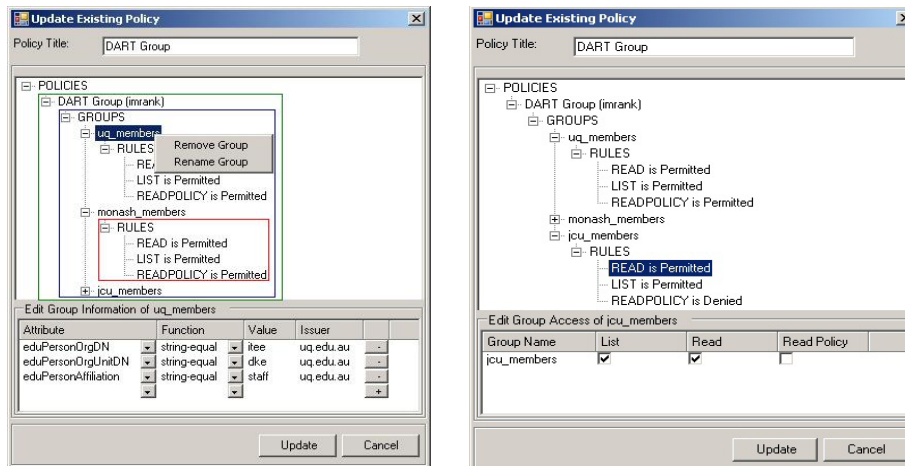


Figure 6: User Interface for defining Policy (a) Groups and (b) Rules

5 System Evaluation and Future Work

5.1 System Evaluation

To date, system evaluation has comprised thorough unit and system testing. This involved testing the creation, editing and deletion of (the complete set of possible) policies and annotations. We also tested policy enforcement by logging on as users with different attributes and modifying attributes directly in the LDAP directory. In all cases the annotation server behaved as expected – restricting user access to annotations in compliance with the policies. However the testing phase did reveal a number of problematic issues. These included:

Allowing the deletion and update of annotations can lead to ‘hanging references’ where replies refer to annotations which have been updated or deleted.

The use of URLs to identify policies enables them to be re-used and applied to multiple annotations. However this may cause problems when a policy referred to by multiple annotations is updated/deleted.

5.2 Future Work

Aspects of this work that would benefit from further investigation include:

- *Thorough user evaluation*: detailed usability studies are required to acquire user feedback and determine: functional requirements of different user groups; how intuitive, friendly and efficient the user interface is; and improvements, refinements and extensions to the system.
- *Reduced reliance on Shibboleth*: approaches other than Shibboleth and the *eduPerson* profile will enable the annotation server to be used outside of the Higher Ed sector.
- *Annotation of PDF files, database columns and spreadsheets*: the popularity of publishing scholarly information in PDF format and storing scientific data in databases or spreadsheets (e.g., Excel) indicates an increasing demand to be able to annotate data in proprietary formats.
- *Access policies based on document attributes*: the current policies are dependent on user attributes. It would be interesting to investigate policies that are based on attributes of the digital resources or their annotations.
- *Complex querying*: the integration of SPARQL to allow more complex queries over the annotation server while enforcing access constraints.
- *Web browser extension*: although Annotea-compliant extensions exist for Web browsers, they do not provide support for policy definitions. It is necessary to either extend an existing Annotea plugin or develop a new extension.
- *Post-processing of annotations*: our current version can notify authors via RSS when replies are added to their annotations. Other examples include the Multivalent Browser [24] which can review and incorporate suggested changes within documents. Ontology-based annotations or the annotation metadata (author, institution, citations etc) could be used to process annotations and automatically classify or rank the annotations and resources.
- *Scalability*: further investigation is required to determine how the system performs as the number of annotations, access policies and users grows e.g., Sesame may be more efficient than Jena.

6 Conclusions

This paper describes a secure annotation service that we have developed by combining and extending a number of existing open source technologies. Secure, trusted annotation servers are required in many domains including telemedicine, higher education and collaborative eResearch. By providing clinicians and researchers with the necessary support for authenticating the source and protecting the confidentiality and intellectual property of their annotations, they will be more willing to share their views and engage in inter-organizational collaborations with trusted colleagues. Moreover, the modular design and interoperable technologies that we have adopted, makes it easy to quickly adapt the server to a variety of different media types, different domains and different communities.

References

- [1] Internet2, "Shibboleth Project," 2005, <http://shibboleth.internet2.edu/>.
- [2] S. Proctor, Sun Microsystems, "XACML API," 2004, <http://sunxacml.sourceforge.net/>.
- [3] M.-R. Koivunen and J. Kahan, "Annotea: an open RDF infrastructure for shared Web annotations," in *Proceedings of the 10th International WWW conference*, Hong Kong, ACM Press, 2001.
- [4] R. M. Heck, S. M. Luebke, and C. H. Obermark, Department of Mathematics and Computer Science, Grinnell College, "A Survey of Web Annotation Systems," 1999, http://www.math.grin.edu/~rebelsky/Blazers/Annotations/Summer1999/Papers/survey_paper.html.
- [5] K.-P. Manuele, Jos, L. Valdeni de, and R. S. B. Marcos, "A framework for awareness support in groupware systems," *Comput. Ind.*, vol. 52, pp. 47-57, 2003.
- [6] M. Lewkowicz, G. Lortal, A. Todirascu, M. Zacklad, and M.-F. Sriti, "A Web-based Annotation System for Improving Cooperation in a Care Network," in *ICWE Workshops*, pp. 227-239, 2004.
- [7] M. Gertz, K.-U. Sattler, F. Gorin, M. Hogarth, and J. Stone, "Annotating Scientific Images: A Concept-Based Approach," in *14th International Conference on Scientific and Statistical Database Management 2002*.
- [8] R. Swick, E. Prud'hommeaux, M.-R. Koivunen, and J. Kahan, W3C, "Annotea Protocols," 2002, <http://www.w3.org/2002/12/AnnoteaProtocol-20021219>.
- [9] M. Lorch, S. Proctor, R. Lepro, D. Kafura, and S. Shah, "First experiences using XACML for access control in distributed systems," in *Proceedings of the 2003 ACM workshop on XML security* Fairfax, Virginia ACM Press, 2003.
- [10] Zope, "Zope Annotation Server," 2005, <http://www.zope.org/Members/Crouton/ZAnnot/>.
- [11] D. Brickley and R. V. Guha, W3C, "Resource Description Framework (RDF) Schema Specification 1.0," 2005, <http://www.w3.org/TR/2000/CR-rdf-schema-20000327>.
- [12] M. Wilson, Mozdev.org, "Annozilla (Annotea on Mozilla)," 2000, <http://annozilla.mozdev.org/>.
- [13] W3C, "Amaya," 1994, <http://www.w3.org/Amaya/>.
- [14] W3C, "Perllib Annotations Server HOWTO," <http://www.w3.org/1999/02/26-modules/User/Annotations-HOWTO>.
- [15] M.-R. Koivunen, W3C, "Extending the Annotea addressing schema," 2003, <http://www.w3.org/2001/Annotea/Plan/context/newcontext.html>.
- [16] R. Schroeter, J. Hunter, and D. Kosovic, "Vannotea - A Collaborative Video Indexing, Annotation and Discussion System For Broadband Networks," in *Knowledge Markup and Semantic Annotation Workshop, K-CAP 2003*, Sanibel, Florida 2003.
- [17] J. Hunter, R. Schroeter, B. Koopman, and M. Henderson, "Using the Semantic Grid to Build Bridges between Museums and Indigenous Communities," in *GGF11 Semantic Grid Applications Workshop*, Honolulu 2004.
- [18] N. Harris, S. McLeish, and J. Paschoud, "Access Management Report," London School of Economics 2002.
- [19] R. L. Morgan, S. Cantor, W. Hoehn, and K. Klingenstein, "Federated Security: The Shibboleth Approach," *Educase Quarterly*, vol. 27, pp. 12-17, 2004.
- [20] R. Cover, Oasis, "Cover Pages: Security Assertion Markup Language (SAML)," 2005, <http://xml.coverpages.org/saml.html>.
- [21] Directory Working Group (MACE-Dir), Internet2 Middleware Architecture Committee for Education, "EduPerson Object Class Specification (Draft)," 2006, <http://www.nmi-edit.org/eduPerson/draft-internet2-mace-dir-eduperson-latest.html>.
- [22] T. Scavo and S. Cantor, Internet2, "Shibboleth Architecture," 2005, <http://shibboleth.internet2.edu/docs/internet2-mace-shibboleth-arch-conformance-latest.pdf>.
- [23] B. McBride, "Jena: A Semantic Web Toolkit," *IEEE Internet Computing*, vol. 6, pp. 55-59, 2002.
- [24] A. P. Thomas and W. Robert, "The multivalent browser: a platform for new ideas," in *Proceedings of 2001 ACM Symposium on Document engineering*, Atlanta, Georgia, USA, ACM Press, 2001.