Authentication and Authorization Infrastructure: Portal Architecture and Prototype Implementation

Marc-Alain Steinemann
Thomas Spreng, Aljoscha Bachmayer, Torsten Braun
Christoph Graf
Martin Guggisberg

December 24, 2003

IAM-03-012

Authentication and Authorization Infrastructure: Portal Architecture and Prototype Implementation

Version 1.0

December 24, 2003

Marc-Alain Steinemann
Thomas Spreng, Aljoscha Bachmayer, Torsten Braun
([steine/spreng/bachmaye/braun]@iam.unibe.ch)
Institut für Informatik und Angewandte Mathematik, Universität Bern
Neubrückstr. 10, CH-3012 Bern

Christoph Graf
(graf@switch.ch)
SWITCH
P.O. Box,
CH-8021 Zürich

Martin Guggisberg
(martin.guggisberg@unibas.ch)
Departement für Informatik,
Klingelbergstr. 50, Universität Basel
CH-4056 Basel

KEYWORDS: authentication, authorization, accounting, e-learning, e-community

Abstract A very federal higher educational system with many universities and as many different student authentication systems exists in Switzerland. Initiated by the Swiss Virtual Campus and by the demand for more inter-university work and student mobility, SWITCH, the Swiss education and research network started to evaluate authentication and authorization architectures.

Independently of the selection of Shibboleth Authentication and Authorization Infrastructure for Switzerland, there are Resources that have to be adapted to the chosen infrastructure. To ease this adaptation process, we present a generic Authentication and Authorization Infrastructure portal as the connecting unit between Home Organizations and Resources.

The architecture as well as the portal's environment followed by a chapter about a prototype implementation get discussed.

1 Table of Content

2 Introduction

This document describes an Authentication and Authorization Infrastructure (AAI) portal architecture and a prototype implementation. Authentication and authorization infrastructures are middleware systems consisting of a set of protocols that allow the delegation of authentication and authorization issues to different instances. Authentication is executed by the user's organization and authorization by resources such users want to access. Authentication and authorization infrastructures provide all the necessary mechanisms to enable users, organizations and resources to participate in the system. By these means, authentication and authorization infrastructures connect user communities to resources. Users belong to at least one home organization, for example a university. Resources are systems that provide media content and can be e-learning courses or content management systems. A simplistic overview is given in Figure 2.1.

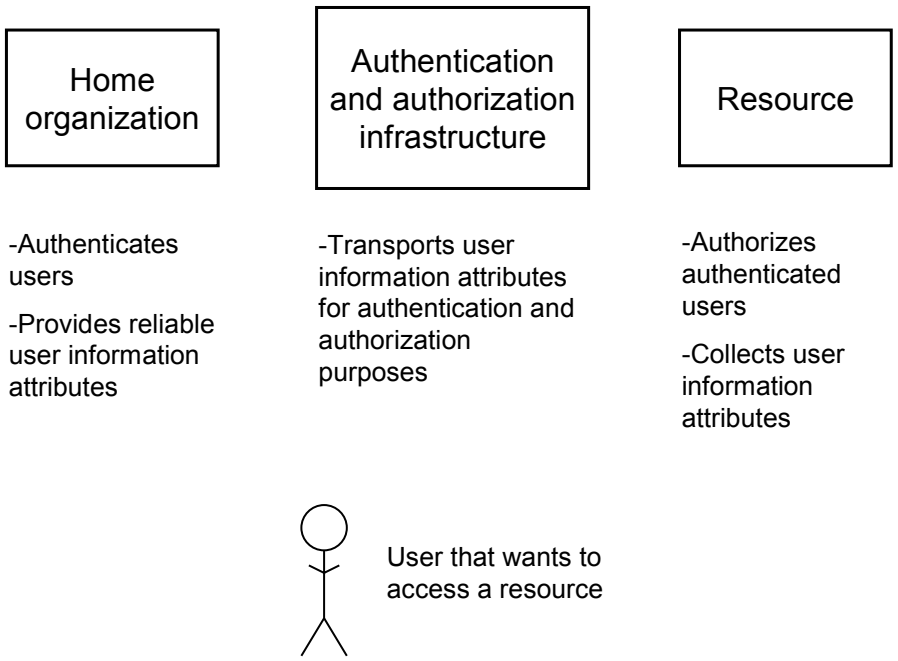

Figure 2.1: Entities in an authentication and authorization infrastructure.

The AAI portal is located between an authentication and authorization infrastructure and a protected resource. Users belonging to an AAI-connected home organization connect to resources hosted by the AAI portal. Figure 2.2 shows a resource's owner site. The AAI interface is responsible for the resource owner's site connection to the AAI. The AAI portal receives user information attributes from the AAI interface and stores them in its database. Users that successfully subscribe to an AAI portal hosted resource agree with the transfer of selected user attributes to the resource on demand of the resource. The resource stores user data in its own database or relays on the AAI portal's database. It is possible that the resource sends back user information to the AAI portal. Beside the access management, the AAI portal provides enhanced user management. The portal allows resources to directly ask users for additional information and enables users to manage self-provisioned user information. The portal further contains basic and enhanced *e*-community management features (community management features designed for the Internet), which can be added in a modular way.

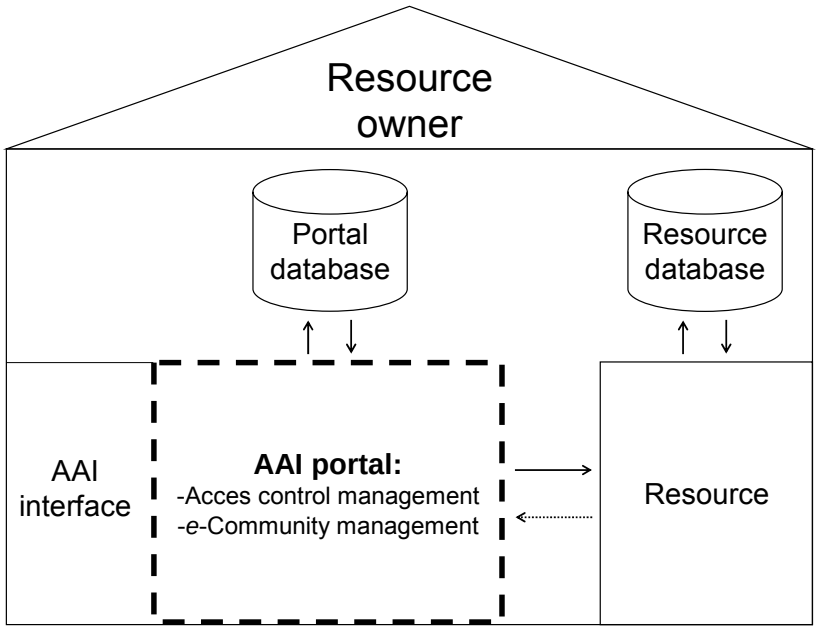

Figure 2.2: The AAI portal at a resource provider's site.

The advantage of connecting resources to authentication and authorization infrastructures lies in the decreased administration overhead and the reliable user information in the form of attributes provided by the user's home organization. Resources define their access policy based upon required user information attributes instead of user identities. Further user management, such as access levels or fine granular access structures remain a resource internal issue.

User Alice, belonging to a home organization named University of Bern tries to access a resource called TCP/IP course. Alice's home organization as well as the resource is connected to the same AAI. University of Bern stores information attributes about Alice that are released to the resource when Alice tries to access it. Only those attributes are released that are allowed by University of Bern's and Alice's attribute release policy. The TCP/IP course receives those attributes and compares them with the own attribute acceptance policy that describes which information attributes are required so that user Alice can access the resource. If Alice's home organization provides enough attributes she can now access the TCP/IP course.

The advantage for home organizations lies in the immediate increase of accessible resources by their members. Authentication and authorization infrastructure users get subscribed by their respective home organizations and have to agree with the local subscription policy. Home organizations define their information release policy to protect their members' privacy on the one hand and on the other hand to enable members to access resources. Resource providers reduce administration overhead as they do not have to manually verify and subscribe foreign persons to their own AAI-connected resources.

The advantage for users lies in a simplified resource access procedure. No on-site or mail registration procedure is needed. It is a precondition in authentication and authorization infrastructures that user information released by home organizations is reliable and accepted by the resource providers. Privacy sphere is guaranteed in two ways: First, users still authenticate with their own home organizations and never with resources. Secondly, users determine which information attributes about themselves are released to the resource they want to access.

The authentication and authorization infrastructure itself is not subject to this document although a basic introduction can be found in Chapter 3. The introduction also covers the solution described in this architecture and related work to authentication and authorization infrastructures. In Chapter 4 we describe a middleware architecture for user and resource management for all types of resources and AAI. The architecture is still extensible in many directions such as accounting, personalized user resources or content adaptation as partly presented in Chapter 4.4. The portal uses a plug-in concept for resource adaptors that provides a high reusability of existing resource adaptors. Chapter 5 describes the implemented prototype software which is licensed under the general public license.

3 AAI Architectures

3.1 Introduction to AAI

The problem of connecting resources to authentication and authorization infrastructures concerns many e-learning projects such as the Virtual Internet and Telecommunications Laboratory of Switzerland (VITELS) [Vitels and Sm02] and the nano science laboratory Nano-world [Gm01]. In some cases, where there is a high pressure from the user side and a resulting business case, resources get adapted to one or more AAI. For example [Webct] may be adaptated to Shibboleth [Shibboleth]. A generic portal hosting non-AAI-enabled as well as AAI-enabled resources to connect them in a simple way to authentication and authorization infrastructures seems to be a new idea.

Authentication and authorization issues were expected to be solved by Public Key Infrastructures (PKI) [Wj01]. The complexity of PKI proved to be a major obstacle in establishing them. Especially the problem of cross-certification and data protection for users could not be solved satisfactory. There exist few commercial certification authorities worldwide. In Switzerland, Swisskey failed its business case and discontinued its services by the end of 2001. PKI have proofed to be useful in other business cases such as online banking, secure sockets layer or IP security.

Nevertheless, user authentication and authorization issues had to be solved and a new generation of infrastructures was developed, called authentication and authorization infrastructures.

3.2 General Case with AAI Portal

A typical authentication and authorization infrastructure architecture environment with non-AAI-enabled and AAI-enabled resources most likely consists of the elements shown in Figure 3.1. Additionally to the typical elements also non-AAI-enabled resources and the AAI portal are shown.

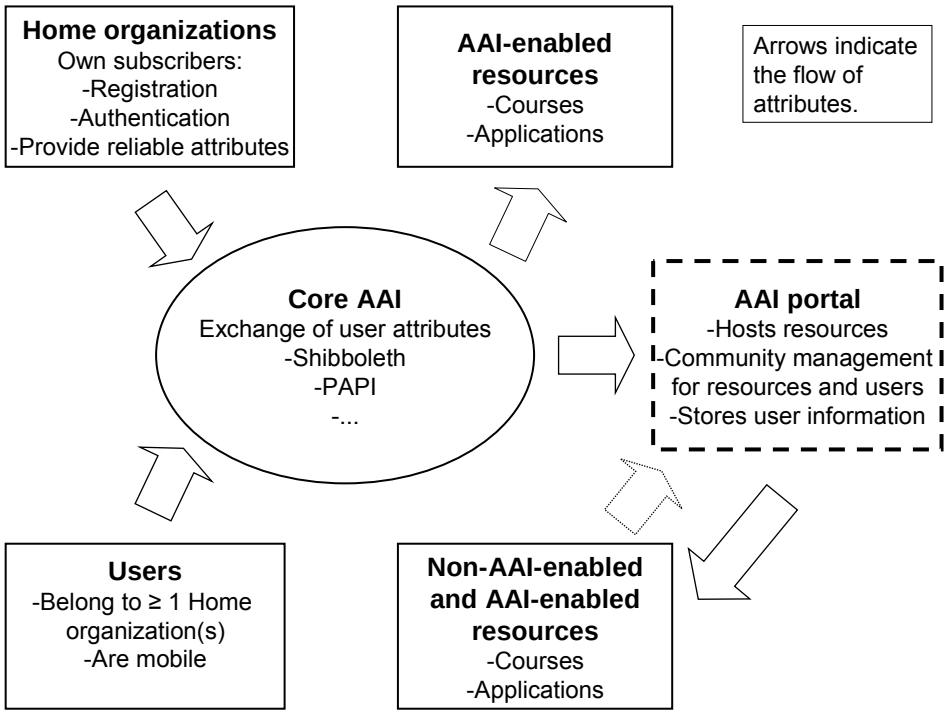

Figure 3.1: Typical AAI environment with the AAI portal.

Home Organizations (HO) are universities or other entities, where potential resource users such as students and staff members are registered. Home organizations may also be created for other organizations that need to participate in the AAI. It makes sense to maintain a virtual home organization for users not belonging to an existing home organization, such as guest resource users from abroad. User registration is fully operated by home organizations that comply with the AAI policy. The possibility should always exist to offer self-registration for visitors that then are clearly

marked as such by specific attributes. Home organizations maintain the databases with the data of the registered persons. The database must have an interface to the AAI. Home organizations only authenticate their own members. The AAI does not dictate any limitations or policies for the authentication process. Authentication can be done for example by means of username and password or smart cards.

Users possess at least one account at their respective home organization and are exclusively authenticated by their proper home organization. A user can be a member of more than one home organization and have more than one role in a certain home organization, for example as student and teacher in different units of the home organization. Each user's role leads to a role dependent account. Users obey the home organizations policy, which itself is conforming to the AAI policy. Through the attribute release policy, users determine which personal data the home organization is allowed to release to resources.

Resources are web-based resources or services offered by providers to users of the authentication and authorization infrastructure. Resources belong either to the group of AAI-enabled or non-AAI-enabled ones.

AAI-enabled resources are connected directly to the AAI; they have specific interfaces to the AAI. Based on users' attribute release policy and resources' Attribute Acceptance Policy (AAP), users are able to access resources. This is a case of user authorization that is executed by the resource owner itself. Only users providing the required attributes are granted access to the resource. Enabling resources for AAI is expensive and time consuming. Since specific AAI implementations differ, resource interface must be re-implemented for each kind of AAI and resource.

Non-AAI-enabled resources cannot be directly connected to the AAI as they lack the specific interfaces to the AAI. Therefore, AAI users cannot access those resources and take profit from AAI-related benefits such as Single Sign-On (SSO) or attribute release and acceptance policies.

To be able to anyhow connect those non-AAI-enabled resources via AAI, a workaround must be applied. This document describes and presents a prototype implementation of such a workaround that is called AAI portal. The AAI portal is situated between the core AAI (all protocols and mechanisms that do not belong to the user's site) and the non-AAI-enabled resources. Non-AAI-enabled resources can be everything, from applications such as Telnet to web learning platforms.

The AAI portal is a mediator between the core AAI and non-AAI-enabled as well as AAI-enabled resources as shown in Figure 3.2. The interfaces depicted as plugs and connectors have to fit to each other. The advantage of this plug-in concept lies in the reusability of the interfaces. Once an interface (known as resource adaptors) has been implemented, other similar resources can reuse a resource adaptor or adapt it according to their own needs. On the right side four example resources are shown. In this case, each of the resources uses a different resource adaptor.

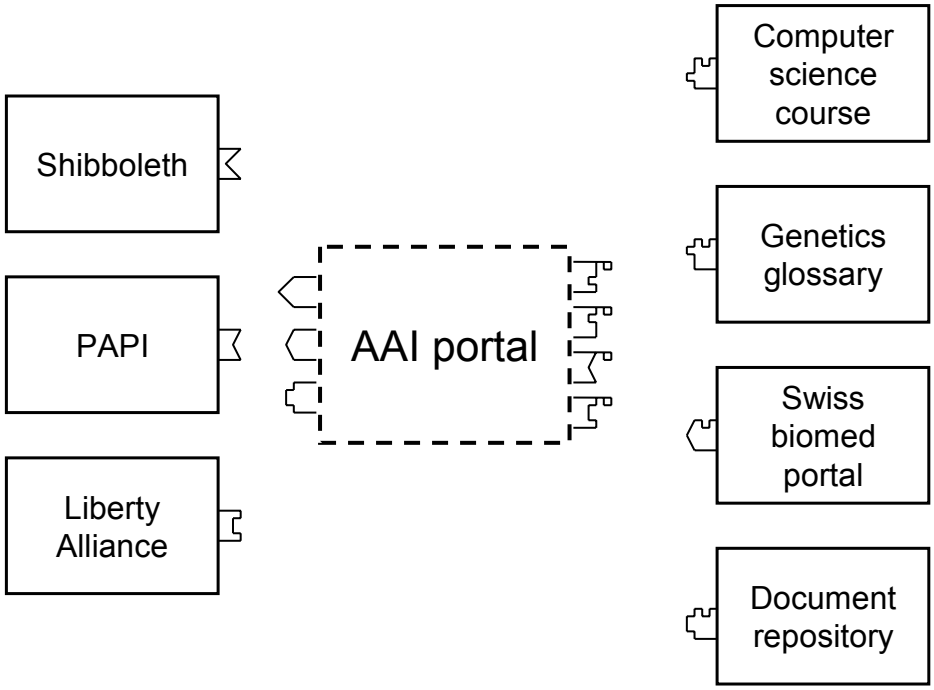

Figure 3.2: AAI portal with its interfaces to AAI and resources.

The AAI portal acts as an AAI-enabled resource towards the core AAI and as user, resource and community management portal towards resource owners and users as shown in Figure 3.3.

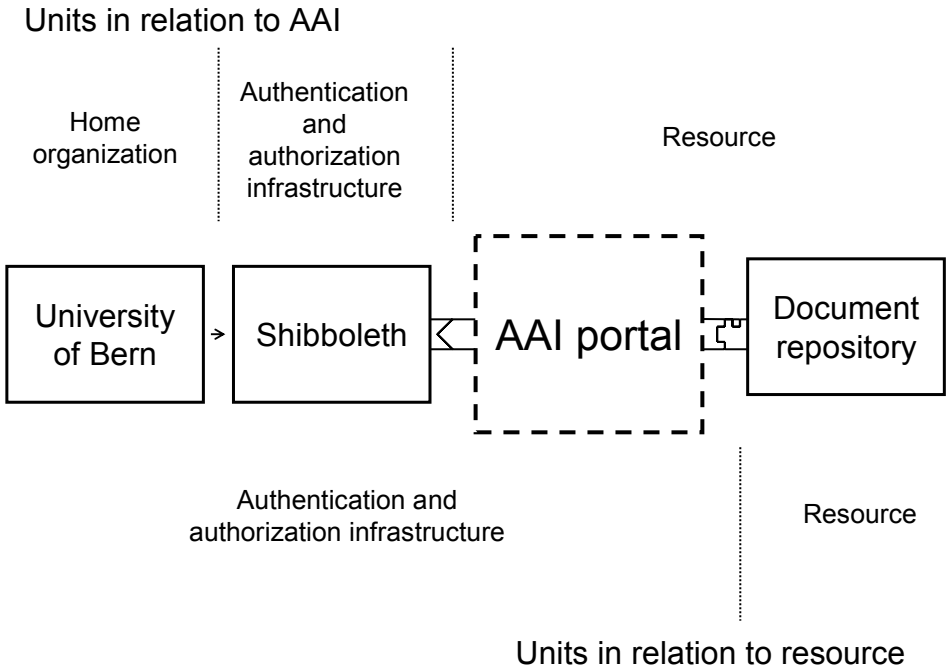

Figure 3.3: Units in relation to the AAI portal.

Persons that log on to the AAI portal are known users (AAI subscribers) with a set of AAI provided user information attributes. We call them 'reliable attributes' since the user's home organization did provide them. The unique AAI identity is always an element of that set. This is the only technically required attribute as it is the identifier for each user in the AAI. The unique AAI identity is an attribute in the form of a cryptic hash @HomeOrganization.ch (i.e. fg98wessed@unibe.ch). Each home organization defines the generation procedure for the cryptic hash on it’s own but conforming to data protection laws. The AAI portal’s attribute acceptance policy can be individually set, by default it is set

to request all the available attributes from the user's home organization. Resources that are hosted by the AAI portal set their attribute acceptance policy individually.

A special feature of the AAI portal is the ability to collect additional user information attributes, directly provided by the user and therefore called 'non-reliable' attributes. This allows users that do not comply with the resources' attribute acceptance policy by means of the reliable attributes delivered by the AAI to provide the missing data. The AAI portal can store user information on the basis of the users' unique AAI identity. The whole AAI portal features are described in greater detail below.

The core AAI includes all the protocols and services belonging to the AAI but not to home organizations or resources. It is responsible for the transport of information regarding user authentication and data needed for user authorization between home organizations to resource providers. The term "core" is only used in this document to distinguish the resource provider's site, without the AAI software installed in the resource provider's web server, from the rest of the AAI (i.e. everything after the shibbolized web server at the resource's place). For example user authentication is part of the AAI but is performed by the respective home organizations. The term AAI is used for the mechanisms and protocols used to link all home organizations, resources and users together. The AAI is middleware and mostly invisible to users. There are few opportunities where users get into contact with graphical user interfaces, for example to specify their information release policy. The AAI must obey to the data protection laws of the place it operates in. Many countries have data protection laws in use. In Switzerland, where this AAI portal will first be implemented, exists a federal data protection law on top level and 26 cantonal data protection laws on the next level below.

The operators of the AAI portal operate databases that contain sensitive user data. Thus in most areas the operators must conform to the local data protection law.

3.3 PAPI

Point of Access to Providers of Information (PAPI) [CL01] is the authentication and authorization infrastructure that is mainly used by Spanish libraries and has been developed by the Spanish national research network provider RedIRIS [Rediris]. PAPI is a system for providing access control to restricted web-based information for resources across the Internet independent from IP origin and open for mobile users. It intends to keep authentication as an issue local to the user's home organization, while leaving full control over the resources to the information providers. The authentication mechanisms are designed to be as flexible as possible, allowing each organization to use its own authentication schema, maintaining user privacy, and offering information providers the attributes required for access control decisions. Moreover, access control mechanisms are transparent to the user and compatible with the most commonly employed web browsers, i.e., Netscape, Internet Explorer, Lynx, and any operating system.

PAPI's main parts and how a user accesses a PAPI-protected resource are depicted in Figure 3.4. Figure 3.4 and the description have been taken from the above mentioned architecture document and have been slightly adapted. Security assertions are placed into cookies. The security measures taken in this process can be seen in the description of the single steps described below. The resource provider grants access based on visitor's public attributes. User privacy is guaranteed as each user is identified by a unique code and only the respective home organization is able to relate personal information to this code.

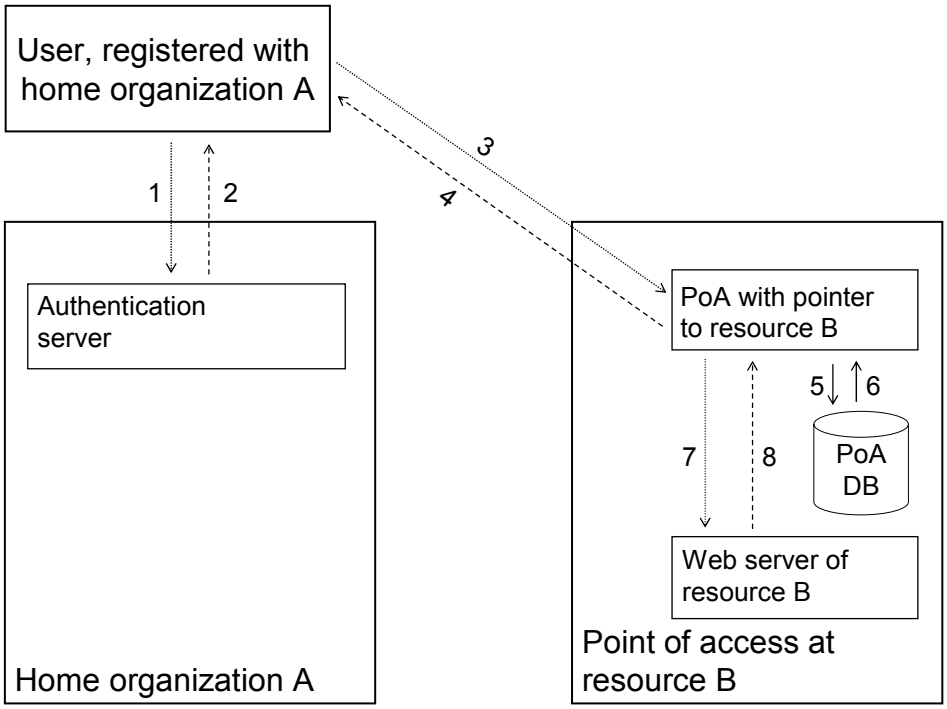

Figure 3.4: PAPI architecture.

The steps of accessing a resource are as follows: 1) A user sends his or her authentication data to its home organization's authentication server. 2) Upon a successful authentication, the authentication server sends back a list of temporarily signed URL of allowed Points of Access to resources (PoA) for the respective user. Each URL contains a user code, the authentication server, a path, the expiration time and the sign time. 3) The user accesses a PoA, which hosts the desired resource. He or she issues a HTTP-request and sends two cookies to the PoA. The first cookie, the so called Hcookie contains the user code (a unique user identifier), the authentication server, a path, the expiration time and a random block. The second cookie, the so called Lcookie contains the user code, the authentication server, a path and the creation time. 5) The PoA stores user information contained in the cookies in its database and 7) performs a HTTP-request to the desired resource's web server. 8) The resource sends back the web page to the PoA. 6) New cookies are generated. 9) The web page together with new H and L cookies are sent to the user.

3.4 Shibboleth

Internet2's initiative is called project Shibboleth [Shibboleth and CE02]. Shibboleth is one of the most promising authentication and authorization architectures under development today. Shibboleth is a joint project of Internet2/MACE (Middleware Architecture Committee for Education) and formerly IBM. It aims to develop an architecture for standard-based vendor-independent web access control infrastructure that can operate across institutional boundaries.

Shibboleth allows federated user administration (i.e. user management is delegated to home organizations), resource access authorization based on user attributes and active management of privacy. It is fully based on standards and uses Security Assertion Markup Language (SAML) [Saml] for the message and assertion formats, and protocol bindings, developed by the OASIS Security Services Technical Committee [Oasis]. In Switzerland, SWITCH leads the implementation of Shibboleth for Swiss universities [Gc03].

The main parts of Shibboleth and how a user gets a handle to access a resource are depicted in Figure 3.5. Figure 3.5 and the description have been taken from the above mentioned architecture draft and slightly adapted.

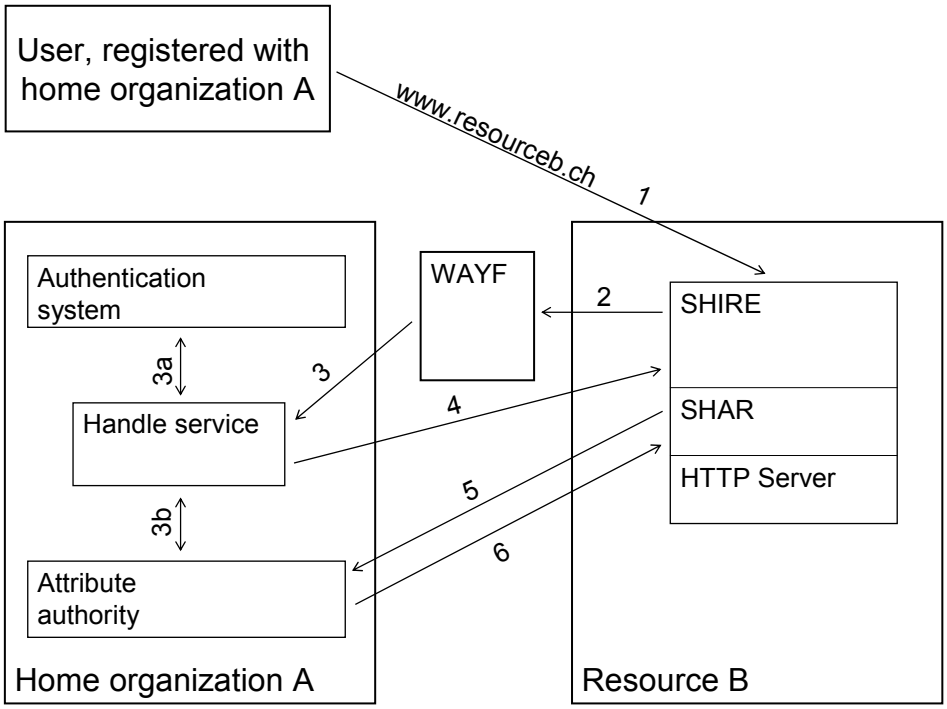

Figure 3.5: Shibboleth architecture.

The steps of accessing a resource described are as follows: 1) A user accesses resource B and gets to the Shibboleth Indexical Reference Establisher (SHIRE). Steps 2 to 6 are HTTP browser redirects. 2) The Shibboleth indexical reference establisher directs the user to the Where Are You From service (WAYF), a service located somewhere in the Internet that possesses full information about connected home organizations together with the handle acceptance URL, the URL where Shibboleth indexical reference establisher expects the answer from the Handle Service (HS), and the user's desired target URL as parameters. 3) The WAYF service redirects the user to its home organization's handle service together with the Shibboleth indexical reference establisher parameters. The handle service makes sure that the user is authenticated. 4) The handle service sends back an opaque (not transparent) handle associated with the user to the Shibboleth indexical reference establisher's "handle acceptance URL" by means of an HTML form posting. The user's desired target URL is passed as an additional parameter. Impersonation counter information is presented as part of a "package" around the handle. 5) The Shibboleth Attribute Requestor (SHAR) sends an Attribute Query Message (AQM) to the Attribute Authority (AA) and 6) receives an Attribute Response Message (ARM). Upon evaluation of the attribute response message the resource manager grants access to the protected resource.

3.5 Other Systems

Liberty Alliance

Liberty Alliance [Wt03] commenced in 2001 and aims to provide open standards for a trust network with a strong emphasis on commercialization. Accounting is a must for commercialization and raises many questions about privacy and security. Liberty Alliance is based on federated user management.

Microsoft .NET Passport

Microsoft .NET Passport [Passport] has one central database where all sensitive user data is stored. From this only one company administrates the stored user data. All resources have to query the same database. Standards are not open and thus make the system insecure compared to open source systems where the code is freely available for anybody to verify. Users cannot define their information release policy to each resource they visit. Several severe security issues have been discovered in the past.

4 AAI-Portal Design

4.1 Introduction

This chapter describes the AAI portal's design and architecture. After a short listing of very useful features the portal offers for user, resource and community management, Chapter 4.2 presents the proposed user roles on the portal. Chapter 4.3 presents a small selection of use cases. Chapter 4 puts the portal system in context its neighbors and Chapter 4.5 presents the system design.

The AAI Portal's Key Features:

The portal provides many features for resource and user management. The list below represents a summary of the key features.

- Hosting of one or multiple resources.
- Resource-related management functions.
- Resources can be visible or invisible on the resource list.
- Resources can be open or closed for subscription.
- Resources can be set open for all or subscribers can be directed to a waiting list where tutors can manually grant or deny access.
- Resource can be suspended.
- Resources can be deleted from AAI portal.
- Users can be blocked out from resources.
- Users can be notified about status changes by Short Message Service (SMS) or e-mail.
- Tutors can notify their users by Short Message Service or e-mail.
- Additional *e*-community management features can be added on a modular base, such as personalized news or chat.
- Attribute request policy can be set individually for each resource.
- Attribute release policy can be set individually by each user and for each resource the user subscribes to.
- Missing attributes according to the resource's attribute request policy can be requested from the user.
- User provided information is marked as `user provided` in the database.

4.2 User Roles

Each user accessing the AAI portal can act in different roles. We propose to implement at least three different user roles. The super user is the AAI portal administrator and allowed to configure everything on the AAI portal. The second user role is the resource administrator. He acts as owner of resources hosted on the AAI portal. The third user role is the resource user, sometimes simply called 'user'. These are persons accessing resources hosted on the AAI portal.

When users access the AAI portal, a set of reliable user attributes is provided by the AAI. In rare cases, users access the AAI portal through the built-in direct access interface. In this case, their user rights have been manually defined by a portal administrator for example during the creation of the respective account.

The three proposed user roles are listed below, together with their specific functions.

Resource users (for example students)

Resource users are able to:

- List all resources visible to them.
- List the set of already subscribed resources.
- List the set of resources with pending subscription requests.
- Subscribe to and unsubscribe from resources.
- View their attributes the AAI has released to the AAI portal.
- Enter additional required information (attributes) for specific resources.
- Define which attributes are released to which resource.

- Close their Account in the AAI portal.

Resource administrators (for example tutors or resource providers)

Resource administrators are able to:

- Own one or more resources.
- Define how the resource is integrated into the AAI portal.
- Define which way users are redirected to the resource.
- Define which attributes a resource user must provide for getting access to their resource.
- Specify additional (not yet existing) user attributes.
- Define the resource's attribute acceptance policy.
- List their resources.
- List subscribed users of these resources.
- View the information their users released to the AAI portal.
- Differentiate between AAI and user delivered attributes.
- Access the pending subscription list of a resource.
- Accept or reject subscription requests.
- Delete users at any time from a resource.
- Suspend a user from a resource.
- Suspend a resource.
- Release a notification messages about to users.
- Make resources visible or invisible.
- Open or close resources for subscription.
- Interlace subscription to a manual selection.
- Add or delete users to the AAI portal.

AAI portal administrator (AAI portal administrators are able to do anything resource administrators are allowed to do. additionally, AAI portal administrators are able to:

- Appoint resource administrators.
- Initialize the AAI portal.
- Add or remove resources.
- Add or remove resource administrators.
- Specify short message service gateway.
- Specify e-mail gateway.

4.3 Selected Use Cases

4.3.1 Introduction to Use Cases

A use case is used to identify and view system requirements in terms of how the user will see them. The use case consists of possible sequences of interactions between systems and users in a particular environment and related to a particular goal.

This chapter presents a selection of the many possible use cases of the AAI portal.

4.3.2 Use Case 1: User Logs-in to the AAI Portal

Figure 4.1 shows user Bob that would like to log-in to the AAI portal. He opens a web browser and enters the AAI portal's URL as shown in step 1. The AAI portal's AAI-enabled web server recognizes an unauthenticated user without credentials and redirects this user in step 2 to the respective AAI services. The AAI services together with Bob's help redirect him to his home organization where the authentication process takes place. After Bob ha been equipped with credentials and security assertions in an opaque handle he is redirected again, this time back to the AAI portal as shown in step 3. The AAI portal's AAI components accept Bob and let him access the entry page. The log-in process is now completed. Step 4 anticipates a possible action Bob takes: he accesses a protected resource and is therefore redirected to it. Steps 2 and 3 are AAI-type specific. They can for example be replaced by PAPI or Shibboleth as explained in Chapter 3. A more concrete example for Shibboleth is given below in Chapter 5.1.

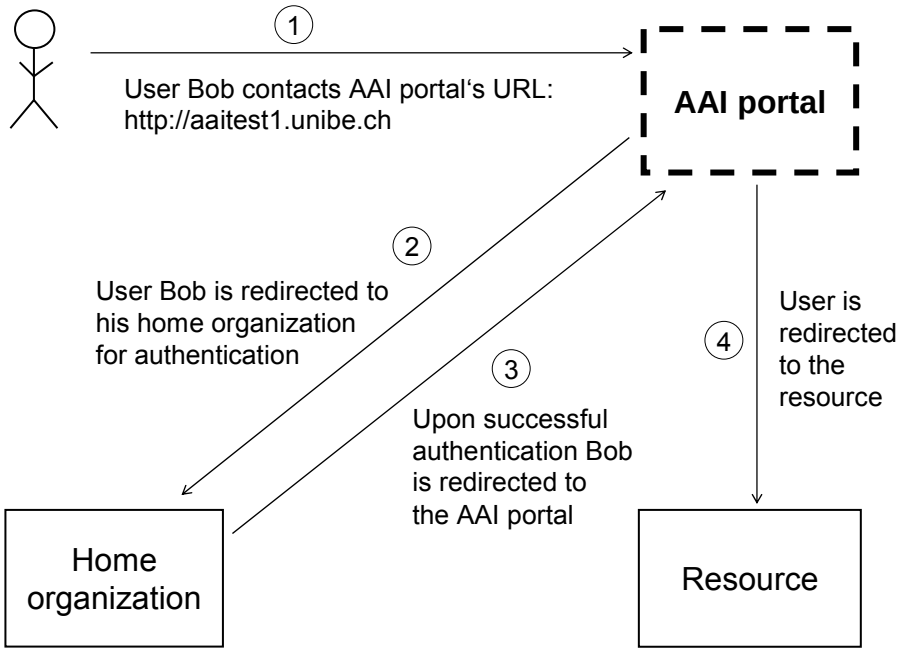

Figure 4.1: Redirection processes to get to a resource.

4.3.2 Use Case 2: Resource Provider Advertises a Resource

Resource providers that would like to open their protected resources without having to maintain an own user administration have the possibility to adapt their resource to each type of AAI in question or to adapt their resource only once to the AAI portal. The AAI portal administrator adds the resource owner to the AAI portal users in the role of a resource administrator.

Figure 4.2 shows the interaction diagram for the process of advertising a protected resource to AAI portal users. The resource provider applies to the AAI portal administrator for becoming a resource administrator and thereby getting the possibility to add his resource to the AAI portal. Upon a successful application, the resource owner adds his resource and opens it for subscription. The newly added resource is advertised at the AAI portal's entry page as new resource. The now operational resource has to be managed. In case the resource owner decides to perform access control, he has to admit access to each applying user. Misbehaving users have to be unsubscribed or their access suspended, for example if they do not pay their utilization fee.

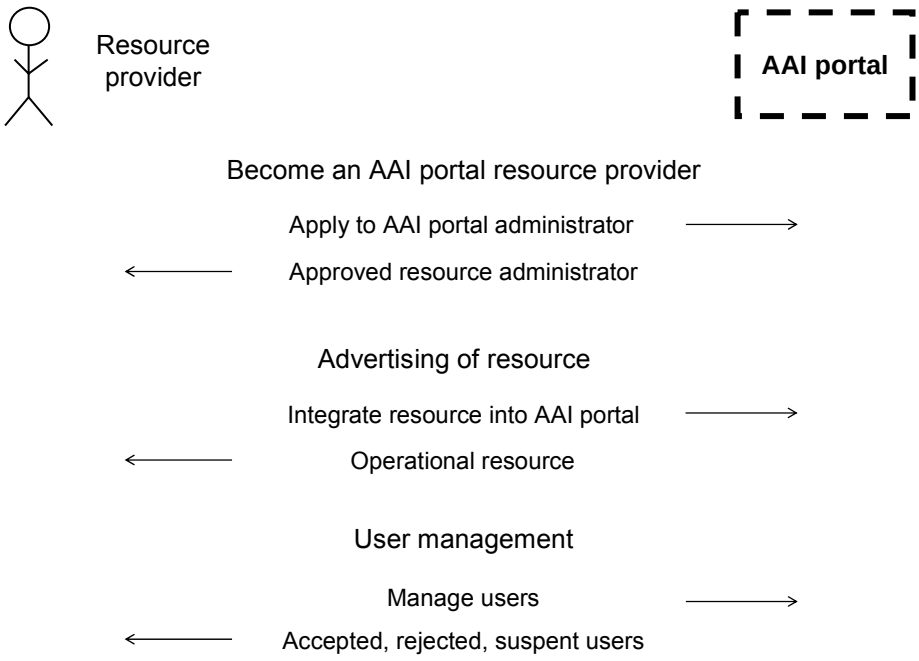

Figure 4.2: Interaction diagram for resource advertising.

4.3.3 Use Case 3: User Subscribes to a Resource

Figure 4.3 shows the interactions between Bob and the AAI portal when Bob tries to subscribe to a resource R. In a first step, Bob has to access the AAI portal and go through the above described authentication process. After that he gets to the AAI portal's entry page and can access a list of all hosted resources.

Bob now chooses resource R and after reading the information about the resource and its owner, decides to subscribe to the resource. The AAI portal now requests missing user attributes, the difference between the user's attribute release policy and the resource owner's attribute acceptance policy.

If the resource is open and freely accessible to all AAI users, he now gets redirected to the resource. If there is a list for pending subscriptions, Bob has to wait until a resource administrator accepts or rejects his subscription request. Upon this action, Bob receives a status change message by e-mail or short message service.

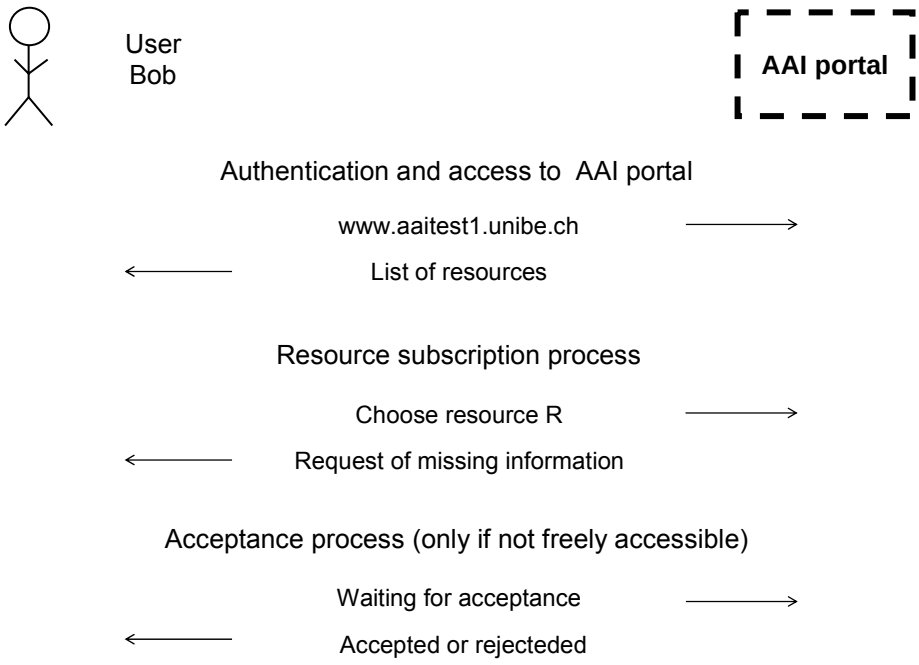

Figure 4.3: Interaction diagram for resource subscription.

4.3.4 Use Case 4: Manually Adding Users to AAI Portal

Normally, the AAI should be able to include all types of users, from regular students and staff to visitors. All the same the AAI portal offers the possibility to add users directly. Figure 4.4 shows user Bob that would like to get access to a resource hosted on the AAI portal. He has to get into contact with an administrator, either directly or by the help of the resource owner. This contact can be by a phone call, a letter or e-mail or any other media. Subsequently, Bob and the administrator regulate the subscription details. If both agree Bob is added to the AAI portal users and gets a confirmation notice. Bob can now access the AAI portal but not as above described via AAI but by an access interface directly on the AAI portal. Adding a user directly to the AAI portal's user database does not open him/her access to any other AAI component.

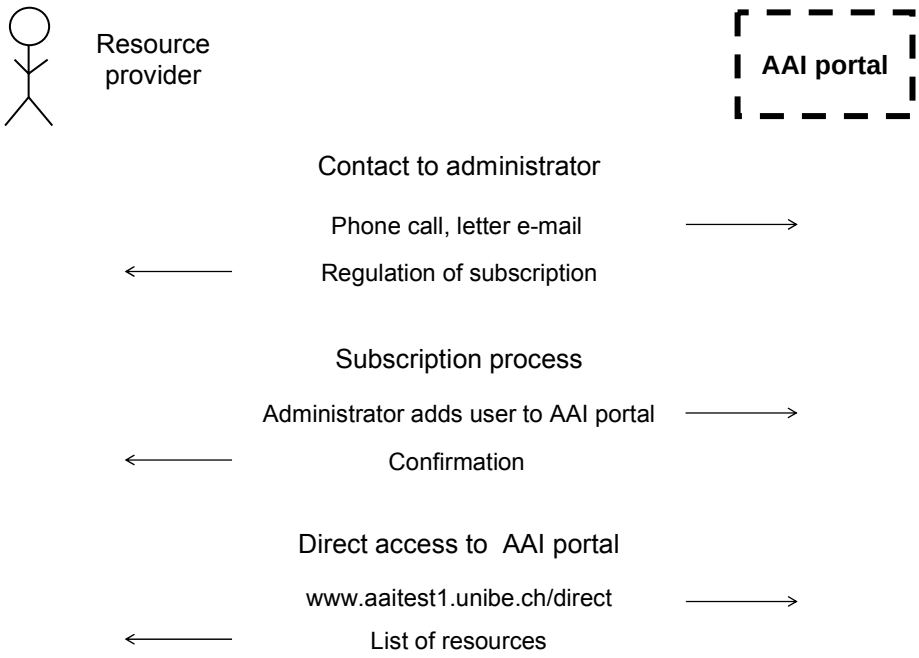

Figure 4.4: Interaction diagram for manually adding users to AAI portal.

4.4 AAI Portal's System Context

This chapter aims to put the AAI portal and its various modules into context with its interacting neighboring systems. Chapter 4.4.1 views the AAI portal in the middle of users and protected resources. Chapter 4.4.2 visualizes and describes the user information attribute flow from home organizations towards the AAI portal and its hosted resources. Chapter 4.4.3 describes the interfaces towards AAI and the chosen adaptor concept. In Chapter 4.4.4 we present the very important concept for the interfaces to resources, called resource adaptors and in Chapter 4.4.5 we address the interfaces to enhanced community management features that can be plugged into the portal. Chapter 4.4.6 discusses authorization issues sometimes caused by certain resource types.

4.4.1 Mediator between AAI and Protected Resources

The AAI portal is designed to be a mediator between the mechanisms of the AAI with the connected home organizations and users on one side and between protected resources on the other side. Figure 4.5 shows a context diagram with the AAI portal depicted as a circle.

On the left side there are users and administrators that access the AAI portal. Administrators access it to manage user and resource data and users access it because they want access to a resource that is protected by the AAI portal. Users get a resource list and the possibility to access resources whereas administrators get a palette of management features.

Protected resources get a large potential audience without having the burden to manage those users and they can advertise their resource on the AAI portal.

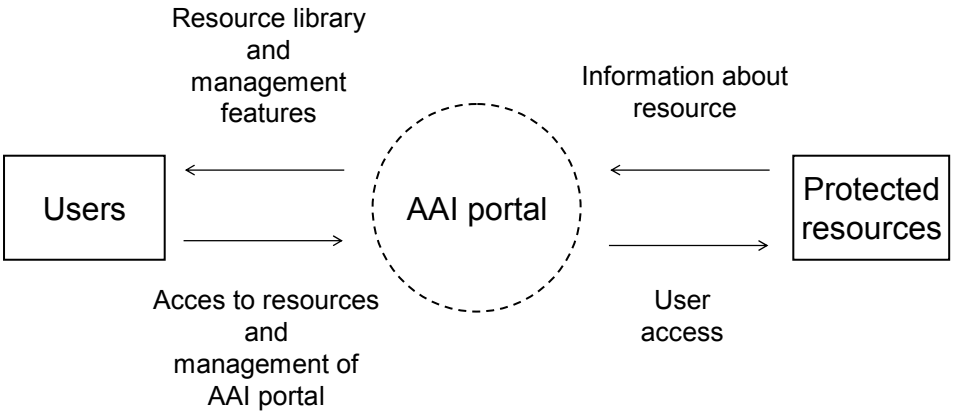

Figure 4.5: AAI portal's context diagram.

The AAI portal can be split up in its functional parts as shown in Figure 4.6 On the left side, the interfaces from the core AAI that could be more than one at a time and from directly connecting users are shown. The connecting unit links the interfaces to three user areas. Connecting users can access all the AAI portal's areas. On the right side, the interfaces to a set of hosted resources are depicted. There is no minimum or maximum number for hosted resources. Each of the interfaces, either to an AAI or to a resource, has to be designed individually but can be reused again whenever the same type of AAI or resource is connected to the AAI portal. The attribute acceptance policy of the AAI interfaces is per se set to request all the user's attributes the home organization can deliver. The AAI portal's database stores collected data.

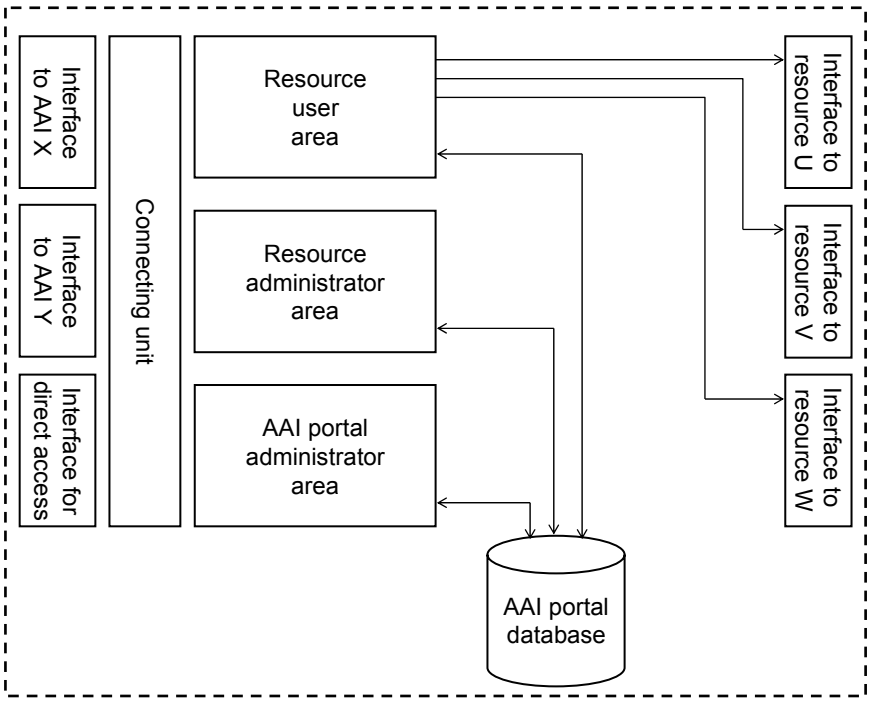

Figure 4.6: The functional parts of the AAI portal.

4.4.2 User Information Attribute Flow

Figure 4.7 shows the attribute flow from a home organization through the core AAI to the resource provider for a resource hosted by the AAI portal. User-specific attributes originate from the user's home organization or are provided by the user to the AAI portal. The AAI portal stores user data in its own database and, upon a successful subscription to a resource, stores the respective user data set into the resource's database or a database that is queried by the resource.

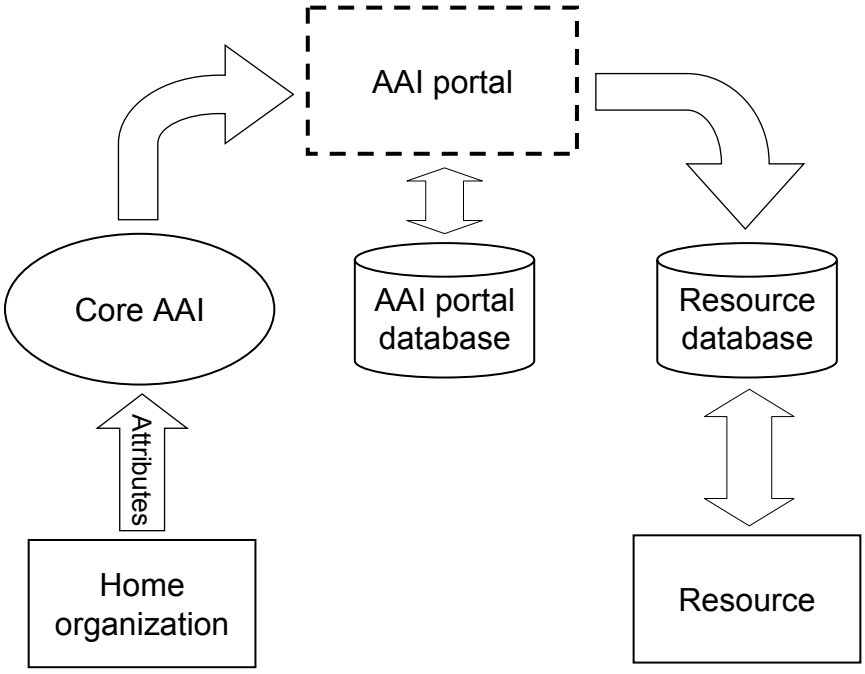

Figure 4.7: AAI portal architecture and user attribute flow.

Attributes flow from the home organizations through the core AAI to the resources, the reverse is not foreseen, although one could imagine it. A reverse attribute flow from resources to home organizations is for example needed for a financial accounting or for a reporting of student work results. Attributes could also be retrieved from the AAI portal by the home organizations by means of special users that poll the AAI portal for resources' data they are allowed to access.

Figure 4.8 is the AAI portal's configuration flow chart. An unknown person enters on the top left side and first authenticates at its home organization in an AAI registration process. The now authenticated person, called user Bob hereafter on, gets to the web interface for resource selection on the AAI portal. The web interface lists resources that are unsubscribed, subscribed, or pending for subscription. In a next step, the AAI portal checks for all needed data (authorization) depending on the resource's attribute acceptance policy. If there are missing attributes (in other words, attributes that are not delivered by the home organization), the user is directed to the AAI portal's web interface for collecting user data.

Missing user attributes collected on the AAI portal are clearly marked as non-reliable attributes. If the attributes satisfy the resource's attribute request policy, the AAI portal checks if the user is a new subscriber or has already subscribed to the resource. In the case of a new subscriber, the AAI portal verifies if there is a waiting list for the subscribed resource. Already subscribed resource users that have been granted access to the resource and new resource users accessing resources without waiting lists are forwarded to the resource. For new subscribers of resources without waiting list, user data is prepared for the resource and stored in the respective database. For resources with waiting list, subscribers get to the waiting list, where an administrator has to deblock them. Upon deblocking user data the AAI portal database is updated and the status of the deblocked resource changed on the resource list of the respective subscriber. Those subscribers log-in again later and follow the already described way for already subscribed users.

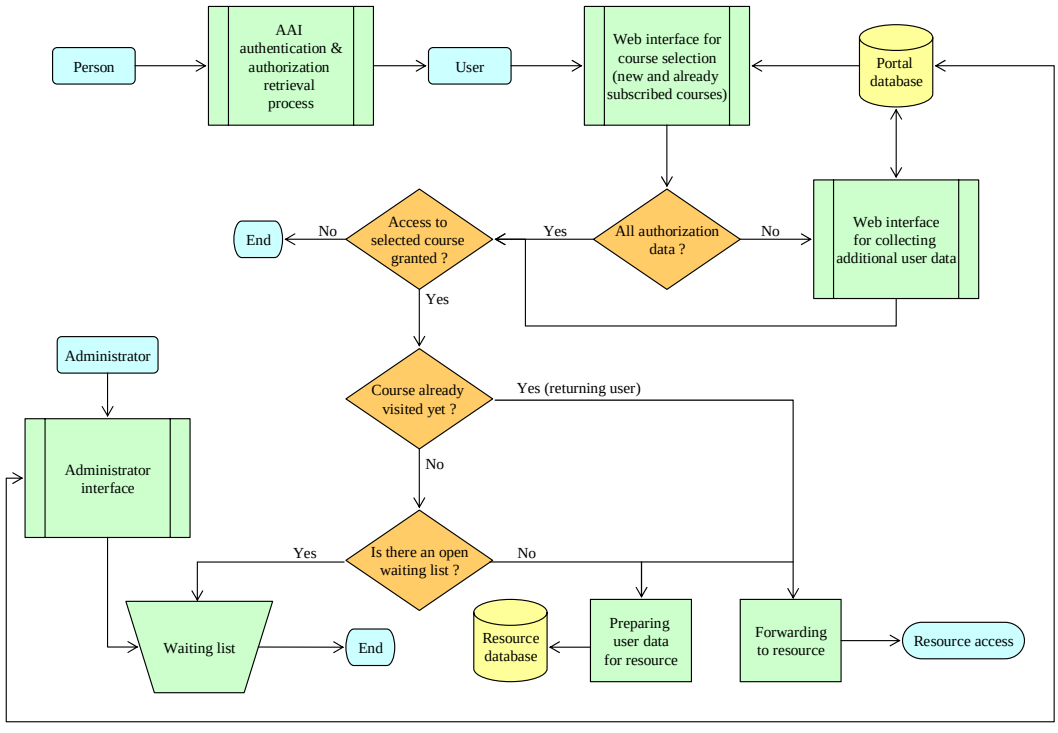

Figure 4.8: AAI portal configuration flow overview.

4.4.3 Interface to AAI (AAI Adaptors)

The AAI portal's interface to the AAI receives user attributes from the authentication and authorization infrastructure. Each type of AAI delivers attributes in a different way. Figure 4.9 shows the principle of the attribute flow towards the AAI portal.

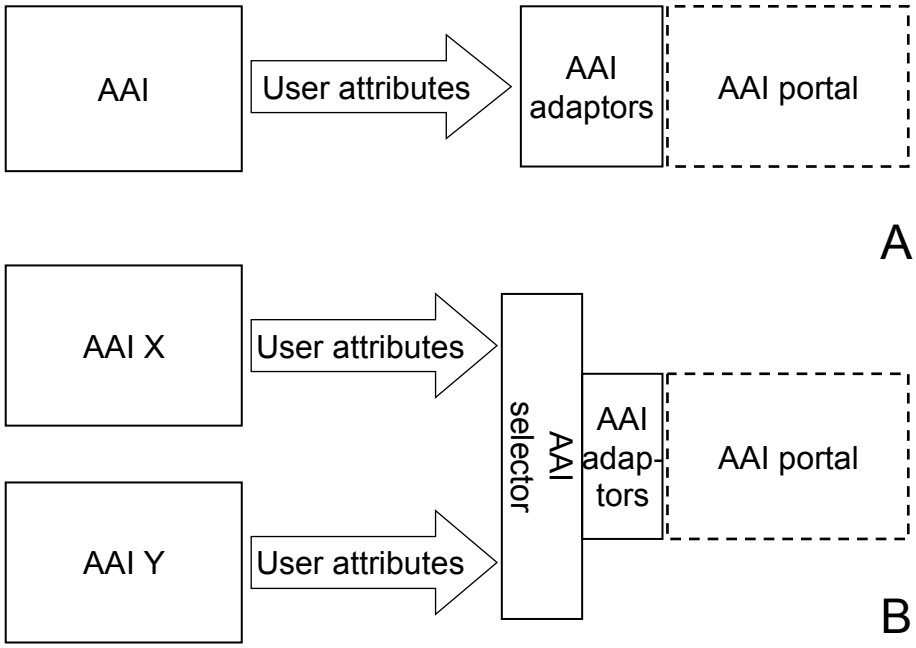

Figure 4.9: Interfaces to AAI.

In Figure 4.9 A only one AAI is connected to the AAI portal whereas in Figure 4.9 B two different AAI are connected. The AAI selector gives the users the possibility to choose their proper authentication and authorization infrastructure.

4.4.4 Interfaces to Resources (Resource Adaptors)

Introduction to Resource Adaptors

There is no general way to connect external resources to the AAI portal because each individual resource has its own authentication and authorization system and security properties.

In general, the AAI portal HTTP-redirects users to the resource together with a transmission of user attributes as shown in Figure 4.10. The interfaces to the resources are called resource adaptors. A proper redirection provides users with opaque information that can be decrypted only by the resource.

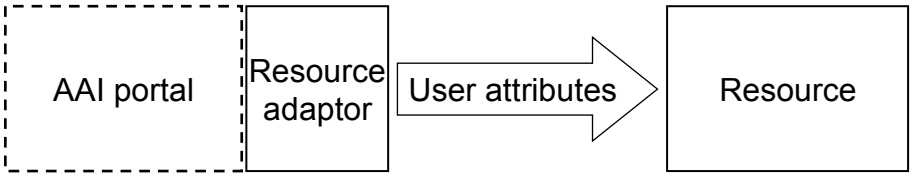

Figure 4.10: Interface to resource.

Suppose for each case, that user Bob accesses the AAI portal and chooses to access resource Vanilla. Bob is equipped with all the necessary user information attributes to access the resource Vanilla.

Scenario 1: Simple user redirection.

Resource Vanilla does not need any information about Bob, and therefore, Bob is only HTTP-redirected to Vanilla. This is the simplest type of a resource adaptor and only listed for the sake of completeness as it does not really need an AAI or the AAI portal.

Scenario 2: User redirection with cookie containing username and password.

Resource Vanilla wants to restrict access to AAI users but keep a username and password-based access procedure. Therefore, Bob is provided with an encrypted and time stamped cookie with a limited validation length that contains Bob's user identity (uid) and password. The password is generated by the AAI portal and not known by Bob to prevent a direct login to the resource. A direct login to the resource would take off admission control and accounting functions from the AAI portal. The portal has to store username and password into the resource's database before Bob is redirected to Vanilla. Vanilla checks whether the cookie is valid and the user exists in its database. This scenario is interesting for existing resources that cannot be re-programmed and/or must remain with username and password access control.

Scenario 3: User redirection with generation and display of username and password (not Single-Sign-On capable!).

The AAI portal generates a user for the resource Vanilla and first stores the username and password in the resource's database and secondly displays them to Bob in the form of an image, additionally to the recommended SSL or HTTPS transmission for security reasons (it is easier to scan and process traffic for passwords if it is in clear text). Bob is now HTTP-redirected to Vanilla and has to provide username and password. This scenario should only be used for one time logins as Bob could login directly to the resource if username and password remain valid after the first session.

Scenario 4: User redirection with opaque cookie.

The AAI portal stores Bob's information in its own database but in an area accessible by the resource or directly in the resource's database. Bob is provided with an opaque cookie containing a generated handle and redirected to Vanilla. Vanilla tracks back the information found in the form of the handle contained in the cookie and lets Bob access. The handle is a string without any useful information to third party i.e. it is not a password or a username. It is a string that allows the resource to link the accessing user with the database entry about Bob in this case.

Scenario 5: User redirection with a Message Authentication Code (MAC) protected ticket in a cookie or URL.

The AAI portal generates a MAC protected ticket (hashed user data concatenated with a shared secret key and then again hashed), which Vanilla can validate. This especially requires an exchange of the shared secret between portal and resource in advance.

A special case of scenario 5 is the generation of the ticket compatible to the Institutional Management System (IMS / IMS ticket) [Ims]. Institutional management system is a standard in the world of e-learning. The IMS Global Learning Consortium develops and promotes the adoption of open technical specifications for interoperable learning technology.

Scenario 6: User redirection with a Message Authentication Code (MAC) protected ticket in a cookie or URL plus additional attributes.

The AAI portal generates a ticket that is enhanced with additional attributes such as an additional username or handle. Bob accesses Vanilla and out of this resource another resource called Cinnamon that is operated under the same domain name to avoid the rejection of third party cookies by users' browsers. Cinnamon has an own database in which the AAI portal has stored Bob's username or it can track back the handle to Bob. By means of Bob's ticket, Bob can now access Cinnamon without manually providing any credentials.

A special case of scenario 6 is the generation if the ticket following the institutional management system standard (IMS ticket).

Scenario 7: User redirection with IMS API and Autosignon API.

The AAI portal uses the institutional management system API to check if Bob already exists in resource Vanilla. If not, the user Bob is generated in the resource's database. The AAI portal now HTTP-redirects Bob to the automatically sign-on API provided by Vanilla.

Scenario 8: User redirection to an AAI-enabled resource.

Resources that are enabled to an AAI (e.g. the same as the AAI portal is) are connected by a resource adaptor with the same functionalities as the AAI provides. This adaptor allows resource owners to connect their AAI-enabled resource to the AAI portal and profit from the user, resource and community management features it offers.

Integrated Resource Adaptors

Administrators see a list of built-in resource adaptors when they add new resources to the AAI portal. Each built-in resource adaptor can be chosen for the integration of new resources whenever needed. A resource adaptor has a unique name and displays additional information about its functionality. Further information is displayed about the integration procedure of the resource to which the resource adaptor fits.

One of the already implemented resource adaptors uses the IMS API and is used to redirect AAI portal users to a course home page hosted on a WebCT 3.8 to 4.1 server. The WebCT adaptor passes authentication data directly to the WebCT user database in order that the user is automatically signed on. When a user tries to access the WebCT course guarded by this adaptor, the adaptor first creates a

WebCT user for the given AAI user. It also subscribes the user to the course. This is carried out by invoking respective calls in the WebCT IMS API. Secondly, the resource adaptor redirects users to the course home page via the WebCT Autosignon API.

The already integrated resource adaptors at the time of writing this document are described on greater detail in Chapter 5.

How to Integrate Resource Adaptors

One goal of the AAI portal is to easily integrate new resources. Therefore, an implementation of the AAI portal must offer an application programming interface with at least the possibilities to:

- Read-out user data from the AAI portal database.
- Write user data into the AAI portal database.
- Read resource data from the AAI portal database.
- Write resource data into the AAI portal database.
- Display a help text to a specific resource adaptor
- Create a resource specific database on the AAI portal which it can access

4.4.5 Interface to Community Management Features (CMF Adaptors)

Community Management

The AAI portal includes many basic community management features for the management of users and hosted resources. Basic community management features are absolutely necessary for the needed operations on users and resources.

However the AAI portal can do much more than just manage resources and users; it can offer enhanced features such as discussion boards or chat. The most important point is that those resources are personalized and fully manageable by resource administrators.

Figure 4.11 shows the AAI portal, here connected to Shibboleth as authentication and authorization infrastructure together with a document repository as connected resource. This figure also shows that community management features follow the same plug-in concept as resource adaptors do. The resource owner would like to add a forum to his resource without changing the resource platform that does not foresee a forum. He asks the AAI portal administrator to add a forum to the portal and the forum is ready to use.

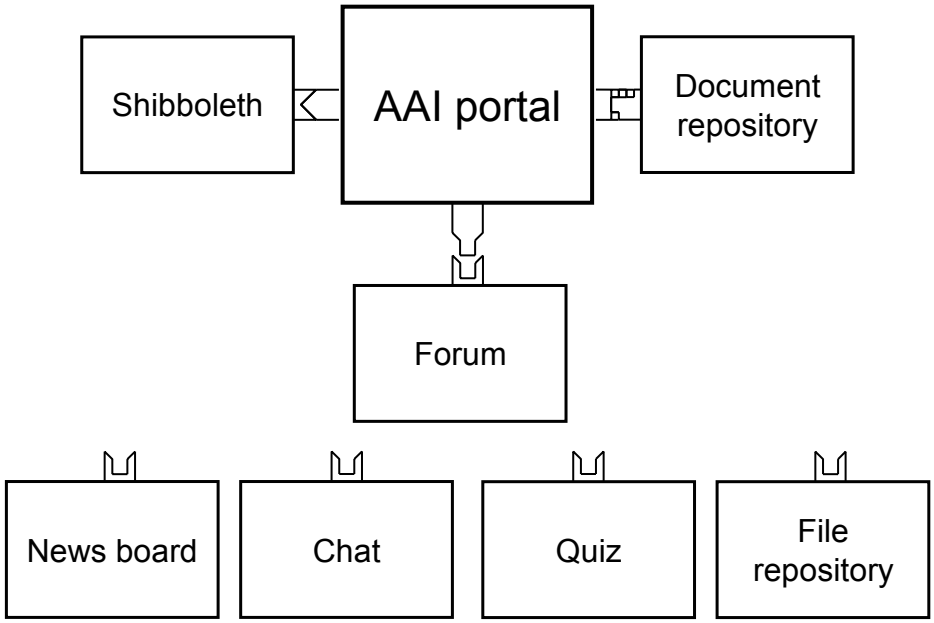

Figure 4.11: Several authorization levels in resources.

A set of useful asynchronous and synchronous features are explained below. They can for example fulfill tasks of features missed in hosted resources or even replace them.

Synchronous Community Management Features

Synchronous tools give tutors and students the possibility to instantly interact together. The advantage lies in the spontaneous nature of these tools that encourage direct communication and prevent users from exchanging only well prepared material like it happens in asynchronous tools such as discussion boards.

Video/Audio Conferencing allows close to reality communication between tutors and course subscribers. Main obstacles of this tool are the bandwidth and hardware requirements that most users of today's Internet do not fulfil. Conferencing sessions must be planned in advance and announced by other services. Big advantages are that discussions can be close to classroom reality and that students can communicate between themselves.

Chat Tutors have the possibility to create internet relay chat rooms and act as moderators in their rooms. Students have the possibility to visit the respective rooms of their course. Unmoderated chat rooms can be opened by students but only students subscribed to the same course are able to join. Internet relay chat is synchronous tool that can be used simultaneously to other tasks and combines variable response times with an extreme low bandwidth consumption.

Instant Messaging Students see when their course colleagues are online and can send messages that are instantly delivered. Instant messaging is designed for bilateral message exchange and is not useful for multi user chats. Tutors and students see who is online without connecting to a service such as Internet relay chat.

White Board This tool is normally used together with the chat or the conferencing tool. It allows tutors to explain and draw on a board at the same time. All students of the course can assist simultaneously. If a student wants to interact on the board, the tutor gives the permission to do so and the student can draw on the same board. Students can also use the white board tool for small student groups.

Asynchronous Community Management Features

Asynchronous tools give tutors and course subscribers the possibility to communicate in a time shifted manner and thereby to carefully prepare their statements as involved persons do not have to be simultaneously online.

Discussion Forum Tutors have the possibility to open a subsection of a forum with the name of their resource and to moderate the contributions. Students get automatically access to the forum related to the course they have subscribed.

Calendar Each course tutor can add a calendar to show dates of conferences, deadlines and other important milestones. It is not planned to give write access to students for the calendar function.

Pointers to Webpages Each course tutor can add a page with web links which point to resources related to the course such as web pages, e-books and other downloadable material. Students can add pointers that get on a waiting list and can be released by the tutor.

Group Announcements The tutor should have the possibility to divide big classes into groups. The division of big groups to small groups makes classes easier to manage. Group announcements affect information tools such as email and short message service.

FAQ In most courses students encounter similar problems. Tutors get the possibility to publish frequently asked questions and answers.

Meeting Tools In many courses it is necessary that students work in groups. The meeting tool gives students the possibility to meet others anonymously and to reveal their identity only if desired. Tutors enable or disable this tool for the students of their course.

Email It is not planed to provide students with email addresses. Students already provide their personal email to the AAI portal. As described for the basic course management features, email can be used for information purposes. Tutors can announce important postings in the discussion board or chat meetings.

Short Message Service This tool does about the same as the email tool. The difference is that mobile phones nowadays are broadly spread and it is very easy to reach users very fast.

Already Integrated Community Management Features

Administrators see a list of built-in community management features on the ‘My Resources’ page. Each built-in community management feature can be activated, reactivated or deactivated independently for each resource. Each community management feature has a unique name and displays additional information about its functionality.

How to Integrate Community Management Features

One goal of the AAI portal is to easily integrate new community management features. An implementation of the AAI portal must offer an application programming interface with at least the possibilities to:

- Read user data from the AAI portal database.
- Write user data into the AAI portal database.
- Read resource data from the AAI portal database.
- Write resource data into the AAI portal database.

4.4.6 Authorization Issues with Resources

The AAI portal is an instance that is interlaced between the core AAI and resources, as outlined in Figure 3.1. The AAI portal needs to know which database attributes a resource requires. This especially affects resources not available in open source code because the necessary information is rarely freely available.

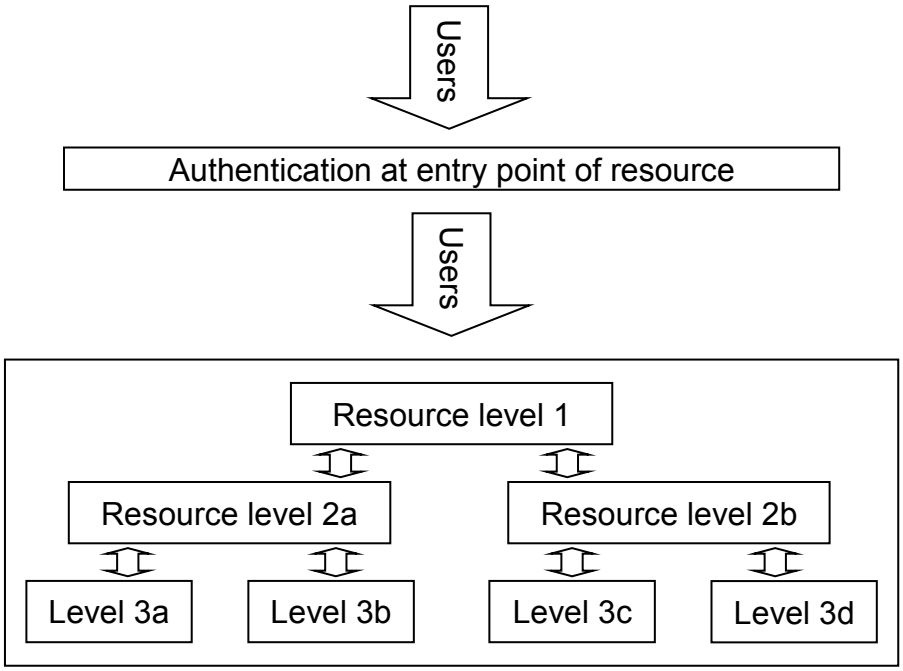

Figure 4.12: Several authorization levels in resources.

User Bob for example is authenticated by its home organization and provides all the necessary attributes to enter a resource on resource level 1 (Figure 4.12). At this level Bob can select different sub areas with higher restriction levels. In an AAI-enabled resource, this would be no problem at all, as only areas could be chosen, where Bob is authorized to.

As already described above, the AAI portal typically creates one type of users that get access to resources where internal user management is non-disclosed and therefore users cannot be equipped with role attributes.

Such issues could be solved with a workaround. The AAI portal generates one type of user that gets a predefined access to the resource. If such a user should get more authorization privileges, a resource administrator has to upgrade him/her manually in the resource itself.

4.5 System Design

4.5.1 Web Application with a Database Backend

The AAI portal potentially offers access to resources for users that originate from different language regions. The AAI portal should adapt user interfaces to the language attribute users provide via AAI and otherwise offer the possibility to switch manually between languages. We propose to use the I18N internationalization scheme [I18N].

Figure 4.13 show a system diagram of the AAI portal. The AAI portal is a web application with web interfaces for users and administrators and a database.

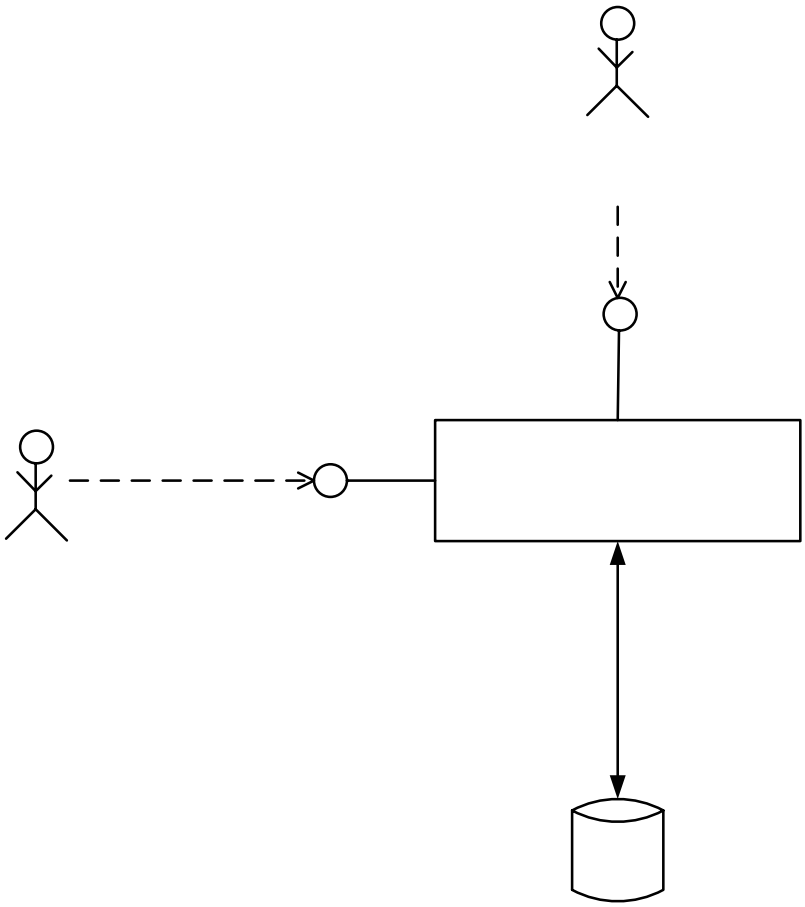

Figure 4.13: System diagram.

4.5.2 Web Front-End Design

Figure 4.14 shows the proposed structure for the AAI portal's web front end. It is split up in three parts:

Header area: We proposed to display the user's role as well as the user's name and given name as required in the header area. This area can host logos and titles.

Menu area: The menu area hosts the menus that guide users through the AAI portal.

Content area: The content area displays the in the menu area selected content.

The AAI portal must allow an adaptation to an existing uniform design.

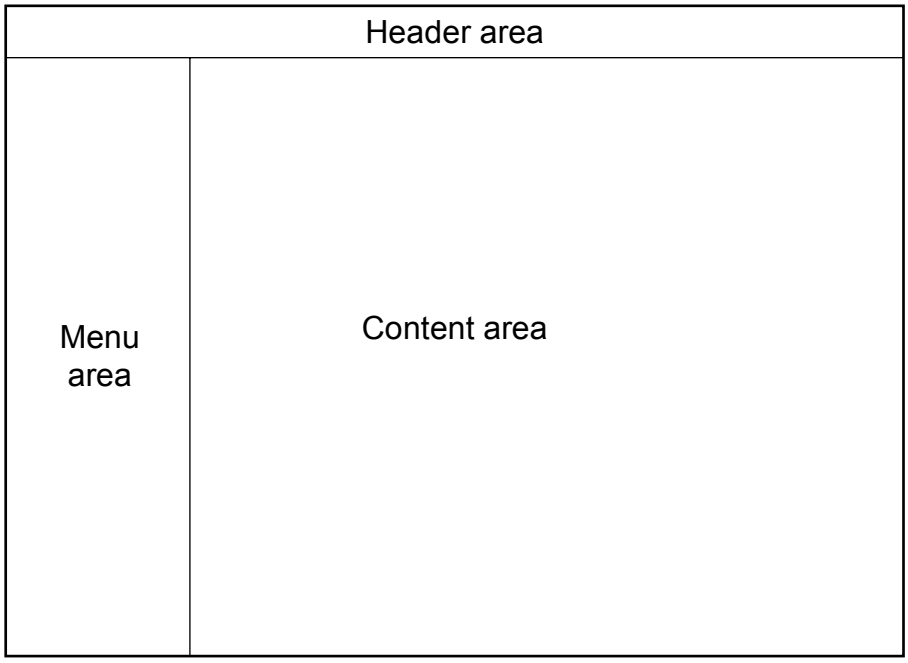

Figure 4.14: Web front-end structure.

4.5.3 Database Design

The AAI portal's database consists of eight SQL tables:

User: The user table holds the unique user id (User_uid) which must be provided by the AAI infrastructure. Each entry must have a privilege level from the UserPrivilege table.

UserPrivilege: The UserPrivilege table has three standard entries, a normal user privilege (student), a resource administrator privilege and a portal administration privilege. It is being used to distinguish normal users from administrators.

Attribute: This table holds the attribute value for one specific attribute. Each user can have any number of attributes and each attribute must have exactly one AttributeType.

AttributeType: Each attribute value must have a type. These types are stored in this table and it is initialized with the standard AAI attributes but it can also be extended by custom, string encoded attribute types.

Resource: The resource table stores the properties for each resource. Each resource entry must have exactly one user with at least resource administrator privilege.

Policy: This table defines what AttributeTypes are needed for each resource; technically it is a relation between the resource and the AttributeType table.

ResourceAdapterParam: This table holds configuration parameters for resource adaptors.

Subscription: The subscription table connects a user table entry to a resource. Like the policy table it is a relation between two other tables, user and resource. It holds some additionally data like access control information and so on.

Figure 4.15 shows a diagram depicting the AAI portal's database schema.

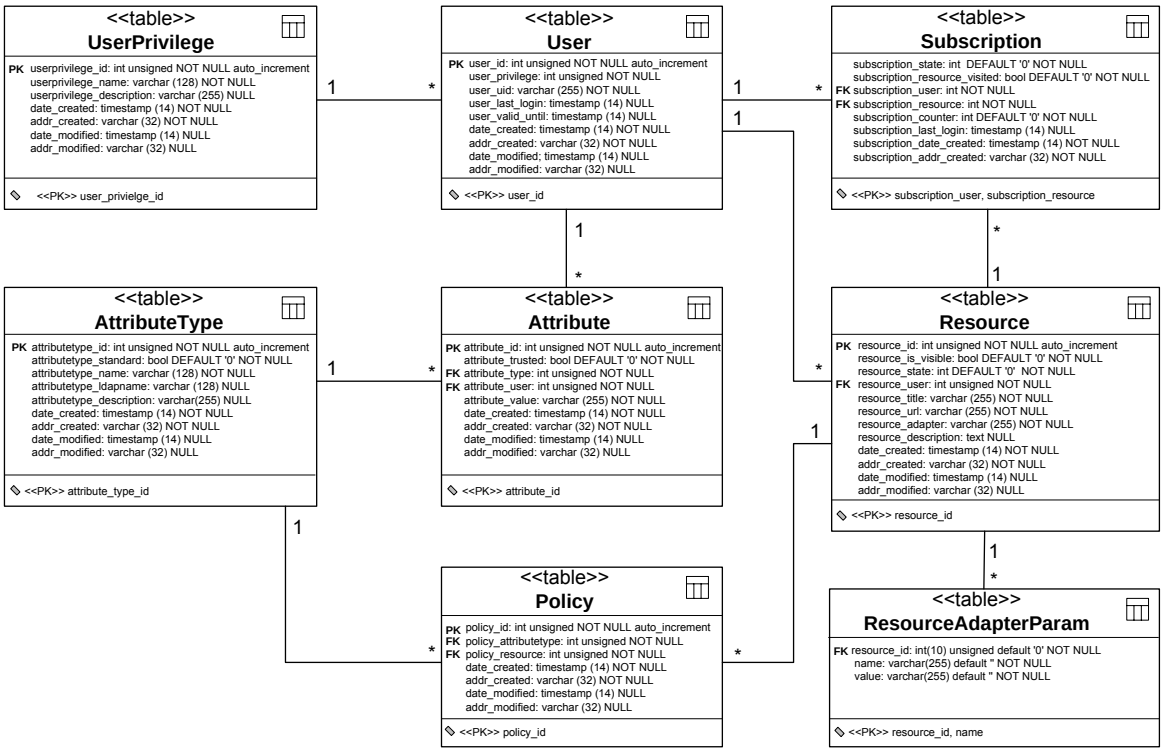

Figure 4.15: AAI portal database schema.

5 AAI Portal Prototype Implementation

The prototype implementation is written in PHP and uses MySQL as database engine. On the AAI portal project site (http://www.switch.ch/aai), the current prototype implementation is publicly available, also for a live demonstration.

5.1 System Context

5.1.1 Authentication and Authorization Infrastructure Adaptors

The prototype implementation of the AAI portal is not tightly coupled to a particular AAI, although the current implementation ships with an adaptor to the Swiss Shibboleth implementation.

A particular AAI must follow two rules when being integrated with the AAI portal. First, it must use a HTTP-redirect to send a user to the AAI portal. The redirection target consists of an entry point in the AAI portal specific to this particular AAI.

Secondly, the request for accessing the AAI portal must be accompanied by a set of attributes in the form of name/value pairs. The AAI portal prescribes that at least an attribute with a unique user identifier must be included in a request. The AAI specific entry point to the AAI portal may accept additional attributes which it may map, if present in a specific request, to a set of attributes used within the AAI portal.

Adaptor to Shibboleth

In order to interface the AAI portal with Shibboleth, the Shibboleth target site software needs to be installed on the web server where the AAI portal is deployed. In addition, the web application container which hosts the AAI portal must be integrated with the Shibboleth target site software according to the Shibboleth installation guide.

The AAI portal provides two entry points, one for the user part and another one for the administration part. Both entry points consist of a dynamic web page which is hosted by a web application container (Apache in our case). The two entry points are summarized in Table 5.1:

Entry point for ...	Related file in AAI portal distribution	Canonical request URL
... user part	`$AAI_PORTAL_HOME/web/user/aai/index.php`	`http://your.host/aai-portal/user/aai/index.php`
... administrator part	`$AAI_PORTAL_HOME/web/admin/aai/index.php`	`http://your.host/aai-portal/admin/aai/index.php`

Table 5.1: Entry points for user and administrator part.

The web application container has to be configured such that users must authenticate with Shibboleth when they access one of these entry points. Usually, web application containers offer multiple configuration options to restrict access to a specific page. If Apache is used as web application container, the respective configuration entries can be entered in the common Apache configuration file (httpd.conf) or in access control file written to the directory which has to be protected (.htaccess). AAI portal ships with sample .htaccess file in the directories containing the two entry points which look similar to the example in Figure 5.1:

```
#
# .htaccess - sample access control file for AAI portal
#             entry point

# enable Shibboleth authentication
#
AuthType shibboleth
ShibExportAssertion On
require valid-user
#
```

Figure 5.1: AAI portals' sample .htaccess file.

Users authenticated in the Swiss Shibboleth implementation have a unique ID provided in the attribute swissEduPersonUniqueID. A set of additional 24 attributes is defined in SWITCH, six of which (including swissEDUPersonUniqueID) are required. The two AAI portal entry points receive specific attribute values in HTTP headers reserved for this purpose. They map these attributes to attributes used within AAI portal, and cache them in the AAI portal database.

5.1.2 Resource Adaptors

Plug-in Concept

In the AAI portal, a resource adaptor is a piece of PHP code which is responsible for redirecting a user from the AAI portal to a specific external resource. The AAI portal design allows any number of resource adaptors to be deployed with an AAI portal installation, because resource adaptors are implemented such that they conform to a kind of plug-in specification. This ensures that resource adaptors can be disabled, replaced, and that new custom adaptors can be developed and installed easily.

The developer of a resource adaptor has to implement at least an adaptor PHP page and to provide an adaptor descriptor.

Resource Adaptor Page

The adaptor PHP page is an isolated PHP page which must be installed in the directory aai-portal/adapters. The page must accept a single resource ID in the request parameter rid. It must also ensure that the current session is a valid AAI portal session. Given the current user and the resource ID, the adaptor PHP page must prepare access to the remote resource. Finally, it must redirect the user to the target URL. The execution of an adaptor PHP page therefore usually finishes with emitting a HTTP redirect header. Unless an error occurs, now HTML markup is usually sent to the client.

Resource Adaptor Descriptor

The adaptor descriptor is a set of configuration parameters. Each resource adaptor has a unique ID and an array of name/value pairs.

The following properties can be configured for a resource adaptor:

- `display-name` (optional):
  The adaptor's display name to be used in UI widgets like drop-down lists, and the like. If missing, the adaptor ID is used instead.

- `adaptor-file` (mandatory):
  The relative URL to the Resource Adaptor implementation. The URL is relative to PORTAL_BASE_URL (see aai-portal.config)

- `help-text` (optional)
  An optional help text for the resource adaptor.

- `parameters` (optional):
  An optional list of resource adaptor parameters

Each resource adaptor may specify a list of resource adaptor parameters (in the property 'parameters', see above). A resource parameter is an information item the resource owner must enter when he configures a resource adaptor for a specific resource. Consider for instance a resource adaptor which can redirect AAI portal users to a resource protected with a resource specific password. Then 'password' would be a resource adaptor parameter the resource owner has to fill in when he assigns this resource adaptor to a specific resource.

Resource adaptor parameters are an array of key/value pairs. This is the parameter's unique name (i.e. 'password') and the value is another array of key/value pairs. For each parameter the following key/value pairs can be specified:

- `display-name` (optional):
  The display name for the resource adaptor parameter to be used in UI widgets and input forms. If missing, the parameter ID is used instead.

- `description` (optional):
  A short description of the parameter.

Resource adaptor descriptors are stored in the global configuration file config/adapters.config. This configuration file declares an array of resource adaptor descriptors using standard PHP syntax; actually, the configuration file will be loaded using a call to require_once(...).

Therefore any kind of PHP literals, i.e. numeric literals like 999 or 123.45, string literals like "test" or 'test', etc. can be used.

Key/value pairs have the form key => value.

All of the many forms of writing comments in PHP can be used as well.

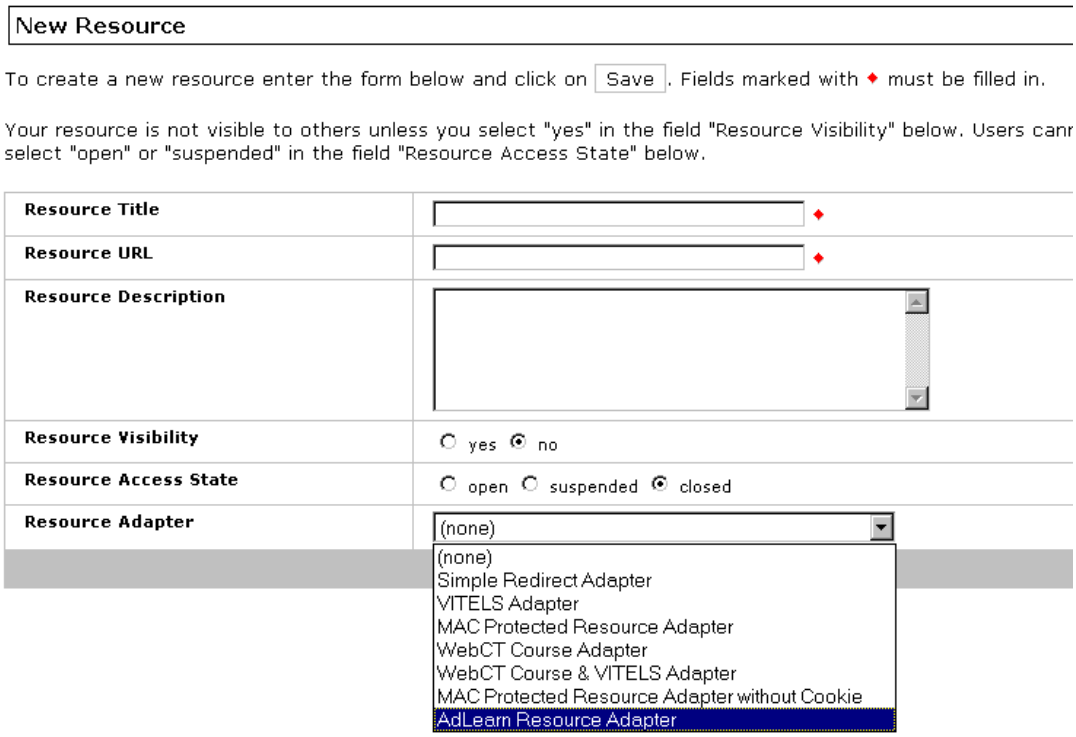

Figure 5.2: New resource page for administrators.

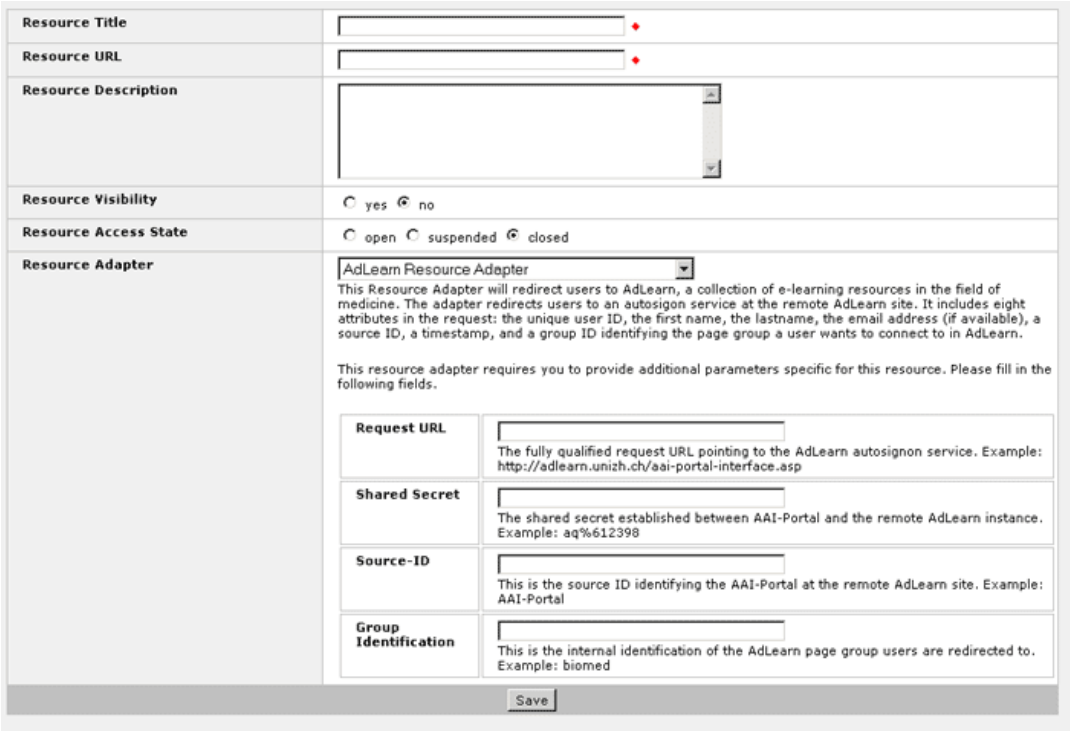

Figure 5.3: New resource page with ResourceDataForm.

Resource Adaptors API Functions:

To make it easier to create a new resource adaptor, there is an API with the following available functions (to use them, one has to include the file: include/Adapting.class.inc) into the resource adaptor. See Figure 5.2 and 5.3 for a visualization of the Help-Text and the resource DataForm.

- array getUserAttributes(string $uid):
  Gets all userattributes of a user (based on his uid), even the resource specific ones. Returns an array with all Attributenames and –values.
- void storeTransportedData(string $uid):
  Before redirecting the user to the resource, all data to be transported and an additional timestamp are stored for further use in the table ResourceAccessLog of the portal - db
- array listAllAdaptors():
  Returns an array with all available resource adaptors
- array listAdaptorsHelp():
  Returns an array with all help-files to resource adaptors
- array listMissingHelp():
  Returns an array with all resource adaptors which have no HelpText yet
- void redirectUser(string $url, Boolean $secure):
  Redirects the user to the resource's $url. If $secure is true, the user is HTTPS-redirected
- string createUsername(integer $length):
  Returns a username with length $lenght. It can be used for resources without connection to the AAI (portal) database and an own authentication system
- string createPassword(integer $length):
  Returns a password with length $lenght. It can be used for resources without connection to the AAI (portal) database and an own authentication system
- void createDB (string $host, string $user, string $pwd, string $dbName, string $tablename, array $attributes):
  Creates a DB on the AAI portal with standard admin rights for the AAI portal and select rights for $user.
- string createMAChash($key, $data):
  Creates a MAC using the shared $key and some $data.

Built-In Resource-Adaptors

Simple HTTP-redirection

This resource adaptor simply redirects the user to the target URL.

HMAC-based Authentication

This resource adaptor redirects the user to the target site and generates a cookie which authenticates the user at the target site.

The cookie includes the AAI User ID and a HMAC generated according to RFC 2104 (HMAC: Keyed-Hashing for Message Authentication).

The following configuration parameters can be set:

- `shared-key`:
  The shared secret established between the AAI portal and the resource.
- `cookie-name` (optional):
  The cookie's name. If empty, defaults to 'AAIPortalAuthCookie'.
- `cookie-expires` (optional):
  The number of seconds after which the cookie expires. If empty, defaults to 7200s.
- `cookie-path (optional)`:
  The cookie path specifying the subset of URLs in a domain for which the cookie is valid. If empty, defaults to /.

- `cookie-domain (optional)`:
  The cookie domain. If empty, defaults to the server's host name which generated the cookie response.

WebCT Adaptor

This resource adaptor redirects AAI portal users to a course home page hosted on a WebCT 4.0 server. The WebCT adaptor is a fairly complicated adaptor, because it both provisions the WebCT user database and does an automatic sign on for the user.

When a user tries to access the WebCT course guarded by this adaptor, the adaptor first creates a WebCT user for the given AAI-User. It also enlists the user in the course. This is carried out by invoking respective calls in the WebCT IMS API.

Secondly, the resource adaptor redirects users to the course home page via the WebCT Autosignon API.

The following configuration parameters can be set:

- `WEBCT_IMS_API_REQUEST_URL`:
  The fully qualified URL to the IMS API script on the remote WebCT instance. Example: https://host.domain.com:8900/webct/systemIntegrationApi.dowebct.
- `WEBCT_IMS_SHARED_SECRET`:
  The shared secret established between AAI portal and the remote WebCT instance.
- `WEBCT_IMS_SOURCE_SYSTEM_IDENTIFIER`:
  The IMS source identifier to be used when a user is created in the remote WebCT instance.
- `WEBCT_AS_REQUEST_URL`:
  The fully qualified URL to the Autosignon API script on the remote WebCT instance. Example: https://host.domain.com:8900/webct/public/autosignon.
- `WEBCT_AS_SHARED_SECRET`:
  The shared secret established between AAI-Portal and the remote WebCT instance for autosignon.
- `course-id`:
  The WebCT course ID.

VITELS Adaptor

This resource adaptor redirects AAI portal users to a VITELS page such as the scheduling system or a module portal. VITELS now accepts an authentication cookie which is written by this adaptor. Users that visit VITELS for the first time will be added to the LDAP directory.

The following configuration parameters can be set:

- `cookie-based (optional):`
  If set to '1' (one) the adaptor uses a cookie-based authentication for VITELS. If empty or set to something else it is assumed that VITELS is already AAI enabled and therefore no cookie has to be written.
- `cookie-path (optional)`:
  The cookie path specifying the subset of URLs in a domain for which the cookie is valid. If empty, defaults to /.
- `cookie-domain (optional):`
  The cookie domain. If empty, defaults to the server's host name which generated the cookie response.

- `cookie-expires (optional):`
  The number of seconds after which the cookie expires. If empty, the cookie will only be stored in the memory of the client machine (not on the hard disk drive). It will be valid until the browser is closed.
- `vitels-ldap-host (optional):`
  Hostname or IP of the VITELS LDAP server. If empty, a localhost connection will be made.
- `vitels-ldap-port (optional):`
  Port of the VITELS LDAP server. If empty, the default LDAP port (389) will be used.
- `vitels-bind-dn (optional):`
  The distinguish name (dn) of the account which will be used for the LDAP binding. If empty, 'uid=admin,ou=Staff,o=VITELS,c=CH' will be used. Please note that this account needs permission to add/modify user entries.
- `vitels-bind-pw (optional):`
  Password for the distinguish name binding. Optional. If empty, the password from the variable definition in the adapter script will be used.

VITELS and WebCT Adaptor

A combination of the before described WebCT and VITELS adaptors is integrated under the name of WebCT and VITELS adaptor. The combination of these two resource adaptors allows logging in students to the VITELS WebCT platform and directly connecting out from there to VITELS scheduling system or laboratory portals.

Technically, all the configuration parameters of both adaptors are available.

AD Learn Adaptor

This adaptor redirects the user to another Access Control System (ACS) called Securisite. This ACS is used in two projects, AD Learn and the Bio-Med portal.

The adaptor transmits needed authentication parameters, HMAC-authenticated according to RFC 2014.

The configuration parameters are:

- Request URL:
  The URL identifying the “entry point” to the target system.
- Shared secret:
  The secret phrase shared between the AAI portal and the target system.
- Source ID:
  An identifier for the AAI-Portal instance to the target system.
- Group Identification:
  The group code tells the ACS Securisite which authorization rights the user gets granted. These rights define conditions of access to a subset of resources and services.

As a principle, a user tries to access a Securisite-protected resource via the adaptor. The ACS checks if the user already has got an account:

If the user has got an account, the system proceeds a transparent login and redirects the user to the resource. The system also manages the cookie, session and authentication aspects.

If the user does not yet have an account, the system automatically subscribes the new user and grants him the required access rights up on deblocking by the respective resource administrator.

5.1.3 Community Management Features Adaptors

Adaptor Parameters

These parameters are not yet implemented.

5.2 Implementation Technologies

The AAI portal is implemented as dynamic web application.

5.2.1 Web Application Container

The AAI portal is implemented in PHP. It requires at least PHP 4.3.2 because it relies on the fsockopen() method which understands the SSL URL schema, a feature, which was introduced in PHP 4.3.2. In addition, the following modules have to be compiled into PHP (we give a short canonical name for the module and a reason, why this module is required):

- mhash - for generating checksums (MD5 or SHA-1 based).
- mysql - for accessing the AAI portal database.
- openssl - for accessing resources protected by SSL. This module is only necessary if the resource adaptor for WebCT courses is installed and used.
- session - session handling.

If the AAI portal is deployed under Apache, Apache 1.3.27 or 1.3.28 is required. Apache 2 is not recommended, mainly because Apache 2 is not yet fully supported by Shibboleth. As soon as Shibboleth supports Apache 2, the AAI portal is likely to support it as well, since there is nothing in the implementation of the AAI portal which would make it obviously incompatible with Apache 2.

The AAI portal is best installed on a web server running Linux. It has been installed and tested on Linux RedHat 7.2 and it is likely to run without modifications under other UNIX dialects. The AAI portal has not yet been installed on a Windows platform running another web application container than Apache (for instance Internet Information Server).

5.2.2 Database

The AAI portal uses MySQL 3.23.x as database engine (see http://www.mysql.com).

The AAI portal database should be setup in two steps. First, a database and a set of database users must be created. The individual steps are described in the AAI portal installation guide. Alternatively, the distribution includes SQL scripts which create a database based on standard settings and which can be executed in a MySQL shell. Secondly, the AAI portal schema must be created and initialized. Here again, the AAI portal distribution includes a SQL script which can be executed in a MySQL shell.

5.3 Web Front-End

Users access the AAI portal through one of two web interfaces, the user part or the administrator part respectively. Only a standard web browser is required to access the AAI portal because both the user interfaces are based on standard web technologies such as HTML 4.0, CSS, and JavaScript.

The design of the AAI portal user interfaces can be customized if the AAI portal is to be installed in an organization which requires web applications to conform to specific web design guidelines.

The AAI portal's customization concept is twofold. First, the AAI portal is shipped with two include files for a common page header and a common page footer. By including these HTML fragments at the beginning and at the end of every page, a common visual design of the AAI portal’s user interfaces is ensured. Both the page header and the page footer can be replaced in a deployed AAI portal instance. Table 5.2 summarizes the files involved:

File	Description
$AAI_PORTAL_HOME/include/header.inc	Page header for the user part of the AAI portal
$AAI_PORTAL_HOME/include/footer.inc	Page footer for the administrator part of the AAI portal
$AAI_PORTAL_HOME/include/adminHeader.inc	Page header for the admin part of the AAI portal
$AAI_PORTAL_HOME/include/adminFooter.inc	Page footer for the admin part of the AAI portal

Table 5.2: Files used for the layout.

Secondly, the AAI portal strictly separates visual design elements from structural design elements. The AAI portal generates HTML markup tags like <h1> for a heading on level 1 or <table> for a table, but it never generates HTML markup like <b> which visualizes a text in bold face or <font> which changes the font a text fragment is rendered in. Instead, all visual design elements are defined in a cascading stylesheet. As an example, Figure 5.4 shows a CSS fragment that defines how error messages look like in the AAI portal:

```
/* style for an error message in the AAI portal */
span.error-message {
  color:            red;
  background-color:  rgb(254,205,208);
}
```

Figure 5.4: Error message style.

The AAI portal ships with two independent visual designs. At installation time, one of these designs can be selected, or a third design can be developed if necessary. Visual design elements are provided in $AAI_PORTAL_HOME/web/lookandfeel. In the current distribution, this directory has one subdirectory styles, in which include files and style sheets for two visual designs are included.

5.4 Graphical User Interfaces and Workflows

Many flows and functions of the AAI portal are better understood with a few pictures. The following screen shots of graphical User interfaces (GUI) are design studies for the AAI portal's implementation. They are not meant to be complete in their functions nor in their design.

At the top of all GUI the surname and given name are shown.

Figure 5.5 shows the entry page for a user.

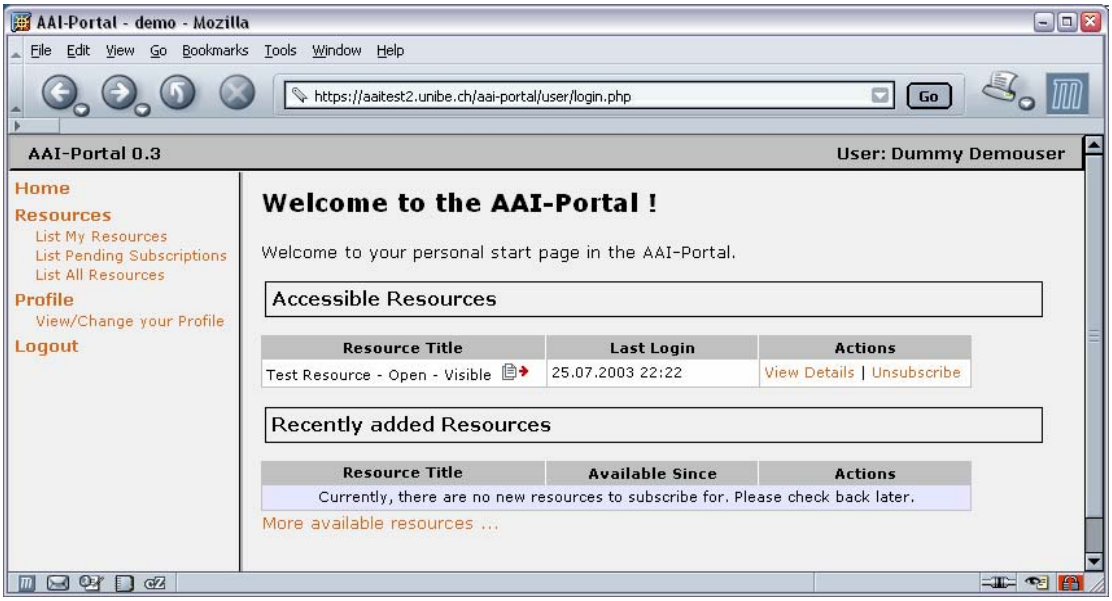

Figure 5.5: Entry page for users.

Figure 5.6 shows the entry page for a Resource Administrator. This resource administrator's entry page looks so empty because he does not own resources.

Figure 5.6: Entry page for resource administrators.

Figure 5.7 show the entry page for an AAI portal administrator. As expected, the AAI portal has much more selections due to his higher access rights than resource administrators or users.

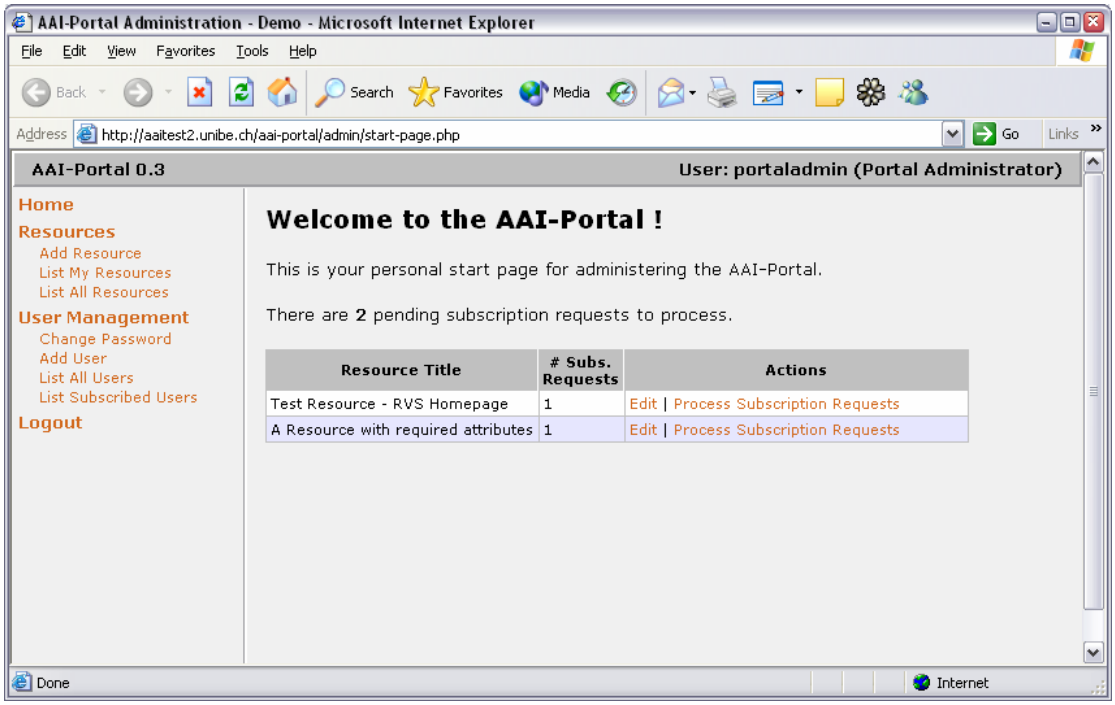

Figure 5.7: Entry page for the AAI portal administrator.

Figure 5.8 shows the 'My Resources' page, Figure 5.9 the 'Pending Subscriptions' and Figure 5.10 'All Resources' for users. There are three types of resources indicated:

All Resources: All resources hosted by the AAI portal the users can apply for.

My Resources: All resources the student is subscribed to.

Pending subscriptions: All resources that are on a waiting list

Users can subscribe to new resources or go directly to already subscribed resources. A button next to each resource displays information provided by the resource administrator describing the resource.

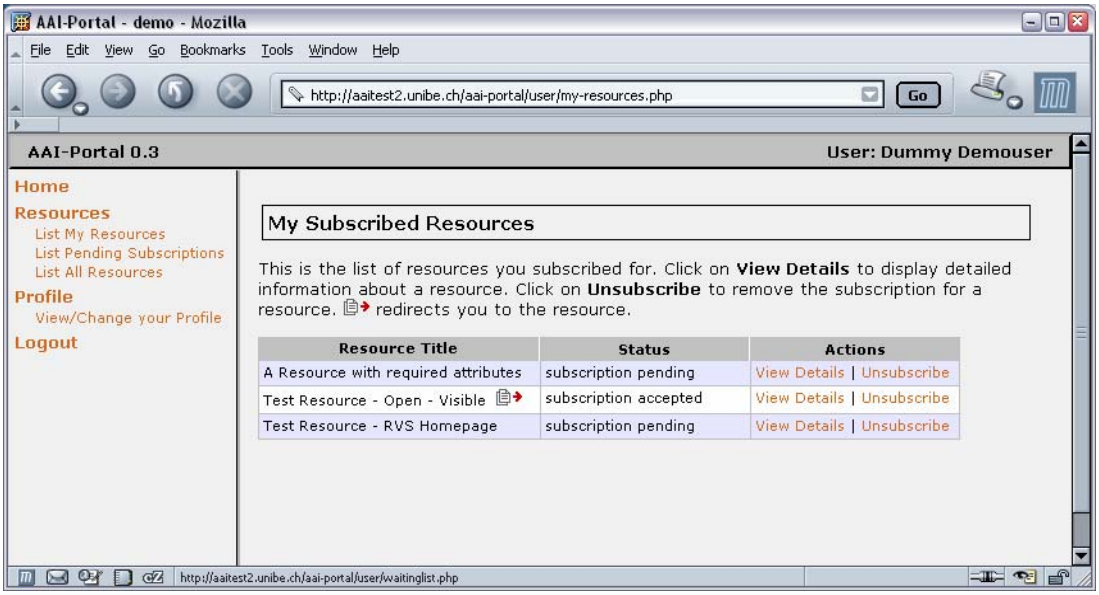

Figure 5.8: List of subscribed resources for users.

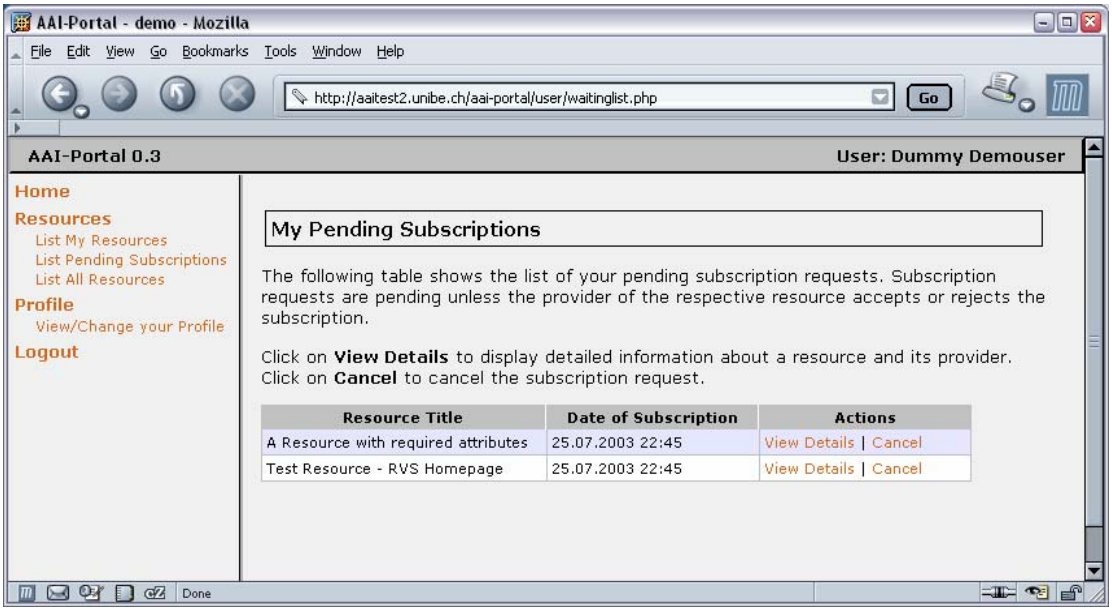

Figure 5.9: List of pending subscriptions for users.

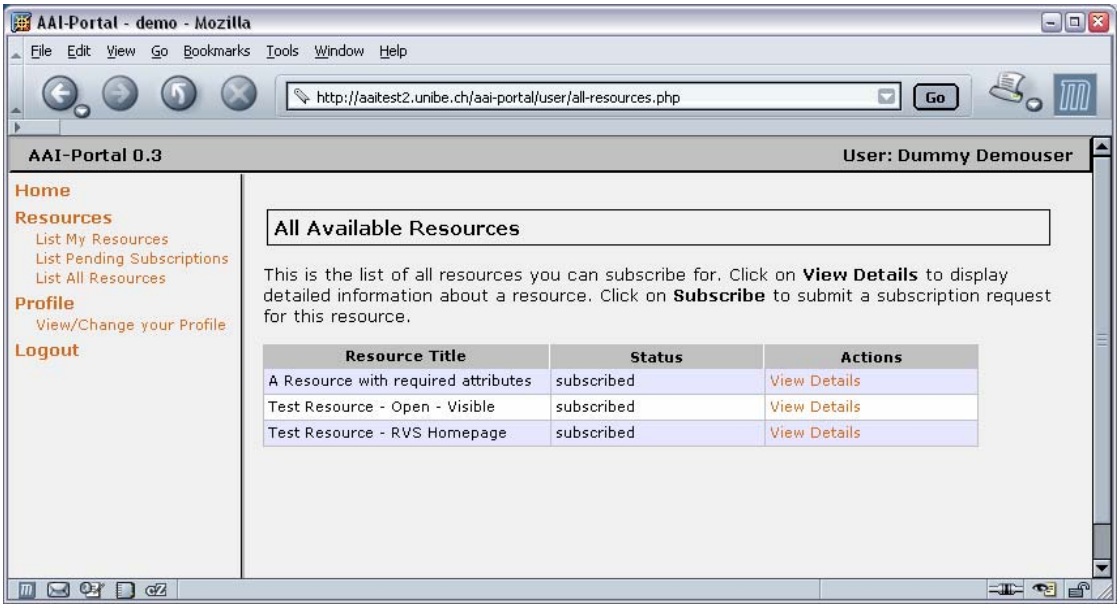

Figure 5.10: List of all resources for users.

If users need or want to change their already stored data or if they subscribe to a new resource that requires more attributes than the home organization delivered via AAI, users get to the page shown in Figure 5.11. Data originating from a home organization cannot be changed, neither by users nor by administrators. Only the home organization can change it in their local database.

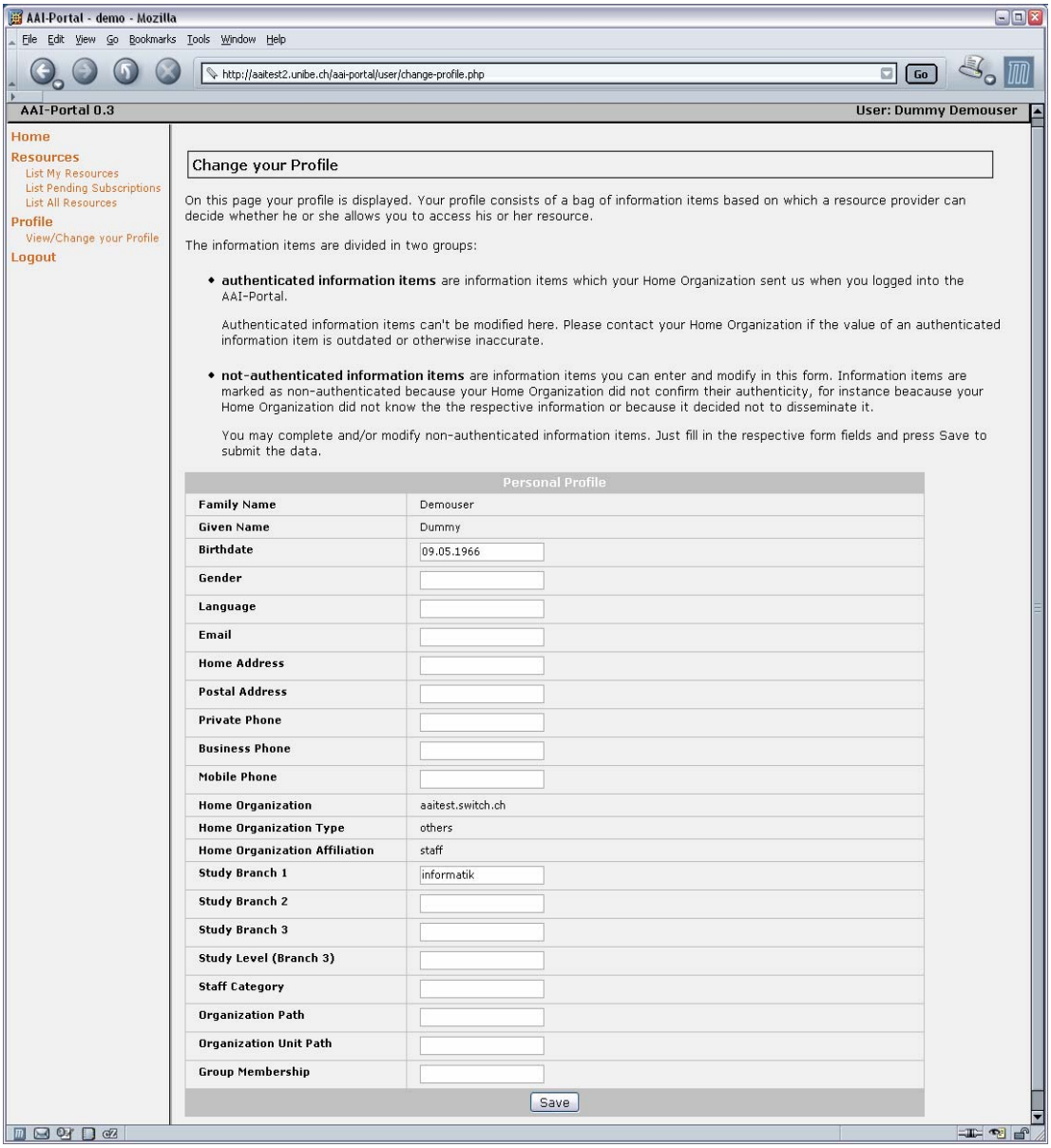

Figure 5.11: User profile.

Figure 5.12 shows the new resource page that is accessible by resource administrators and AAI portal administrators. Administrators must enter the resource title, URL, a resource description and the resource owner. The title, resource description, resource owner are displayed to the users upon their demand.

The administrator can open the resource for subscribers, which means it is displayed under new resources on the resource selection page for users. The administrator can block or deblock the resource access for all users.

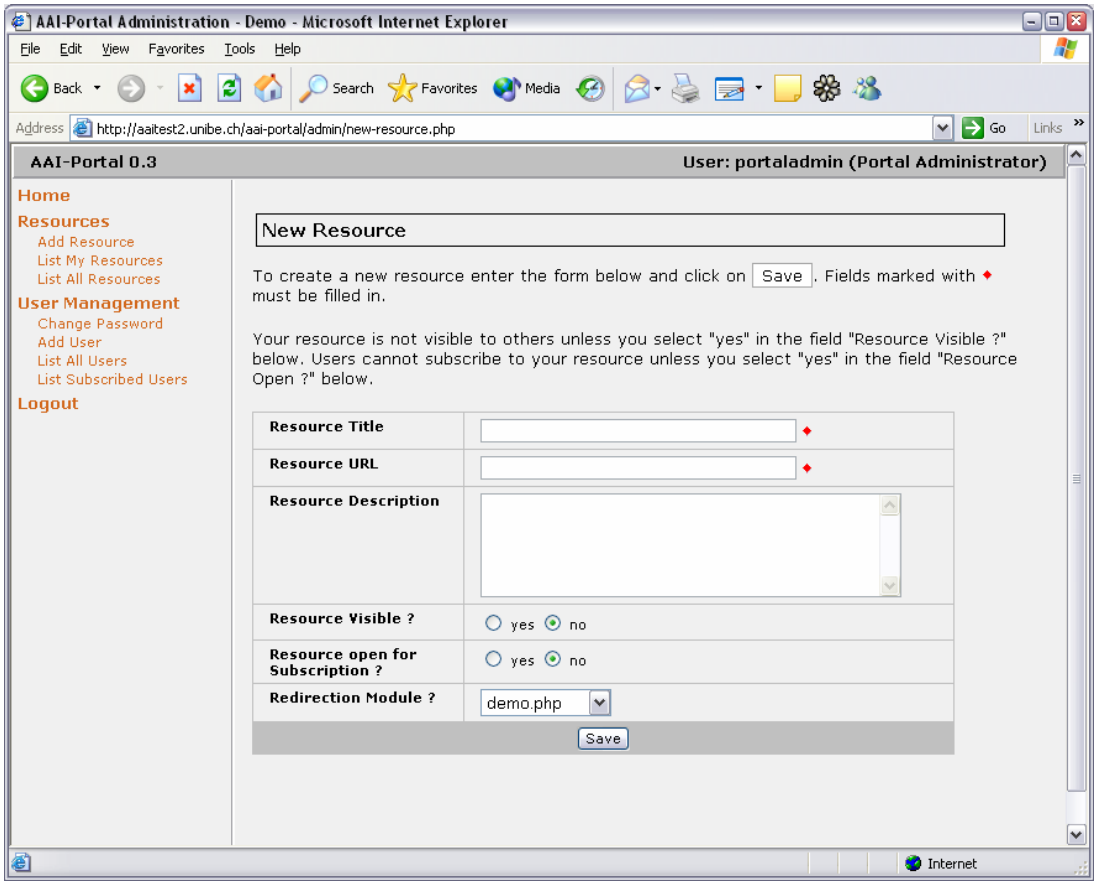

Figure 5.12: New resource page for administrators.

Figure 5.13 shows the ‘My Resource’ page for administrators and Figure 5.14 the ‘All Resources’ page that shows all hosted resources to AAI portal administrators and the own resources to resource administrators.

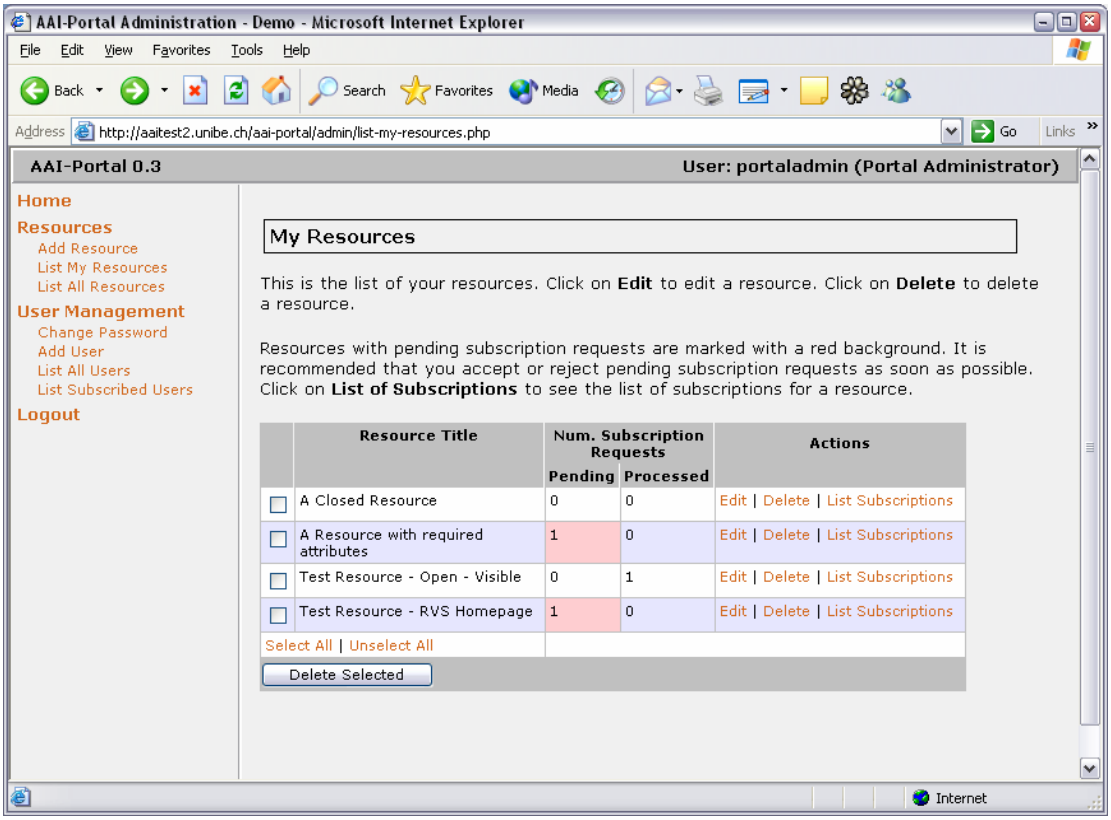

Figure 5.13: My resources page for administrators.

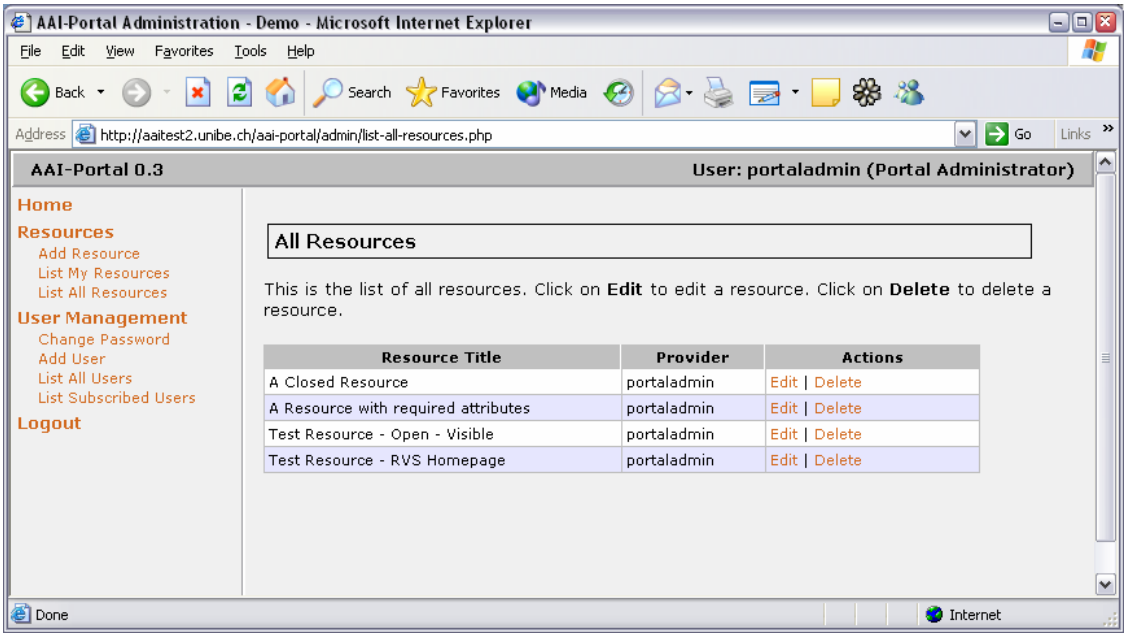

Figure 5.14: All resources page for administrators.

Figure 5.15 shows the resource authorization policy page. Each resource has one resource authorization policy page. Here, administrators can choose which of the predefined attributes from the AAI they want to get from the user (AAI or user provided) and can additionally define further attributes that are not on the list. Figure 5.15 does not show the complete attribute catalog from the AAI because we did not want to overload the image, in reality there would be all attributes. All attributes chosen here are required for the resource subscriber. Administrators cannot update the resource policy if the resource is open for subscription. If there is a need for other policy settings for the same resource it is recommended to add the same resource with another suffix to the AAI portal.

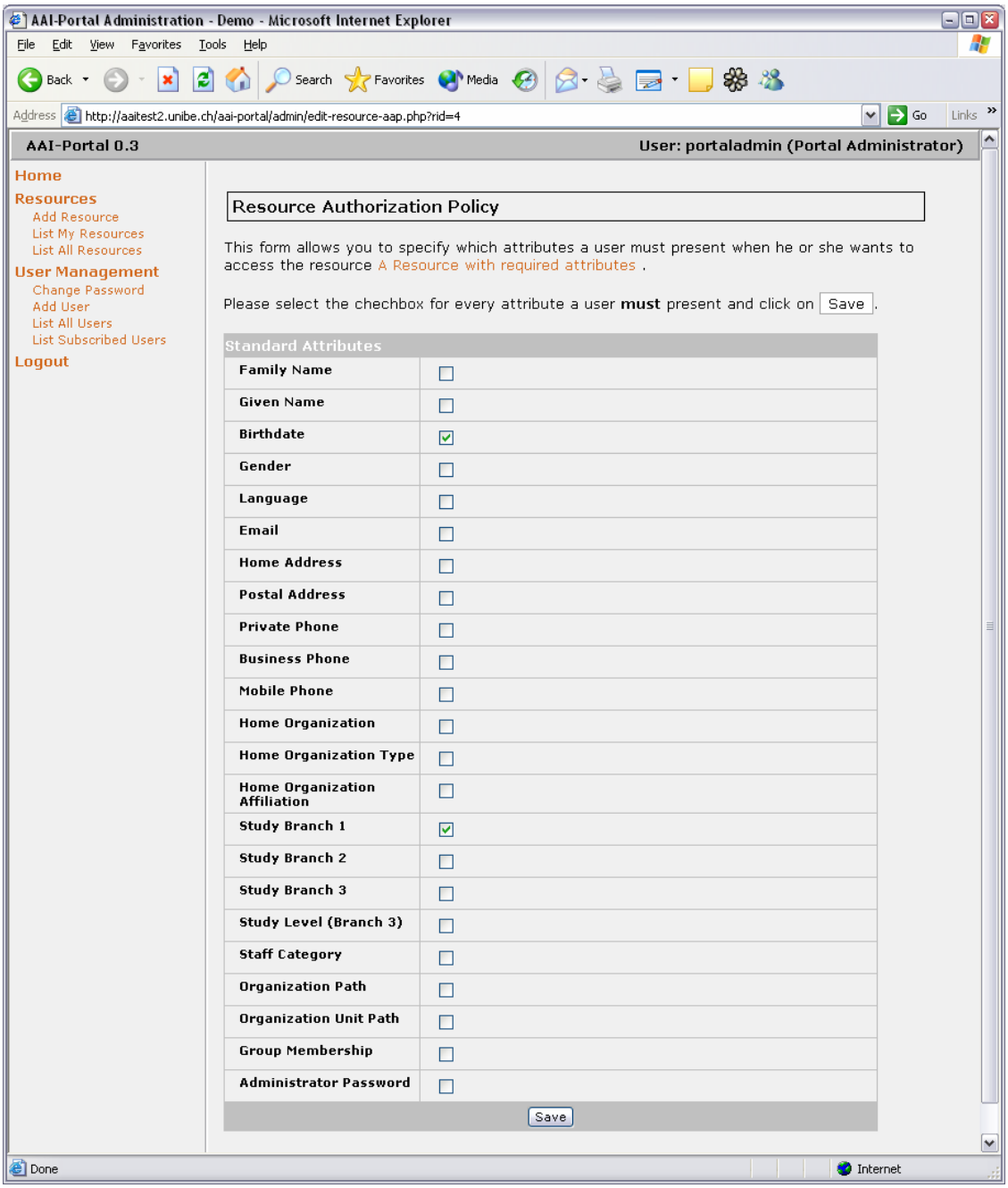

Figure 5.15: Resource authorization policy page for administrators.

Figure 5.16 shows the resource subscriber list. The page displays two lists; subscribers waiting to get accepted or declined and accepted rejected or revoked subscribers. On this page, administrators can accept, decline or revoke users for the resource. The state of the user is displayed as accepted, declined or revoked. Pressing accept or decline buttons is only possible if the resource policy includes a pending subscription list for the subscribers. Declined users remain on the resource list until the resource is deleted. The revoke button is for misbehaving users that need to be taken off the resource. There must be contact information on the same screen. “Mailto” and “SMS” allow the administrator to send notification messages about the status of the respective user. “Mailto all subscribers” sends a status notification message to all the users on the resource list. This is not the same notification as in the case of a status change. Each status change automatically releases a notification message to the user if an email or SMS number was provided.

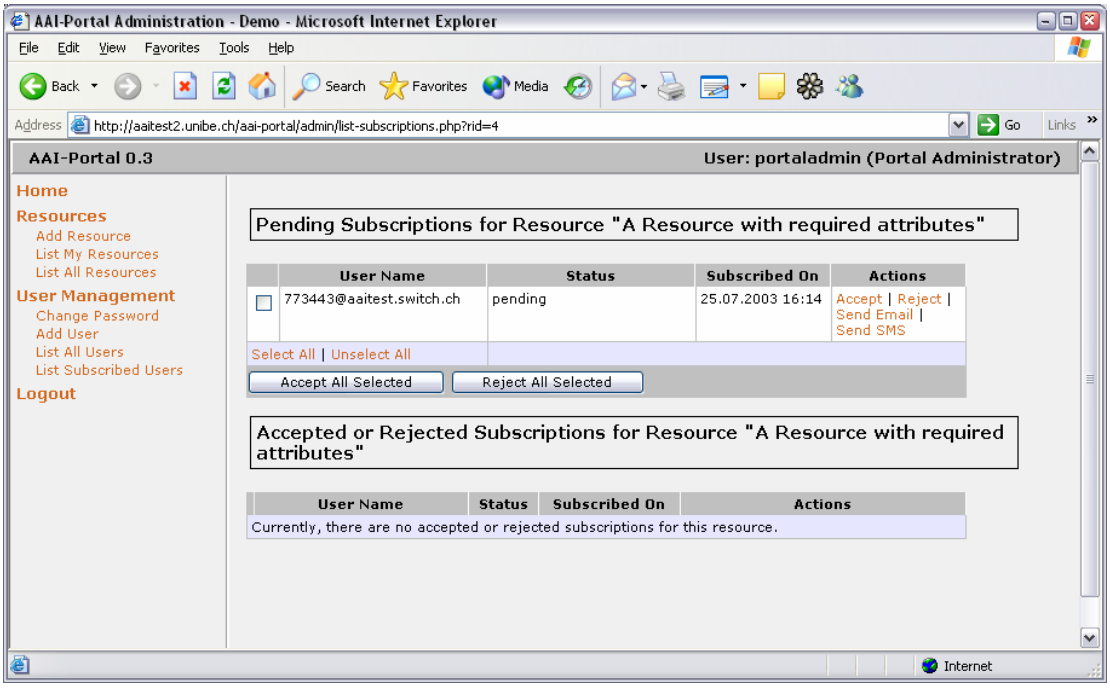

Figure 5.16: Resource subscriber list for administrators.

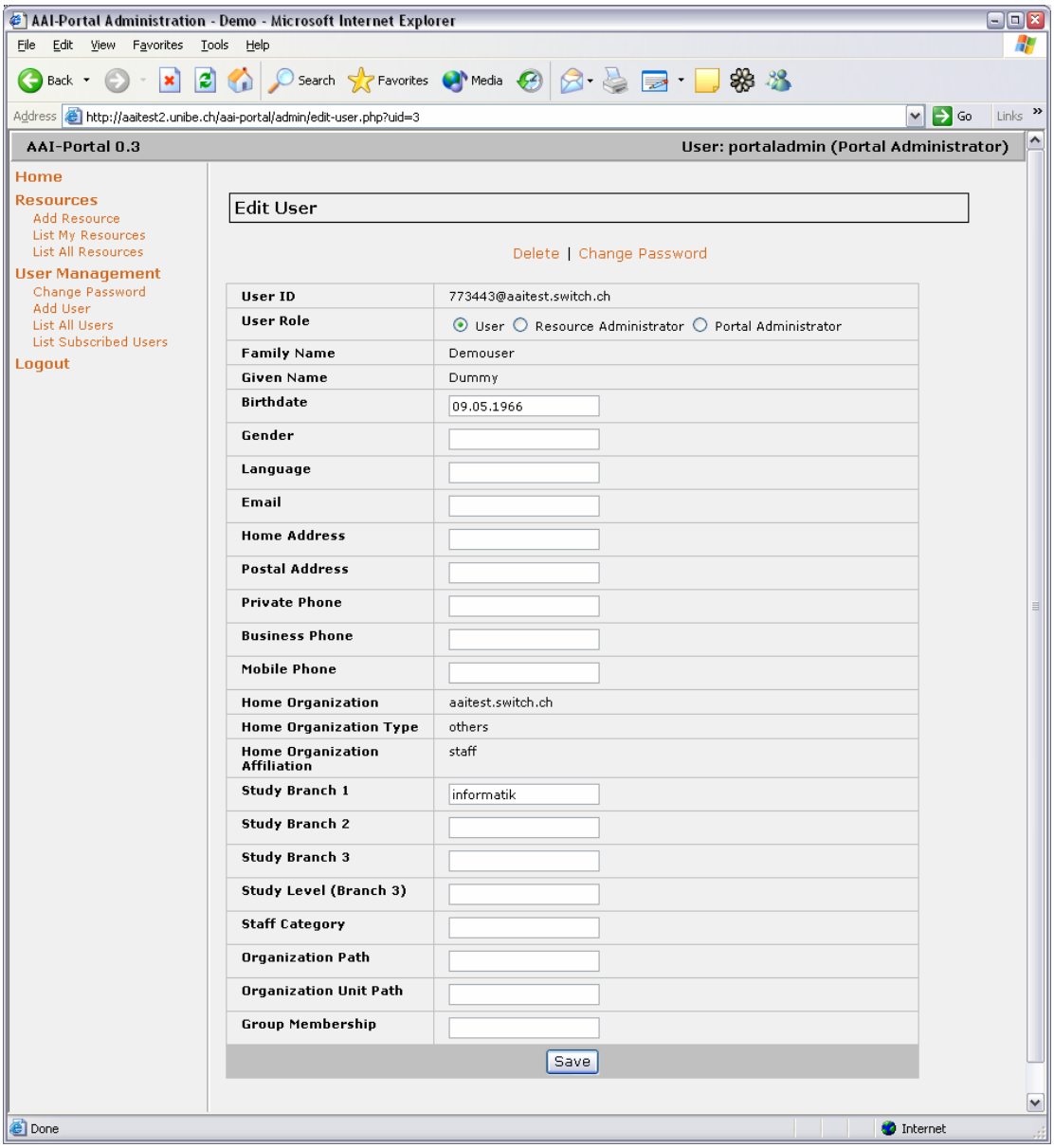

Figure 5.17: User details.

Figure 5.17 shows the detailed entry for a subscribed user. There may be more than one data sets per user, for each resource one. All collected user data is shown, data delivered by the AAI cannot be modified.

5.5 AAI Portal Policy in the Case of our Prototype Implementation

As the AAI portal is the gateway between the realm of the core AAI and the realm of the resources it possesses two policy sets:

a) The policy as a resource connected to the AAI. AAI-enabled resources must be conforming to the AAI policy. For AAI portal operators the attribute acceptance policy is very important. The default portal policy is set to get all the available attributes from the AAI as there can be many resources on the portal all with their own requirements for authorization data.

b) The policy as a resource host. Everything that is beyond the technical policy is freely definable by the AAI portal operator. The AAI portal policy applies to the hosted resources. Hosted resources have to define their own policies for their resource subscribers. They must be conformant to the before mentioned policies. The technical part of such a policy, the AAP of the hosted resources can be defined by the resource administrator by choosing the required attributes from the list of available attributes. Additionally, freely definable attributes can be added. Added attributes are per se required.

The attributes provided by the AAI are reliable and indicated in a way that administrators as well as users can differentiate AAI or non-reliable user provided attributes. Users get the possibility to fill in missing attributes that were not delivered by the AAI. Nobody can modify AAI provided attributes, whereas administrators can modify user provided information.

By default, user provided attributes are stored six month longer than subscribed resources are open or six month longer than the last user login occurred.

The AAI portal displays on each page the user’s surname and given name.

6 Outlook

The AAI portal architecture shows how to connect AAI-enabled and non-AAI-enabled resources to an authentication and authorization infrastructure. The AAI portal also provides a set of resource management features that is enough to manage resources but not enough to comfortably manage resource communities.

The architecture could be extended in the areas of accounting and community management. Accounting can provide valuable statistical information about resource visits and the time users spent in resources. Accounting is also related to resource billing, a controversial topic for resource subscribers as well as for resource providers.

It is very tempting to add community management features to the AAI portal. All resource subscribers regularly visit the AAI portal. Each user could get personalized access to chat and discussion boards, group matching processes and more that belong to the respective subscribed resources.

One could also imagine extending the AAI portal by collecting students' work results. For that purpose, resources must provide back user data. The architecture is not yet prepared for this enhancement.

Before implementing the above mentioned routines it is highly recommended to enhance the architecture. This helps to implement uniform extensions for a generic use.

7 References

CE02 Cantor S., Erdos M.: Shibboleth-Architecture DRAFT v05. NSF Middleware Initiative Draft. May 02, 2002

CL01 Castro-Rojo R., López D.R.: The PAPI System: Point of Access to Providers of Information. Terena. 2001

Gc03 Graf Christoph et al.: Architecture Evaluation. SWITCH. Jan 20, 2003. www.switch.ch/aai

Gm01 Guggisberg M. et al.: An Interdisciplinary Virtual Laboratory on Nanoscience. Electronic Notes in Future Generation Computer Systems, Elsevier, Vol. 1 (2001)

I18N Character Model for the World Wide Web 1.0. W3C Working Draft 30 April 2002. www.w3.org/TR/2002/WD-charmod-20020430/

Ims Institutional Management System, new name: IMS Global Learning Consortium, Inc. www.imsglobal.org

Oasis Organization for the Advancement of Structured Information Standards. www.oasis-open.org

Passport Microsoft .NET Passport. www.passport.com

Rediris RedIRIS. Spanish National Research Network. www.rediris.es

Saml Security Assertion Markup Language (SAML) is an OASIS standard. www.oasis-open.org

Shibboleth Internet2 Middleware. http://shibboleth.internet2.edu/

Sm02 Steinemann M.-A. et al.: Global Architecture and Partial Prototype Implementation for Enhanced Remote Courses, Computers and Advanced Technology in Education (CATE 2002), Cancun, Mexico, May 20-22, 2002, ISBN 0-88986-332-6, pp. 441-446

Svc Swiss Virtual Campus. www.swissvirtualcampus.ch

Switch The Swiss Eduacation & Research Network. http://www.switch.ch

Vitels Virtual Internet and Telecommunications Laboratory of Switzerland. www.vitels.ch

Webct Web Course Tool. http://archives.internet2.edu/guest/archives/I2-NEWS/log200212/msg00002.html

Wj01 Joel Weise. Public Key Infrastructure Overview. Sun BluePrints Online. August 2001. www.sun.com/solutions/blueprints/0801/publickey.pdf

Wt03 Wason Tom: Liberty ID-FF Implementation Guidelines. Draft Version 1.2-02. Liberty Alliance Project. April 14, 2003

8. Acknowledgements

We would like to acknowledge the contributions to the work presented in this article of all involved persons. We especially appreciate the financial support by the Swiss Virtual Campus [Svc] and the technical support by the SWITCH AAI team.

Appendices

Appendix A, Swiss AAI Attributes

The final version 1.0 of the Attribute Specification document is available at:
http://www.switch.ch/aai/docs/AAI_Attr_Specs.pdf

The directory: http://www.switch.ch/aai/docs/
will always provide the most current versions of final documents.
The names of the documents will not change but just link to the latest version available in one of the subdirectories which are directly accessible as well.

Table A-1 lists individual attributes and Table A.2 lists group membership attributes how they were in use in October 2003.

"aai_swissEduPersonUniqueID"
"aai_surname"
"aai_givenName"
"aai_swissEduPersonBirthdate"
"aai_swissEduPersonGender"
"aai_preferredLanguage"
"aai_mail"
"aai_homePostalAddress"
"aai_postalAddress"
"aai_homePhone"
"aai_telephoneNumber"
"aai_mobileTelephoneNumber"

Table A.1: Individual attributes.

"aai_swissEduPersonHomeOrganization"
"aai_swissEduPersonHomeOrganizationType"
"aai_eduPersonAffiliation"
"aai_swissEduPersonStudyBranch1"
"aai_swissEduPersonStudyBranch2"
"aai_swissEduPersonStudyBranch3"
"aai_swissEduPersonStudyLevel"
"aai_swissEduPersonStaffCategory"
"aai_swissEduPersonOrgDN"
"aai_swissEduPersonOrgUnitDN"
"aai_swissEduPersonEntitlement"

Table A.2: Group membership attributes.

Table A.3 shows the seven required attributes that a Home Organization must provide to the AAI when this text was written in October 2003.

"aai_swissEduPersonUniqueID"
"aai_surname"
"aai_givenName"
"aai_swissEduPersonHomeOrganization"
"aai_swissEduPersonHomeOrganizationType"
"aai_eduPersonAffiliation"

Table A.3: Required attributes in the Swiss AAI.

Appendix B, Acronyms

AA	Attribute Authority
AAI	Authentication and Authorization Infrastructure
AAP	Attribute Acceptance Policy
ARP	Attribute Release Policy
Clubs	A club is a group of organizations who agree to exchange attributes using the SAML/Shibboleth protocols and abide by a common set of policies and practices.
DS	Directory Service
HS	Handle Service
IMS	Institutional Management System http://www.imsproject.org/
LDAP	Lightweight Directory Access Protocol
MACE	Middleware Architecture Committee for Eduacation
OASIS	Organization for the Advancement of Structured Information Standards
OpenSAML	Open Security Assertion Markup Language http://middleware.internet2.edu/opensaml/
PAPI	Point of Access to Providers of Information
PoA	Point of Access server in the PAPI architecture
Pubcookie	-> WebISO
RM	Resource Manager
SAML	Security Assertion Markup Language http://www.oasis-open.org/committees/security/
SHAR	Shibboleth Attribute Requester
Shibboleth	Shibboleth \Shib"bo*leth\, n. [Heb. shibb[=o]leth an ear of corn, or a stream, a flood.] http://middleware.internet2.edu/shibboleth/
SHIRE	Shibboleth Indexical Reference Establisher
MySQL	My Structured Query Language Open Source relational database management system that uses Structured Query Language http://www.mysql.com/
SSO	Local Single Sign On system
Tomcat	Tomcat is the servlet container that is used in the official Reference Implementation for the JavaServlet and JavaServer Pages technologies. http://jakarta.apache.org/tomcat/
WAYF	Where Are You From? Server in Shibboleth
WebISO	Web Initial Sign-On http://middleware.internet2.edu/webiso/

Appendix C, List of Figures

Appendix D, List of Tables