**COMPETITIVENESS AND INNOVATION FRAMEWORK PROGRAMME**
ICT Policy Support Programme (ICT PSP)

Towards pan-European recognition of electronic IDs (eIDs)

**ICT PSP call identifier:** ICT-PSP/2007/1
**ICT PSP Theme/objective identifier:** 1.2

# Project acronym: STORK

Project full title: Secure Identity Across Borders Linked
Grant agreement no.: 224993

# Moodle eID Implementations
# Technical Documentation v1.8

| | |
|---|---|
| **Deliverable Id :** | |
| **Deliverable Name :** | |
| **Status :** | **draft** |
| **Dissemination Level :** | |
| **Due date of deliverable :** | |
| **Actual submission date :** | |
| **Work Package :** | **WP 6.2** |
| **Organisation name of lead contractor for this deliverable :** | **AT TUG** |
| **Author(s):** | **Thomas Knall, Tobias Kellner** |
| **Partner(s) contributing :** | |

**Abstract:** This document describes the components that have been developed to enhance Moodle's authentication capabilities in order to meet the pilot's requirements. These components can be categorized into two categories: one which involves Moodle specific enhancements like an authentication plug-in as well as a modified chat module, the other category consists of a so called "eID Connector" which acts as the connecting interface between Moodle authentication and the STORK WP5 infrastructure.

**Project co-funded by the European Community under the ICT Policy Support Programme**

# History

| *Version* | *Date* | *Modification reason* | *Modified by* |
|---|---|---|---|
| 0.1 | 13 01 2010 | Initial draft | AT TUG |
| 0.2 | 21 01 2010 | Installation of Moodle specific component | AT TUG |
| 0.3 | 25 01 2010 | Installation of the eID Connector | AT TUG |
| 0.4 | 26 01 2010 | Configuration of Moodle components | AT TUG |
| 0.5 | 27 01 2010 | Configuration of eID Connector | AT TUG |
| 0.6 | 28 01 2010 | Configuration of eID Connector, Moodle Registration, Moodle Authentication | AT TUG |
| 1.0 | 29 01 2010 | Finalization | AT TUG |
| 1.1 | 16 02 2010 | Minor updates regarding configuration of eID Connector (password salting, legacy passwords) | AT TUG |
| 1.2 | 23 02 2010 | Support for defining QAA levels added.<br>Some lines regarding section "Installing/replacing Moodle chat module" added.<br>Documentation in alignment with implementation version v1.2.2 | AT TUG |
| 1.3 | 17 03 2010 | Maintenance mode configuration option added.<br>Notes on signature request/signature response added. | AT TUG |
| 1.4 | 15 04 2010 | Text added noting that QAA level can also be set for plain Moodle authentication.<br>Setting introduced that allows enabling/disabling usage of existing Moodle accounts for STORK.<br>Explanation added on how to suppress the display of surnames for data protection purposes. | AT TUG |
| 1.5 | 04 07 2010 | Section added dealing with configuration of PEPS specific interface.<br>Updates regarding package structure (configuration for signature validation framework).<br>Updates in terms of the section describing the setup of tomcat.<br>Example configuration extended with section "STORKResponseValidation".<br>Outline of generic PEPS authentication provider added. | AT TUG |
| 1.6 | 24.09.2010 | Section for logging added. | AT TUG |
| 1.7 | 20.12.2010 | Important note regarding Moodle cleanup service added.<br>Instructions added for compilation of the source project. | AT TUG |
| 1.8 | 21.10.2011 | Instructions for modification of Moodle login form added | AT TUG |

# Table of contents

# List of figures

# List of tables

# List of abbreviations

| | |
|---|---|
| AJP | Apache JServ Protocol |
| CAS | Central Authentication Service |
| CCS | Citizen Card Software |
| CSV | Comma Separated Value |
| eID | Electronic Identifier |
| HTTP | Hypertext Transfer Protocol |
| IDP | Identity Provider |
| IETF | Internet Engineering Task Force |
| IP | Internet Protocol |
| ISO | International Organization for Standardization |
| JCE | Java Cryptography Extension |
| JDBC | Java Database Connectivity |
| JDK | Java Development Kit |
| MOODLE | Modular Object-Oriented Dynamic Learning Environment |
| PEPS | Pan European Proxy Services |
| QAA | Quality Authentication Assurance |
| RFC | Request for Comments |
| SAML | Security Assertion Markup Language |
| SMS | Short Message Service |
| SQL | Structured Query Language |
| SSL | Secure Sockets Layer |
| STORK | Secure idenTity acrOss boRders linKed |
| URL | Uniform Resource Locator |
| UTF-8 | 8-bit UCS Transformation Format |
| V-IDP | Virtual Identity Provider |
| WAR | Web (Application) Archive |
| XML | Extensible Markup Language |

# 1 Overview

In order to connect Moodle to the STORK interoperability layer three components have been developed or adjusted respectively:

- A Moodle STORK authentication plug-in
- A modified Moodle chat module (now named "SaferChat")
- An external component, referred to as the "eID Connector"

*Figure 1* shows the architecture that has been chosen.

## 1.1 Components

This chapter outlines the main components used for SaferChat authentication.

### 1.1.1 eID Connector

Apart from the platform kids are supposed to use in order to collaboratively work on common projects (making use of the platform's chat rooms) a second component, called "eID Connector" plays a crucial part. Its purpose is to perform more complex processes like communication with the STORK WP5 modules or registration tasks. For the connection with the STORK interoperability layer the connector provides a generic interface which can easily be adapted. This makes the connector very flexible. As far as Moodle is concerned the connector provides a lightweight interface.

For registration purposes the eID connector makes use of the internal Moodle database. Therefore it is highly recommended to run the eID connector and Moodle on the same server.

### 1.1.2 Moodle STORK authentication plug-in

Once a user has authenticated using the eID Connector and the WP5 modules, the credentials are handed over to the Moodle STORK authentication plug-in. Since Moodle provides means to implement own authentication plug-ins ([1]) a special STORK authentication plug-in has been developed which communicates with the eID connector.

### 1.1.3 Moodle Chat Module

The already built-in chat Module has been extended in order to support arbitrary age ranges as needed for SaferChat. Since the SaferChat Module uses a modified database scheme the existing Moodle chat room activity has to be uninstalled and replaced (refer to section 2.2.3).

Chat rooms in terms of Moodle are so called "activities" which can be arbitrarily set up within any Moodle course. Each SaferChat activity can be individually configured so that multiple chat rooms for kids of different age groups are possible.

## 1.2 Prerequisites

The basic requirements are:

- a working instance of Moodle 1.9.x (refer to [2] for Moodle specific requirements) Note that the implementations have been tested with Moodle 1.9.5.
- physical access to the server hosting Moodle (for replacement of chat activity by SaferChat)
- eID Connector
    - Apache Tomcat 5.x or 6.x
    - JDK 1.5.x[1] or 1.6.x[2] (with Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files)
      Please note that the eID Connector has been tested with JDK 1.6.20.
    - MySQL 5.x database (MyISAM or InnoDB)
    - Access to the internal Moodle database (MySQL 5.x, MyISAM or InnoDB)

## 1.3 Basic authentication concept

The basic steps to perform authentication are as follows (refer to *Figure 1*):

1. The first step of getting access to a Moodle instance is always done by a kid entering the site and choosing "eID authentication" instead of performing an old-fashioned username/password authentication.
2. The "Moodle Connector" plays an essential part as far as the connection with the STORK interoperability layer is concerned. Upon authentication the connector calls the registered country specific WP5 modules (e.g. a framework which handles the local PEPS communication which is not explicitly defined in STORK context). These modules need to implement a certain interface (refer to section 3.4 for further information).
3. The country specific interface implementation contacts the PEPS/V-IDP of its country which in turn contacts the kid's PEPS or invokes the kid's Middleware implementation (*Figure 1* contains a simplified version just denoting an "Identity Provider"). In case of a V-IDP contacting a PEPS a signature request is added.
4. The "Identity Provider" performs authentication using the kid's eID implementation.
5. After completion of the authentication the Identity Provider is in possession of the kid's credentials.
6. The Identity Provider creates a corresponding response including the information needed for a successful Moodle authentication (the age, an unique identifier, the underlying STORK QAA level and optionally the name of the kid for registration purposes if desired) as well as a signed signature response. If the Identity Provider is a PEPS responding to a V-IDPs request a signature response containing the user's certificate is added. The Identity Provider returns the credentials to the eID connector using the country specific interface implementation.

---

1 http://www.oracle.com/technetwork/java/javase/downloads/index-jdk5-jsp-142662.html

2 http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html

7. The connector gets the response, retrieves the information needed and maps the unique identifier to a Moodle user identifier. If needed a registration dialogue is shown allowing the user to create a new Moodle account or to use an existing account (Note that the usage of existing accounts may be disabled by configuration. Refer to section 3.2.2 for further information). Finally the eID attributes are transferred in a lightweight manner using HTTP header fields to Moodle.
8. Moodle's STORK authentication plug-in receives the credentials, logs the user in and temporarily puts the age information as well as the underlying QAA level to the user's account data.

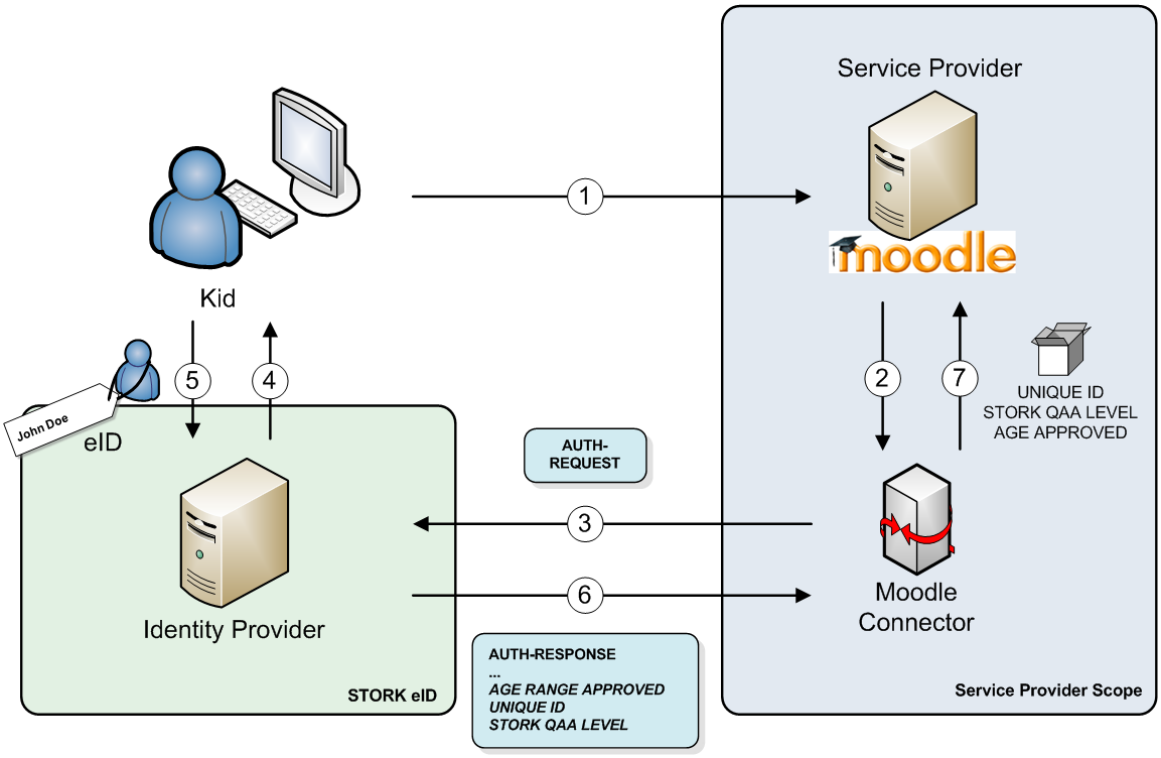


***Figure 1: Authentication concept***

## 1.4 Technical details

*Figure 2* shows the authentication process from a more technical point of view including the Austrian V-IDP. In order to allow the connection with different authentication services the external interface has been implemented in a generic manner. In case of Austria a special implementation of this interface is in use that communicates with the Austrian V-IDP (*Figure 2*, step 3-6).

As shown in *Figure 2* some interfaces are reachable from the Internet (left hand side) and some are only reachable within the server infrastructure (right hand side). The external communication mainly involves redirection of the user to the respective authentication service (step 3, V-IDP in case of *Figure 2*).

The internal communication is used to transmit credentials from the eID Connector and the Moodle STORK authentication plug-in, step 8-10 (as well as in the case of Austria to transmit credentials from V-IDP to the Austrian authentication implementation of the eID connector, step 5 and 6).

eID attributes are transferred from the eID Connector to the Moodle STORK authentication plug-in using HTTP header fields. Refer to sections 3.1.1 and 3.2.4 for more information on the mapping of eID attributes to HTTP header fields. Since the values being transferred are based on UTF-8 which allows the usage of special characters these values have to be converted into Base64 (refer to [9] for more information) values. The STORK Moodle authentication plug-in automatically decodes these Base64 encoded values resulting in the original UTF-8 based values. The body of the response (see below) contains a single "<ok/>" to indicate that the authentication has successfully completed.

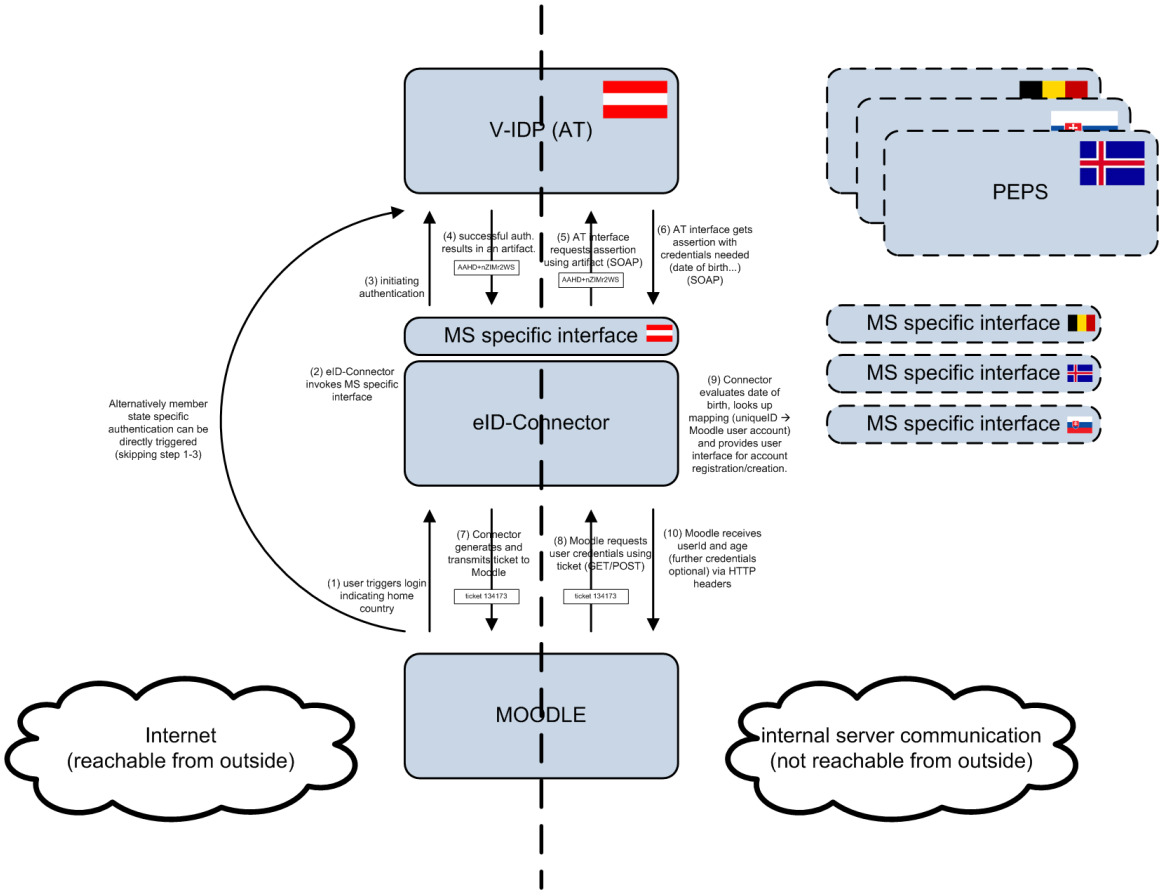


***Figure 2: Detailed authentication process***

### Step 8: Example request (Moodle STORK authentication plug-in requesting eID Connector)

```
GET /moodle-eid-
    connector/connector.do?action=getAuthenticationData&ticket=AAFvFYBiH5uk4itj%2FiE2abYM
    Tro8yn1GWd70tcDPoYRf70M90G2eDEnd HTTP/1.0
Host: 127.0.0.1:18080
User-Agent:
Accept:
    text/xml,application/xml,application/xhtml+xml,text/html,text/plain,image/png,image/j
    peg,image/gif,*/*
Accept-encoding: gzip
Accept-language: en-us
```

### Step 10: Example response (in response to request in step 8)

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=B97FDD2AB5F01615A394305EB0D3431B; Path=/moodle-eid-connector
Pragma: No-cache
Cache-Control: no-cache,no-store,max-age=0
Expires: Thu, 01 Jan 1970 00:00:00 GMT
X-STORK-givenName: TWF4
X-STORK-qaaLevel: NA==
X-STORK-age: NzA=
```

```
X-STORK-surname: TXVzdGVybWFubg==
X-STORK-Moodle-username: OWQwZDRiNzVkMDFkNWRjMGEyODJlYzQ0MmUxYTNjM2YwZDMxYjBmNQ==
X-STORK-service: Moodle eID-Connector
Content-Type: text/xml;charset=UTF-8
Content-Length: 5
Date: Thu, 28 Jan 2010 10:46:08 GMT
Connection: close

<ok/>
```

# 2 Installation

The installation of the components needed for the pilot are to be divided into two categories: Installation of Moodle specific components and installation of the eID Connector.

## 2.1 Package Structure

The preliminary deployment package features the following structure.

| Path | Description |
|---|---|
| `deployment` | Folder containing files needed for deployment with Apache Tomcat. |
| `deployment/jdk` | Files that need to be put in the corresponding JDK installation. |
| `deployment/jdk/unlimited_strength_jurisdiction_policy_files` | This folder contains the JCE Unlimited Strength Jurisdiction Policy Files for JDK 1.5 and 1.6 that need to be added to the JDK being used. |
| `deployment/moodle/modules_for_moodle-1.9.x` | Folder containing files needed to be added to the Moodle installation. |
| `deployment/moodle/modules_for_moodle-1.9.x/admin/cron.php` | Modified cron job that prevents STORK accounts being removed by Moodle's internal clean up service (refer to section 3.1.2) |
| `deployment/moodle/modules_for_moodle-1.9.x/auth/stork` | The Moodle authentication plug-in for STORK. |
| `deployment/moodle/modules_for_moodle-1.9.x/lang` | Some texts (for the new modules) that need to be added to the internal Moodle language pool. |
| `deployment/moodle/modules_for_moodle-1.9.x/mod/chat` | The SaferChat activity module. |
| `deployment/moodle/modules_for_moodle-1.9.x/login` | Modification of the Moodle login form to include STORK login |
| `deployment/moodle/modules_for_moodle-1.9.x/modifications_for_dataprotection_purposes` | Modified Moodle files that provide maximum data protection (removing display of last name for instance). Note: This is optional and does not affect plain authentication. |
| `deployment/tomcat` | Folder containing files needed to be added to an Apache Tomcat when using Moodle eID Connector. |
| `deployment/tomcat/conf/moodle-eid-connector` | Sample configuration for the Moodle eID Connector. |
| `deployment/tomcat/conf/moodle-eid-connector/moa-spss` | Sample MOA-SP (signature verification for SAML responses) configuration. |

| Path | Description |
|---|---|
| deployment/tomcat/conf/moodle-eid-connector/moa-spss/certstore | Certstore of the sample MOA-SP configuration (the certstore which is used as temporary storage for certificates is needed for building of certificate chains) |
| deployment/tomcat/conf/moodle-eid-connector/moa-spss/certstore/toBeAdded | Certificates that are put into that folder are automatically added to the certstore. |
| deployment/tomcat/conf/moodle-eid-connector/moa-spss/trustProfiles/PEPS | Trust anchors (i.e. root certificates) for certificate validation of SAML messages. |
| deployment/tomcat/endorsed | Files that need to be put into the endorsed folder of the underlying Tomcat instance. |
| deployment/tomcat/scripts | Start scripts for the respective Apache Tomcat. |
| doc | The documentation. |
| Moodle-eID-Connector | The complete Moodle eID Connector Java project. |
| Moodle-eID-Connector/builds | A built of the Moodle eID Connector. |
| Moodle-eID-Connector/impl | This folder contains the complete Eclipse/Maven2 project. |
| Moodle-eID-Connector/impl/src | The Java source files (based on Maven2). |
| Moodle-eID-Connector/javadoc | JavaDoc |
| Moodle-eID-Connector/maven2/repository | The Maven2 repository that need to be used when compiling the project. |
| sample/tomcat-6.0.20-moodle-connector | Folder that contains a sample Tomcat instance with deployed and configured Moodle eID Connector. |

***Table 1: Package structure***

## 2.2 Installing Moodle components

In order to install the components within Moodle for SaferChat purposes administrative rights (i.e. the role "Administrator") are required. Preparing Moodle for SaferChat requires several steps to be conducted. First of all some language resources have to be put into the Moodle language pool (section 2.2.1). Second the STORK authentication plug-in has to be installed (section 2.2.2) within Moodle. The last step involves the replacement of the existing chat module with the new SaferChat module (section 2.2.3).

### 2.2.1 Updating Moodle language data

deployment package path: `deployment/moodle/modules_for_moodle-1.9.x/lang/en_utf8`

The modules being deployed need some extensions to the Moodle language pool. Just copy the file `chat.php` from the deployment package path mentioned above to `/lang/en_utf8` of your Moodle installation. As a second step please **append** the content of `add_to_auth.php` to your

`auth.php` in `/lang/en_utf8`. (Note: You can also perform this task using the administrative interface of Moodle: *Site Administration* → *Language* → *Language editing*).

### 2.2.2 Installing STORK authentication plug-in

deployment package path: `deployment/moodle/modules_for_moodle-1.9.x/auth/stork`

Just copy the folder stork into the `auth` folder of your Moodle installation. Check if the module has been recognized by clicking Site *Administration* → *Users* → *Authentication* → *Manage authentication*. The plug-in "STORK" should be listed.

### 2.2.3 Installing/replacing Moodle chat module

deployment package path: `deployment/moodle/modules_for_moodle-1.9.x/mod/chat`

In order to install the chat module the old chat activity module has to be uninstalled at first. To remove a module click *Site Administration* → *Modules* → *Activities* → *Manage Activities* and remove the chat module by clicking "Delete". Do not forget to physically remove the chat module from the file system (remove the folder `/mod/chat` from your Moodle installation) otherwise Moodle will reinstall it next time you access the site administration.

The installation of the SaferChat module can be accomplished as follows:

1. Copy the folder `chat` from the deployment package to the `moodle/mod` folder of your Moodle installation.
2. In your browser, go to the Moodle site: *Site Administration* → *Notifications* → *Continue*
3. Visit the admin notifications page of your web site (…`/admin/index.php`) to allow the plug-in to install itself.
4. Check if the new module has been installed: *Site Administration* → *Modules* → *Activities* → *Manage activities*. The "SaferChat" activity should be listed.
5. Do not forget to set your Server's name and IP address: *Site Administration* → *Modules* → *Activities* → *Manage activities* → *SaferChat*. Scroll down to the last section "Chat server daemon" and enter valid values for "Server name" and "Server ip". (Otherwise an error will be shown when entering chat rooms.)
6. Refer to [3] of the Moodle documentation for detailed information on installation and removal of modules.

### 2.2.4 Modifying Moodle login form

deployment package path: `deployment/moodle/modules_for_moodle-1.9.x/login`

To make the STORK login available to users, the Moodle login form has to be modified. The `login` folder in the deployment package contains such a sample modification:

1. Modify `moodle/login/index_form.html` according to the modifications outlined in the deployment package, use the provided `index_form.patch`, or simply overwrite the form.
2. Copy the other content from the deployment package to the `moodle/login` folder
3. Modify `ccs_selection.html`, `iframeOnlineBKU.html` and `iframeHandyBKU.html` with the correct URLs for your installation, replacing:
    - YOUR_MOODLE with the URL of your Moodle installation
    - YOUR_MOODLE_CONNECTOR with the URL of your Moodle eID connector
    - YOUR_MOA_ID with the URL of your MOA-ID installation
    - YOUR_BKUONLINE with the URL of your BKUOnline webapp

### 2.2.5 Removing display of surname

deployment package path: `deployment/moodle/modules_for_moodle-1.9.x/modifications_for_dataprotection_purposes`

For data protection purposes the display of the surname can be suppressed in Moodle. This is optional and does not affect plain authentication. This requires some Moodle source code to be modified:

1. First of all the text "Firstname" should be changed. For SaferChat purposes the text was changed to "Nick name/pseudonym". Edit file `/lang/en_utf8/moodle.php`.
2. Modify `/lib/moodlelib.php` (functions `fullname` and `user_not_fully_set_up`) according to file `modify_fullname_in_moodlelib.php` of the deployment package.
3. Modify `/user/editlib.php` (function `useredit_shared_definition`) according to file `modify_useredit_shared_definition_in_editlib.php` of the deployment package.
4. In order to enable those data protection settings put the line
   `$CFG->enablestrictdataprotection = true;`
   into `/config.php` of your Moodle instance.

Alternatively, you can use the provided patch file `modifications_for_dataprotection_purposes.patch`.

Note that screenshots of this document still contain the surname of the respective person. Once the surname suppression setting is enabled surnames are not displayed any more.

## 2.3 Installing eID Connector

The installation of the eID Connector can be done in two steps:

- Preparing the database
- Setting up Apache Tomcat

### 2.3.1 Preparing the database

The eID Connector needs two database connections. One connection to its own database, which is used to store mappings between eIdentifier and Moodle user accounts and one connection to Moodle's database in order to access the user table.

#### 2.3.1.1 Database for the eID Connector

There is no need to create any tables by hand, just set up a database user with rights to SELECT, INSERT, UPDATE and DELETE data as well as to CREATE, ALTER, and DROP tables. Open the configuration file of the eID Connector (refer to section 3.2.6 for details on the configuration of the databases) and enter the connection URL, the type and the credentials for accessing the database. On start-up the connector verifies the database and automatically creates tables if needed.

#### 2.3.1.2 Moodle database connection

As a second step the eID Connector needs to get access to the internal Moodle database. It is recommended to create a user with minimum rights needed (SELECT and UPDATE). Again enter the connection URL, the type of database and the credentials to the connector's configuration (section 3.2.6).

### 2.3.2 Setting up Tomcat

The eID Connector is a web application according to the Java Servlet Specification [5] 2.4. The application was designed, developed and tested using an Apache Tomcat 6.0 servlet container. Please refer to [4] for information on how to set up a Tomcat 6.0 servlet container.

The deliverable package already contains a Tomcat 6.0 servlet container with an already deployed eID Connector including a sample configuration.

In case of a manual deployment of the eID Connector the following three files/folders are essential:

- The configuration (file `application_config.xml`) is located in folder `deployment/tomcat/conf/moodle-eid-connector`.
- The endorsed libraries (folder `endorsed`) for the respective Apache tomcat instance which is located in folder `deployment/tomcat/endorsed`.
- The web application (`moodle-eid-connector.war`) is located in folder `Moodle-eID-Connector/builds`.

In order to set up an instance of the eID Connector both the configuration and the web application are required. After adjusting the configuration (adjusting data base credentials and declaring specific authentication implementations.) the servlet container can be started up.

## 2.4 Building Moodle eID Connector

In order to build the Moodle eID Connector from Java source the following components are necessary:

- JDK 1.5 or newer
- Apache Maven2[3]
- The Maven2 repository that is included in the package that comes along with this documentation (refer to section 2.1).

The package also contains Eclipse project files so that the project can easily be imported and built using Eclipse Development Environment.

In order to build the project without using Eclipse conduct the following steps:

1. Make sure that javac can be invoked from your shell.
2. Copy the Maven2 repository that is included in the package that comes along with this documentation into your Maven2 repository.
3. Change to `Moodle-eID-Connector/impl`.
4. Build the package:
   ```
   mvn clean process-resources package
   ```
5. You'll find the resulting WAR file in `Moodle-eID-Connector/impl/target`.

[3] http://maven.apache.org/

# 3 Configuration

The configuration to be performed is divided into two domains:

- configuration of Moodle components: This involves configuration of the STORK authentication plug-in (refer to section 2.2.2 for installation of the plug-in), configuration of the Moodle cleanup service (so that accounts with incomplete user information are not automatically being removed) and configuration of an arbitrary number of SaferChat activities (refer to section 2.2.3 for information on installation of the modified chat module)
- configuration of the Moodle eID Connector

## 3.1 Configuring Moodle components

Once the STORK authentication plug-in and the SaferChat module have been installed, some configuration tasks have to be performed.

### 3.1.1 Configuring STORK authentication plug-in

Like any other Moodle authentication plug-in the STORK authentication plug-in can be configured using the administration interface. Login as administrator and click *Users* → *Authentication* → *Manage authentication*. If the STORK authentication plug-in installation was successful the plug-in should be listed. If the plug-in is disabled activate it by clicking on the (closed) eye symbol.

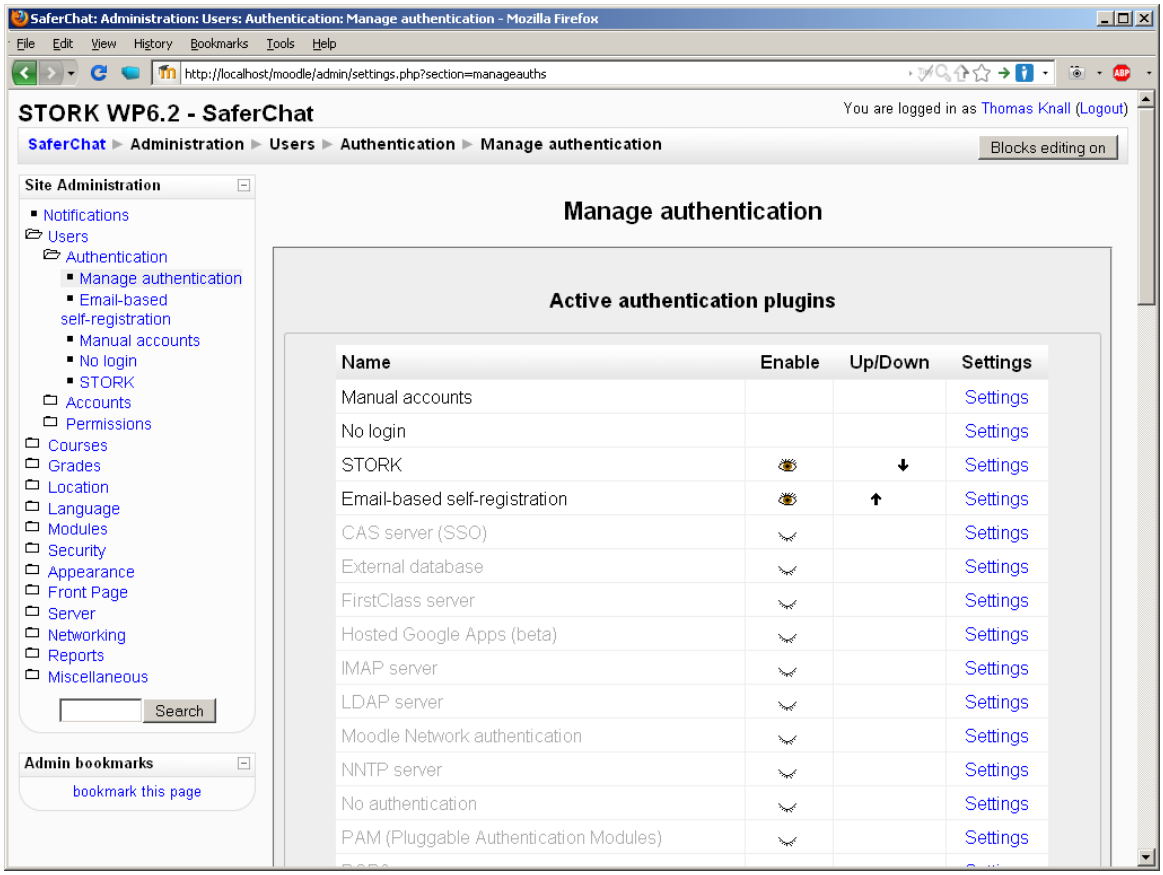


***Figure 3: List of authentication plug-ins***

In order to configure the plug-in click *Settings*. The following configuration interface should be shown:

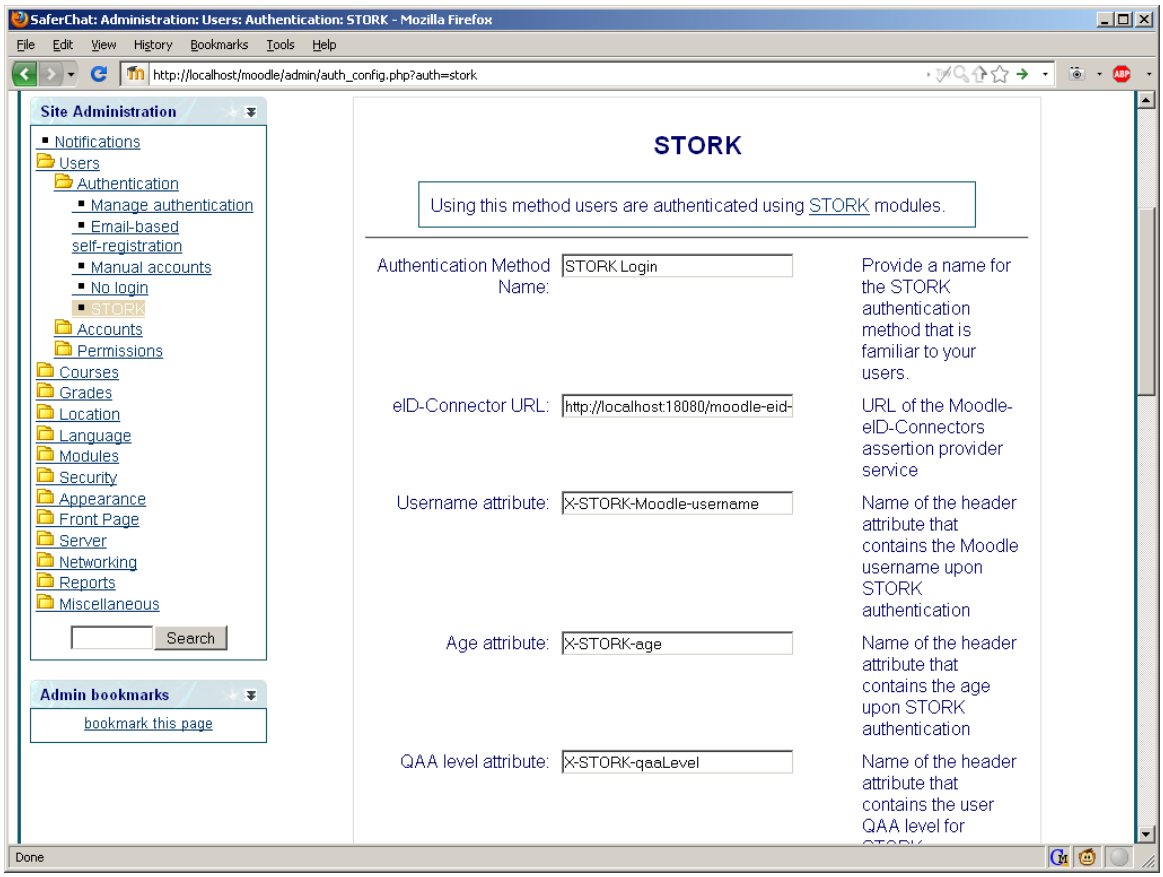


***Figure 4: Configuration interface of the STORK authentication plug-in***

The following table gives an overview and description of the configuration keys:

| Configuration key | Example | Description |
|---|---|---|
| Authentication Method Name | `STORK Login` | This is just a term Moodle is using. The name does not affect the behaviour of the authentication plug-in. |
| eID-Connector URL | `http://localhost:18080/moodle-eid-connector/connector.do?action=getAuthenticationData` | This is the internal connection (therefore "localhost") to the Moodle eID Connector. The STORK authentication plug-in uses this connection to retrieve the user's credentials. The credentials are transferred using the below mentioned HTTP header fields. |
| Username attribute: | `X-STORK-Moodle-username` | This is the HTTP header field that is used to transmit the Moodle username of the (newly) registered user.<br>Since the eID Connector inside the deployment package has already been pre-configured it is recommended to use this exemplary header name. |

| Configuration key | Example | Description |
|---|---|---|
| Age attribute: | `X-STORK-age` | This is the HTTP header field that is used to transmit the age of the authenticated user.<br>Since the eID Connector inside the deployment package has already been pre-configured it is recommended to use this exemplary header name. |
| QAA level attribute: | `X-STORK-qaaLevel` | This is the HTTP header field that is used to transmit the QAA level (indicating the quality of the authentication).<br>Since the eID Connector inside the deployment package has already been pre-configured it is recommended to use this exemplary header name. |
| Minimum QAA level: | `QAA level 4 – High assurance` | Using this setting a minimum QAA level needed for authentication at Moodle can be defined. Authentication methods not satisfying that minimum level are denied. This setting also allows to turn off QAA level check.<br>Note that QAA level for single chat rooms can be individually configured within the respective chat room configuration. Since Moodle has to be entered before any chat room can be used it makes no sense to configure chat rooms with QAA levels lower than this QAA level for Moodle authentication. |
| Data mapping: First name (optional) | `X-STORK-givenName` | This is the HTTP header field that is used to transmit the given name of the authenticated user.<br>Since the eID Connector inside the deployment package has already been pre-configured it is recommended to use this exemplary header name.<br>This configuration option allows to define the behaviour of the STORK authentication plug-in if a given name attribute has been transmitted. If desired (“Update local”) the given name of the Moodle account is automatically updated on every login using the given name from the eID credentials or the given name is only set during registration. Additionally the Moodle account data field given name can be locked (“Local value”, always or only if empty) preventing the user from changing it. |

| Configuration key | Example | Description |
|---|---|---|
| Data mapping: Surname (optional) | `X-STORK-surname` | This is the HTTP header field that is used to transmit the surname of the authenticated user.<br><br>Since the eID Connector inside the deployment package has already been configured it is recommended to use this exemplary header name.<br><br>This configuration option allows to define the behaviour of the STORK authentication plug-in if a surname attribute has been transmitted.<br><br>This configuration option allows to define the behaviour of the STORK authentication plug-in if a surname attribute has been transmitted. If desired ("Update local") the surname of the Moodle account is automatically updated on every login using the surname from the eID credentials or the surname is only set during registration. Additionally the Moodle account data field surname can be locked ("Local value", always or only if empty) preventing the user from changing it. |

***Table 2: Configuration of the STORK authentication plug-in***

Note that data mapping fields being used to transfer eID attributes have to match the attribute mappings configured for the eID Connector (section 3.2.4).

More data mapping fields than actually needed for SaferChat purposes are supported by the STORK authentication plug-in. This was just done for future use.

### 3.1.2 Configuring of Moodle's cleanup service

A standard Moodle instance automatically invokes a periodic cleanup service. This cleanup service processes RSS feeds, sends notification emails, creates statistical reports and finally removes user accounts that have not been confirmed or that are incomplete.

When creating a user account Moodle sends a confirmation email to the email address provided by the user. This email contains a confirmation link the user has to click within a certain period (normally 7 days). After that an account is regarded as "confirmed".

A user account is regarded "incomplete" when no nickname, no family name or no email address has been provided. Since SaferChat is a service that tries to minimize the user of personal user data, family names are not requested and therefore not stored. If the cleanup service for incomplete user accounts would be activated, Moodle would remove STORK accounts (due to the fact that there are no family names available).

**Therefore it is absolutely necessary to assure that cleanup is disabled for incomplete users**

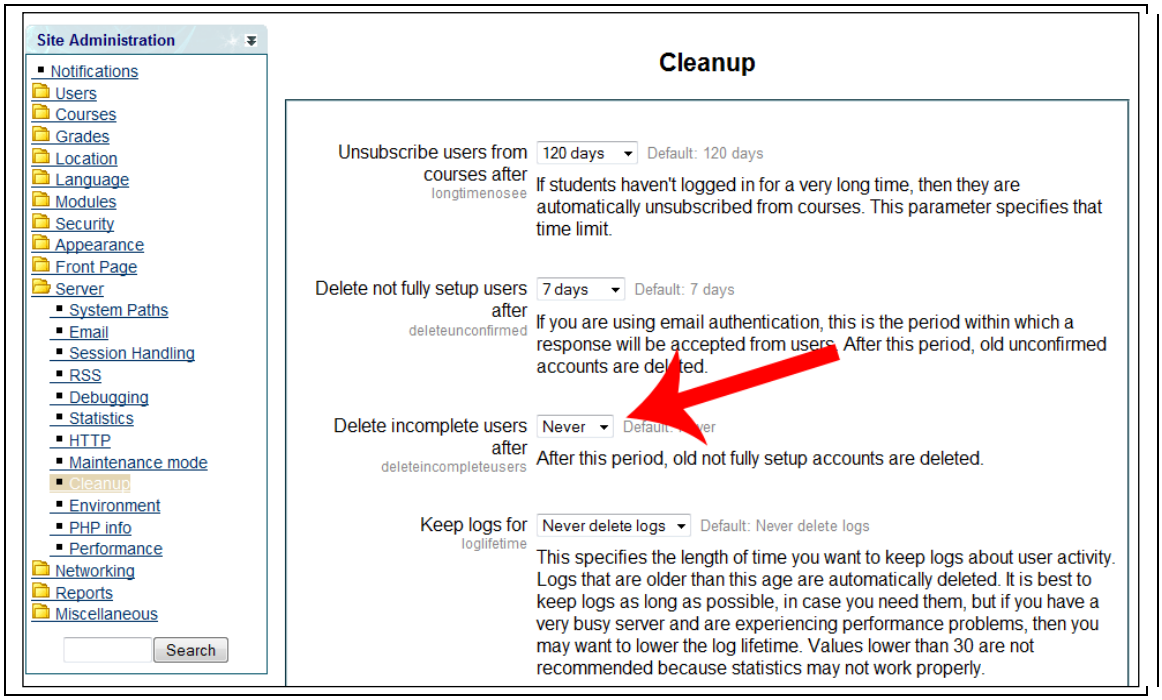


*Figure 5: Configuration of the Moodle Cleanup service*

### 3.1.3 Configuring Moodle chat activities

For setting up a SaferChat room create or edit an arbitrary Moodle course. Refer to the Online Moodle documentation [6] for information on how to create or edit courses.

The next step is to add the activity "SaferChat" (*Figure 6*).

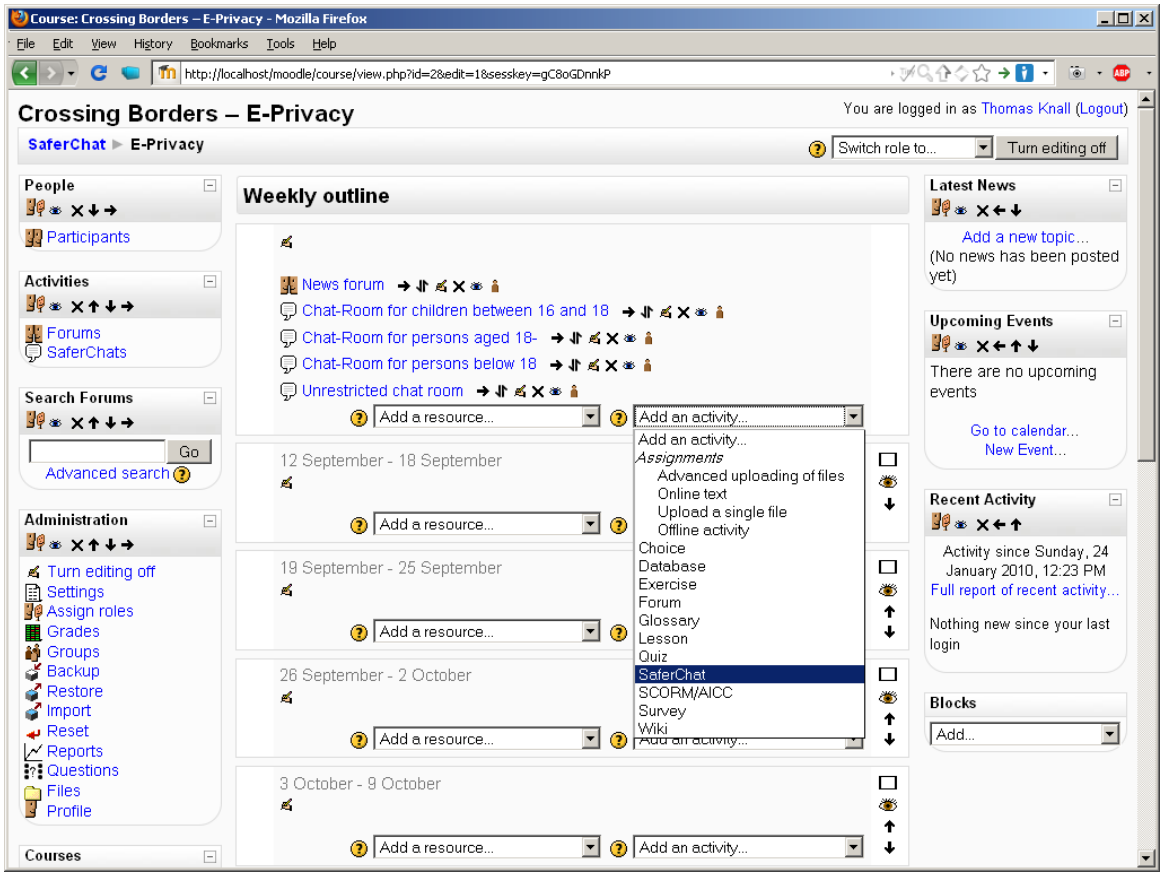


***Figure 6: Configuration interface of the STORK authentication plug-in***

Once a SaferChat activity has been set, it can be individually configured by clicking on the update icon (✍) next to the SaferChat activity.

A configuration interface like shown in *Figure 7* is displayed. Regarding the configuration keys of the default Moodle Chat module three new keys have been added for SaferChat purposes:

The *Minimum required age* defines the minimum age that is needed to be allowed to enter this chat room. The *Maximum allowed age* defines the upper age limit. Persons older than this limit are not allowed to enter this chat room. Each configuration entry can be disabled by un-checking the checkbox next to the entry. Disabling the respective limit means no certain limit. To configure a SaferChat activity for persons not older than 18 the *Minimum required age* entry has to be disabled and the *Maximum allowed age* has to be set to 18.

The third STORK specific configuration entry is the *Minimum QAA level*. In accordance with [11] the entry defines the minimum quality of the authentication method required to be allowed to enter the specific SaferChat.

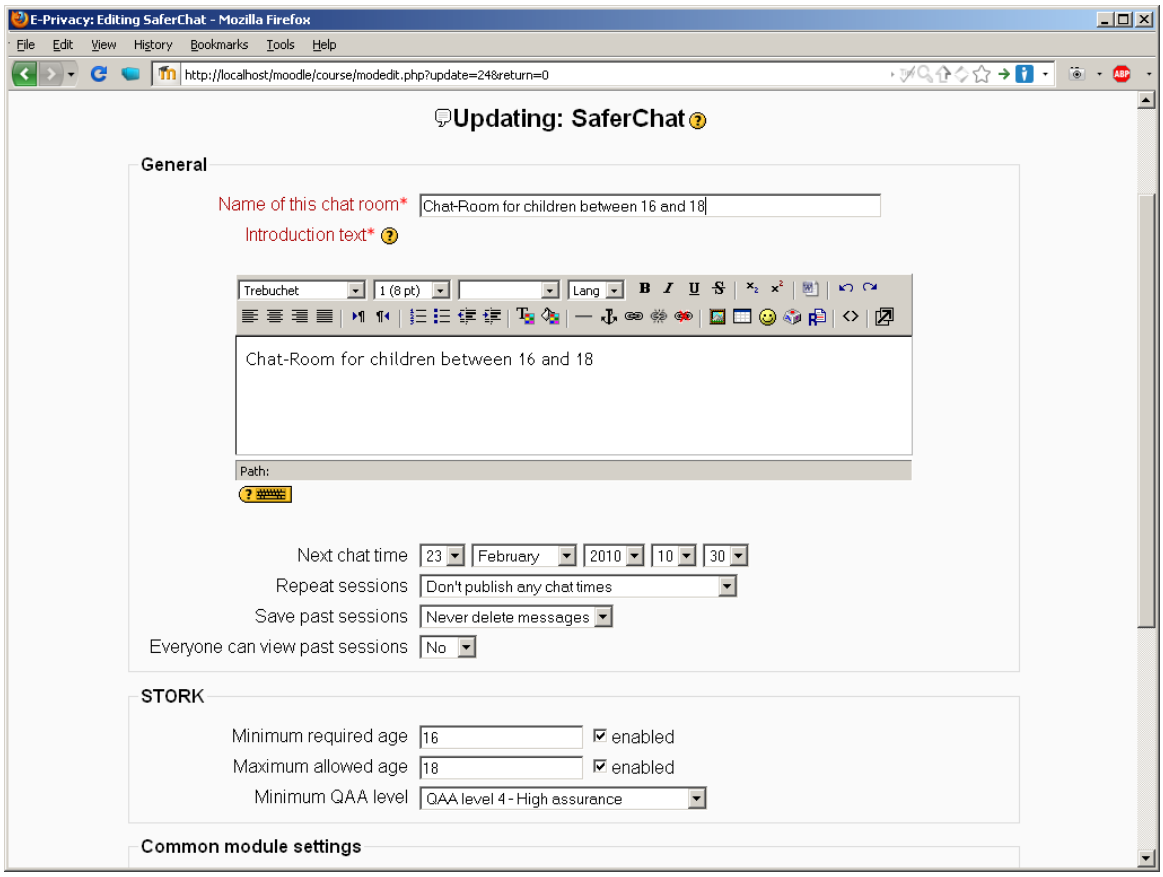


***Figure 7: Configuring a SaferChat activity***

## 3.2 Configuring the eID Connector

The eID Connector has been designed, developed and tested to be run within an Apache Tomcat 5.x or 6.x Servlet container. The deployment package contains an instance of Apache Tomcat 6.0.20 which has already been pre-configured.

The pre-configured ports are:

| Name | Port |
|---|---|
| SHUTDOWN | `18005` |
| HTTP | `18080` |
| HTTPS (using a test SSL certificate) | `18443` |
| AJP | disabled |

***Table 3: Pre-configured ports of the sample Tomcat instance***

Refer to [4] for information on how to configure the Servlet container. For productive use it is highly recommended to put a web server in front of the Servlet container. This requires the AJP port to be enabled. In that case the HTTPS port can be disabled. Additionally the Tomcat specific ports should be restricted to `localhost` usage (`address="127.0.0.1"`) for security reasons.

The connector's configuration file is located at:

`sample/tomcat-6.0.20-moodle-connector/conf/moodle-eid-connector/application_config.xml`

The configuration is structured as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<properties>

  <category name="auth">

    <!--
      <category name="UNIQUE ISO 3166-1 alpha-2 COUNTRY CODE or 'STORK' ">
        <startauthentication.url>
          URL THE USER IS REDIRECTED TO FOR AUTHENTICATION
        </startauthentication.url>
        <authentication.provider.impl>
          AN IMPLEMENTATION OF at.gv.egiz.moodle.eidconnector.auth.AuthenticationProvider
        </authentication.provider.impl>
        <enabled>true TO ENABLE, false TO DISABLE AUTHENTICATION PROVIDER</enabled>
      </category>
    -->

    <!--
      Authentication Provider for Austria
    -->
    <category name="AT">

      <startauthentication.url>
        https://www.stork-kids.at/moa.stork.web/STORKStartAuthentication?SPID=SP-
        AT&OA=https://www.stork-kids.at/moodle-eid-
        connector/connector.do%3Faction=authenticate&QAALevel=4
      </startauthentication.url>

      <authentication.provider.impl>
        at.gv.egiz.moodle.eidconnector.auth.impl.at. AustrianAuthProviderImpl
      </authentication.provider.impl>

      <enabled>true</enabled>

    </category>

    <!--
      Authentication Provider for Iceland
    -->
    <category name="IS">

      <authentication.provider.impl>
        at.gv.egiz.moodle.eidconnector.auth.impl.PEPSAuthenticationProviderImpl
      </authentication.provider.impl>

      <enabled>false</enabled>

    </category>

  </category>

  <!--  Moodle configuration -->
  <category name="moodle">

    <!-- The service that receives the ticket after a successful authentication. -->
    <moodle.ticket.consumer.service>
      https://www.stork-kids.at/moodle19/auth/stork/index.php
    </moodle.ticket.consumer.service>

    <!-- The http header containing the Moodle user name of the authenticated user. -->
    <moodle.httpheader.username>X-STORK-Moodle-username</moodle.httpheader.username>

    <!-- URL the user is redirected to if registration is aborted. -->
    <moodle.start.page>
      https://www.stork-kids.at/moodle19/login/index.php
    </moodle.start.page>
```

```
  <!-- Types of Moodle accounts allowed to be used for the eID connector. -->
  <moodle.allow.auth.types>manual, stork</moodle.allow.auth.types>

  <!--
    The salt Moodle might be using.
    To define multiple salt values use the comma (,) as delimiter.
  -->
  <moodle.password.salt>
    The quick brown fox jumps over the lazy dog,
    Lorem ipsum dolor sit amet
  </moodle.password.salt>

  <!-- Legacy support for old passwords. -->
  <moodle.password.oldcharset>ISO-8859-1</moodle.password.oldcharset>

  <!-- Allow existing Moodle accounts to be converted to STORK accounts. -->
  <moodle.existing.accounts>true</moodle.existing.accounts>

</category>

<category name="internal">

  <!--
    If set to true, exceptions thrown by an authentication implementations are
    shown otherwise skipped.
  -->
  <show.authenticator.errors>true</show.authenticator.errors>

  <!--
    If set to true, no authentication is possible. An appropriate message is shown.
  -->
  <maintenancemode>false</maintenancemode>

  <!-- Statistical logging -->
  <!-- (must implement at.gv.egiz.moodle.eidconnector.logging.StatisticLogger) -->
  <logger.impl>
    at.gv.egiz.moodle.eidconnector.logging.impl.CSVStatisticLogger
  </logger.impl>

  <!-- If set to true, logging into database is enabled. -->
  <!-- In that case the implementation
       at.gv.egiz.moodle.eidconnector.logging.impl.DatabaseStatisticLogger
       has to be used. -->
  <logger.database.enabled>false</logger.database.enabled>

</category>

<!--
  Mapping of STORK attributes to HTTP Headers. (These are the attributes that are sent
  to the auth plugin. Sub-categories may be arbitrarily named.
-->
<category name="eid-http-header-mappings">

  <!-- not needed since eID connector returns Moodle username
  <category name="eid">
    <eid-attribute>http://www.stork.gov.eu/1.0/eIdentifier</eid-attribute>
    <http-header>X-STORK-eidentifier</http-header>
  </category>
  -->

  <!--
    only needed if we want to prefill Moodle's account creation form or if we want the
    STORK moodle auth plug-in to update the given name on each login (if configured).
  -->
  <category name="givenName">
    <eid-attribute>http://www.stork.gov.eu/1.0/givenName</eid-attribute>
    <http-header>X-STORK-givenName</http-header>
  </category>

  <!--
    only needed if we want to prefill Moodle's account creation form or if we want the
    STORK moodle auth plug-in to update the surname on each login (if configured).
  -->
  <category name="surname">
    <eid-attribute>http://www.stork.gov.eu/1.0/surname</eid-attribute>
    <http-header>X-STORK-surname</http-header>
```

```
  </category>

  <!-- needed for SaferChat (mandatory) -->
  <category name="age">
    <eid-attribute>http://www.stork.gov.eu/1.0/age</eid-attribute>
    <http-header>X-STORK-age</http-header>
  </category>

  <!-- needed for SaferChat -->
  <category name="qaaLevel">
    <eid-attribute>http://www.stork.gov.eu/1.0/citizenQAAlevel</eid-attribute>
    <http-header>X-STORK-qaaLevel</http-header>
  </category>

</category>


<!-- e-mail address shown in case of an internal error. -->
<category name="error">
  <mailto>changeme@moodle-server.xyz </mailto>
</category>

<!-- database configuration -->
<category name="hibernate">

  <!-- database configuration for eId connector (full rights required) -->
  <!-- table is automatically created -->
  <category name="eid-connector">

    <!-- mysql -->
    <hibernate.connection.driver_class>
      com.mysql.jdbc.Driver
    </hibernate.connection.driver_class>
    <hibernate.connection.url>
      jdbc:mysql://localhost/moodleeidconnector
    </hibernate.connection.url>
    <hibernate.connection.username>eidconnector</hibernate.connection.username>
    <hibernate.connection.password>eidconnector</hibernate.connection.password>
    <!--
      allowed dialects:
        at.iaik.commons.helper.hibernate.MySQLInnoDBDialectUTF8
        at.iaik.commons.helper.hibernate.MySQLMyISAMDialectUTF8
    -->
    <hibernate.dialect>
      at.iaik.commons.helper.hibernate.MySQLMyISAMDialectUTF8
    </hibernate.dialect>

  </category>

  <!--
    connection to the moodle database (user needs to have the following rights
    for table mdl_user: TABLE_SELECT, TABLE_UPDATE)
  -->
  <category name="moodle">

    <!-- mysql -->
    <hibernate.connection.driver_class>
      com.mysql.jdbc.Driver
    </hibernate.connection.driver_class>
    <hibernate.connection.url>
      jdbc:mysql://localhost/moodle
    </hibernate.connection.url>
    <hibernate.connection.username>eidconnector</hibernate.connection.username>
    <hibernate.connection.password>eidconnector</hibernate.connection.password>
    <!--
      allowed dialects:
        at.iaik.commons.helper.hibernate.MySQLInnoDBDialectUTF8
        at.iaik.commons.helper.hibernate.MySQLMyISAMDialectUTF8
    -->
    <hibernate.dialect>
      at.iaik.commons.helper.hibernate.MySQLMyISAMDialectUTF8
    </hibernate.dialect>

</category>
```

```
  <!--
    Configuration for PEPS-Connection (can be ignored of instance of eID Connector
    is located in Austria)
  -->
  <category name="STORKResponseValidation">

    <!-- comma separated list of trusted issuers -->
    <trusted.issuers>
      http://C-PEPS.gov.xx
    </trusted.issuers>

    <!-- audience name (has to match the issuer name of the authentication request) -->
    <expected.audience>http://S-PEPS.gov.xx</expected.audience>

    <!-- path to signature verification configuration -->
    <signature-verification.config-path>
      ${catalina.base}/conf/moodle-eid-connector/moa-spss/MOA-SPSSConfiguration.xml
    </signature-verification.config-path>

    <!-- profile (trusted certificate chains) for signature verification -->
    <signature-verification.profile>PEPS</signature-verification.profile>

  </category>


  <!--
    The remaining configuration is specific for the Austrian implementation of
    at.gv.egiz.moodle.eidconnector.auth.AuthenticationProvider
    This configuration should remain "as is" and can be ignored.
  -->
  [...]

</properties>
```

### 3.2.1 Category "auth"

Within this category authentication specific implementations can be declared. (An authentication implementation that communicates with the national PEPS or the national V-IDP for instance.)

In order to declare a certain authentication implementation a sub-category has to be created using an ISO 3166-1 alpha-2 [8] conforming country code as name. (e.g. AT, IS, BE…). An arbitrary number of sub-categories (provided that the used name values are unique) can be configured.

For example:

```
  <category name="AT">

    <startauthentication.url>
      https://www.stork-kids.at/moa-id-auth/
    </startauthentication.url>

    <authentication.provider.impl>
      at.gv.egiz.moodle.eidconnector.auth.impl.at. AustrianAuthProviderImpl
    </authentication.provider.impl>

    <enabled>true</enabled>

  </category>


  <category name="IS">

    <authentication.provider.impl>
      at.gv.egiz.moodle.eidconnector.auth.impl.is.IcelandicAuthProvider
    </authentication.provider.impl>

    <enabled>true</enabled>

  </category>
```

Please note that there is already a generic PEPS authentication implementation available that retrieves the PEPS response, validates it and parses the fields needed for SaferChat.

In order to use that implementation select the following authentication provider instance:

```
<authentication.provider.impl>
  at.gv.egiz.moodle.eidconnector.auth.impl. PEPSAuthenticationProviderImpl
</authentication.provider.impl>
```

| Configuration key | Example | Description |
|---|---|---|
| startauthentication.url | `https://www.stork-kids.at/moa.stork.web/STORKStartAuthentication?SPID=SP-AT&OA=https://www.stork-kids.at/moodle-eid-connector/connector.do%3Faction=authenticate&QAALevel=4` | OPTIONAL: This key can be used to define a service the user is redirected to if the URL `https://<SERVICE-PROVIDER>/moodle-eid-connector/connector.do?action=startAuthentication&country=<COUNTRY_CODE>`[4] is invoked. This is just a redirection service created for convenience reasons. |
| authentication.provider.impl | `at.gv.egiz.moodle.eidconnector.auth.impl.at. AustrianAuthProviderImpl` | Using this configuration key, an authentication implementation can be declared. Refer to section 3.4 for details on how to create and set up a specific authentication implementation. |
| enabled | `true` | If set to `false` the declared authentication implementation will be ignored, if set to `true` the implementation will be used. |

***Table 4: eID Connector – Configuration of Category "auth"***

### 3.2.2 Category "moodle"

This section covers configuration details regarding Moodle.

| Configuration key | Example | Description |
|---|---|---|
| moodle.ticket.consumer.service | `https://www.stork-kids.at/moodle19/auth/stork/index.php` | This key defines the URL to the Moodle STORK authentication plug-in. The URL should be like follows: `http[s]://<SERVICE_PROVIDER_HOST>/<PATH_TO_MOODLE>/auth/stork/index.php` |
| moodle.httpheader.username | `X-STORK-Moodle-username` | This key defines the name of the HTTP header that is used to transfer the username of the authenticated user from the eID Connector to the requesting Moodle STORK auth plug-in. It is recommended to keep the default value `X-STORK-Moodle-username`. |
| moodle.start.page | `https://www.stork-kids.at/moodle19/login/index.php` | This defines the page the user is redirected to if authentication has been aborted by the eID connector. |

[4] ISO 3166-1 alpha-2 COUNTRY CODE

| Configuration key | Example | Description |
|---|---|---|
| moodle.allow.auth.types | `manual, stork` | In general each Moodle account is assigned to a certain type of authentication (e.g. “manual”, “cas”). The type of the Moodle STORK authentication plug-in is “stork”.<br>Using this configuration key, a comma-separated list of account types can be defined which can be used for STORK authentication. During a registration process an existing Moodle account (of a type other than “stork”) is converted into a STORK account (type “stork”).<br>Note that future authentication with this account is limited to STORK. |
| moodle.password.salt | `the brown fox jumped over the lazy dog, Lorem ipsum dolor sit amet` | This setting can be used to configure salt values Moodle is using to increase password security ([10]). Multiple salt values have to be comma-separated. |
| moodle.password.old charset | `ISO-8859-1` | In case an old user database is used for STORK Moodle, stored password hashes may not be based on UTF-8 which is the default behaviour of modern Moodle platforms. In order to allow these passwords to be used a “legacy” charset can be defined with this setting. Moodle automatically converts old passwords based on old charsets into UTF-8 based passwords during a successful login. |
| moodle.existing.accounts | `false` | This setting defines if the connector should allow existing accounts to be converted to Moodle account on registration. Disabling this feature (setting to `false`) directly redirects to the Moodle registration dialogue. |

***Table 5: eID Connector – Configuration of Category “moodle”***

### 3.2.3 Category "internal"

This category covers internal configuration of the eID Connector.

| Configuration key | Example | Description |
|---|---|---|
| show.authenticator.errors | `false` | If set `true` errors occurring while invoking single authentication implementations are verbosely shown to the user and consecutively invocation of registered authentication implementations is aborted.<br>If set `false`, errors are not shown. The erroneous invocation implementation is skipped continuing with the next registered implementation. |
| maintenancemode | `false` | If set `true` the eID connector is put into maintenance mode. That means that no authentication requests will be processed. Instead an appropriate message is shown indicating that the service is in maintenance mode. |
| logger.impl | `at.gv.egiz.moodle.eidconnector.logging.impl.CSVStatisticLogger` | This configuration key declares a logging implementation for statistical purposes. A full qualified name of a class implementing `at.gv.egiz.moodle.eidconnector.logging.StatisticLogger` has to be provided.<br>Refer to section 3.5 for further information. |
| logger.database.enabled | `false` | If this configuration key is enabled (→ `true`) a special table is used for logging provided that logger.impl is set to `at.gv.egiz.moodle.eidconnector.logging.impl.DatabaseStatisticLogger`. This logger supports logging into a database.<br>Refer to section 3.5 for further information. |

***Table 6: eID Connector – Configuration of Category "internal"***

### 3.2.4 Category "eid-http-header-mappings"

In this section mappings of STORK eID attributes (as defined in [7], section 7.2 "Subject Attribute Definitions") to HTTP headers, which are used to transfer these attributes from the eID Connector to the Moodle STORK authentication plug-in, can be defined. Note that these HTTP header names have to match the names configured for the STORK auth plug-in (section 3.1.1).

For each new mapping, a separate sub-category has to be created. The sub-category can be arbitrarily named (except for the fact that these names have to be unique).

For example:

```
<category name="givenName">
  <eid-attribute>http://www.stork.gov.eu/1.0/givenName</eid-attribute>
  <http-header>X-STORK-givenName</http-header>
</category>
```

| Configuration key | Example | Description |
|---|---|---|
| eid-attribute | `http://www.stork.gov.eu/1.0/givenName` | The respective eID attribute name as defined in [7]. |
| http-header | `X-STORK-givenName` | The corresponding HTTP header name that is used to transmit the value of the attribute from the eID Connector to the Moodle STORK authentication plug-in. |

***Table 7: eID Connector – Configuration of Category "eid-http-header-mappings"***

### 3.2.5 Category "error"

This category covers configuration data for the error page which is shown in case of an unexpected internal error.

| Configuration key | Example | Description |
|---|---|---|
| mailto | `admin@myMoodleHost.net` | In order to allow the user to send a message to someone who is responsible to fix the issue the error page provides a link with the email address given by this configuration key. |

***Table 8: eID Connector – Configuration of Category "error"***

### 3.2.6 Category "hibernate"

This category covers the database configuration. In order to support a large amount of different databases the data model has been separated from the underlying database implementation using a framework called "Hibernate[5]".

The eID Connector needs to access two databases:

1. The first one is its own database where mappings from eIdentifiers to Moodle user accounts are stored.
2. The second one is the connection to Moodle's database. This connection is needed to retrieve user information as well as to modify user accounts in order to convert them into STORK user accounts.

For more information on database prerequisites refer to section 2.3.1.

The category "hibernate" contains two sub-categories which both containing the same configuration keys. Category "`eid-connector`" defines the connection to the database of the eID Connector, category "`moodle`" defines the connection to Moodle's database.

Example:

```
<category name="eid-connector">
  <hibernate.connection.driver_class>
    com.mysql.jdbc.Driver
  </hibernate.connection.driver_class>
  <hibernate.connection.url>
    jdbc:mysql://localhost/moodleeidconnector
  </hibernate.connection.url>
  <hibernate.connection.username>eidconnector</hibernate.connection.username>
  <hibernate.connection.password>eidconnector</hibernate.connection.password>
```

5 https://www.hibernate.org/

```
  <hibernate.dialect>
    at.iaik.commons.helper.hibernate.MySQLMyISAMDialectUTF8
  </hibernate.dialect>
</category>

<category name="moodle">
   ...
</category>
```

| Configuration key | Example | Description |
|---|---|---|
| hibernate.connection.driver_class | `com.mysql.jdbc.Driver` | The JDBC driver class. In case of the example a MySQL database driver is referenced. |
| hibernate.connection.url | `jdbc:mysql://localhost/moodleeidconnector` | The JDBC connection URL of to the database. In case of the example the name of the database is `moodleeidconnector`. |
| hibernate.connection.username | `eidconnector` | The username for the JDBC connection. |
| hibernate.connection.password | `eidconnector` | The password for the JDBC connection. |
| hibernate.dialect | `at.iaik.commons.helper.hibernate.MySQLMyISAMDialectUTF8` | The class name of a Hibernate Dialect (implementing `org.hibernate.dialect.Dialect`) which allows Hibernate to optimize database connections suitable for the underlying type of database.<br>In order to enforce the database to support UTF-8 two own dialect implementations (one for a MySQL-MyISAM and one for a MySQL-InnoDB database) have been created:<br>`at.iaik.commons.helper.hibernate.MySQLMyISAMDialectUTF8`<br>`at.iaik.commons.helper.hibernate.MySQLInnoDBDialectUTF8`<br>It is recommended to select one of these dialects according to the given MySQL database configuration. |

***Table 9: eID Connector – Configuration of Category "hibernate"***

### 3.2.7 Category "STORKResponseValidation"

This category covers configuration settings for PEPS response validation. Note that these settings are only relevant if the generic PEPS authentication provider (refer to section 3.2.1) `at.gv.egiz.moodle.eidconnector.auth.impl.PEPSAuthenticationProviderImpl` is used.

Example:

```
<category name="STORKResponseValidation">

  <trusted.issuers>
    http://C-PEPS1.gov.xx,
    http://C-PEPS2.gov.xx
  </trusted.issuers>

  <expected.audience>http://S-PEPS.gov.xx</expected.audience>

  <signature-verification.config-path>
    ${catalina.base}/conf/moodle-eid-connector/moa-spss/MOA-SPSSConfiguration.xml
  </signature-verification.config-path>

  <signature-verification.profile>PEPS</signature-verification.profile>

</category>
```

| Configuration key | Example | Description |
|---|---|---|
| trusted.issuers | http://C-PEPS.gov.xx | A comma separated list of trusted issues of STORK assertions. Assertions from issuers that are not in the list are rejected. |
| expected.audience | http://S-PEPS.gov.xx | This setting defines the expected receiver of the assertion. The assertion is rejected if its attribute "Destination" does not match the `expected.audience` setting. |
| signature-verification.config-path | `${catalina.base}/conf/moodle-eid-connector/moa-spss/MOA-SPSSConfiguration.xml` | This defines the path to configuration for the signature verification framework MOA-SP. Signed PEPS responses are being verified (including certificate trust path validation) using MOA-SP. Note that the path can be set using system properties (like `catalina.base` for instance).<br>Refer to section 3.3 for more detailed information. |
| signature-verification.profile | `PEPS` | This setting defines the trust profile that should be used for signature verification/certificate validation of PEPS responses. Refer to section 3.3 for more detailed information. |

***Table 10: eID Connector – Configuration of Category "STORKResponseValidation"***

## 3.3 Configuring signature verification and trust

The configuration file `MOA-SPSSConfiguration.xml` for signature verification (which is used by the generic PEPS authentication provider `at.gv.egiz.moodle.eidconnector.auth.impl.PEPSAuthenticationProviderImpl`) is located in `deployment/tomcat/conf/moodle-eid-connector/moa-spss` of the deployment package. This file does not have to be modified since it only contains a reference to a folder containing certificates that depict trust anchors of trusted certificate chains.

In order to define trusted certificate chains like

e.g. (1) root-certificate
(2) intermediate certificate #1
(3) intermediate certificate #2
(4) end user certificate

with the root certificate (1) as trust anchor the following steps have to be conducted:

1. Copy the certificate you want to set as trust anchor (e.g. the root certificate) to `<TOMCAT>/conf/moodle-eid-connector/moa-spss/trustProfiles/PEPS`
2. Copy each of the certificates between trust anchor (including trust anchor!) and end user certificate (certificates (1), (2) and (3) in the above mentioned example) to `<TOMCAT>/conf/moodle-eid-connector/moa-spss/certstore/toBeAdded`. Create the `toBeAdded` if it does not exist.
3. Restart the tomcat instance.

Note that revocation check is done for each of the certificates of the chain except for the trust anchor. In the above mentioned example revocation check for (4), (3) and (2) is performed.

## 3.4 Connecting to a proprietary authentication service

Section 3.2.1 describes the configuration of the part of the eID Connector that is responsible for authentication. The architecture of the eID Connector allows (nearly) arbitrary authentication services to be used. In terms of STORK these services can be PEPS, V-IDP or any other national authentication solution.

In general authentication is intended to run like follows:

1. The user is redirected to a certain authentication implementation (e.g. PEPS). This can be achieved by a link (the user has to click) placed at Moodle's login page or this can be done using the mechanism (regarding the configuration key `startauthentication.url`) described in section 3.2.1.
2. The authentication is performed by the external authentication service, PEPS or V-IDP.
3. On successful authentication the user has to be redirected to `https://<SERVICE-PROVIDER>/moodle-eid-connector/connector.do?action=authenticate`.
   If necessary arbitrary parameters can be passed (for instance `…/connector.do?action=authenticate&SAMLArtifact=anVzdCBhIGR1bW15IGFydGlmYWN0`)
4. The eID Connector automatically iterates over all registered authentication implementations trying to find a certain one which declares itself responsible for that type of request (e.g. by examining parameters that have been passed as additional parameters and returning `true` on invocation of method `isResponsible(…)`; see below).
5. If an authentication implementation has been found that claims to be responsible for that request the method `authenticate(…)` is automatically invoked (see below). This method has to do whatever is needed to get credentials from the respective authentication service (e.g. following a SAML Browser/Artifact Profile).
6. These credentials have to be put into a Java `SubjectAttributes`[6] object. This object was designed to store STORK related attributes (refer to the Java API documentation included

[6] Which is part of the Moodle eID Connector.

in the deployment package). Refer to [7], section 7.2 "Subject Attribute Definition" for specification of these attributes.

7. The eID connector automatically evaluates these attributes and does some calculations if needed (e.g. calculates age if only date of birth was provided…), performs user interaction in case of a new user registration and finally transmits the credentials to the Moodle STORK authentication plug-in.

### Setting up a connection to a authentication service

In order to connect to a certain authentication service the Java interface `at.gv.egiz.moodle.eidconnector.auth.AuthenticationProvider` with two methods has to be implemented:

**`boolean isResponsible(HttpServletRequest request)`**

The authentication implementation has to return `true` for this method if it determines to be responsible for that request. (e.g. if the request contains certain parameters…).

**`SubjectAttributes authenticate(HttpServletRequest request)`**

If the above described method `isResponsible(…)` returned `true`, the method `authenticate(…)` is automatically invoked by the eID Connector. This method has to do whatever is needed to get credentials from the respective authentication service (e.g. following a SAML Browser/Artifact Profile). These credentials have to be put into a Java `SubjectAttributes` object (refer to the Java API documentation included in the deployment package). This object was designed to store STORK related attributes. Refer to [7], section 7.2 "Subject Attribute Definition" for specification of these attributes.

The following listing shows the Interface that has to be implemented:

```
package at.gv.egiz.moodle.eidconnector.auth;

import javax.servlet.http.HttpServletRequest;

import at.gv.egiz.moodle.eidconnector.EIDConnectorException;
import at.gv.egiz.moodle.eidconnector.auth.data.SubjectAttributes;

public interface AuthenticationProvider {

   /**
    * Returns {@code true} if the underlying authentication implementation claims
    * to be responsible for the given {@link HttpServletRequest}.
    * @param request The {@link HttpServletRequest}.
    * @return {@code true} if responsible, {@code false} if not.
    * @throws EIDConnectorException Thrown in case of an error.
    */
   boolean isResponsible(HttpServletRequest request) throws EIDConnectorException;

   /**
    * Uses the given request and retrieves/derives credentials from the
    * respective authentication service.
    * @param request The {@link HttpServletRequest}.
    * @return Credentials as {@link SubjectAttributes}.
    * @throws EIDConnectorException Thrown in case of an error.
    */
   SubjectAttributes authenticate(HttpServletRequest request) throws EIDConnectorException;

}
```

## 3.5 Logging

The Moodle eID Connector additionally provides means for statistical logging (how many unique users logged within a certain timeframe? age ranges? national logins or stork logins...?).

The make the solution as flexible as possible a simple interface was created, allowing to use own logging implementations:

```
package at.gv.egiz.moodle.eidconnector.logging;

import java.util.Date;

public interface StatisticLogger {

   /**
    * Creates a log entry using the data provided.
    * Note that timestamp, assertionIssuerId and spID must not be {@code null}, while the
    * remaining parameters might be.<br/>
    * After parameter {@code age} an arbitrary number of various user defined parameters
    * might be passed. These additional fields should also be logged if provided.<br/>
    * Values that are {@code null} should be replaced by an empty String:<br/>
    * e.g. Using CVS logger {@code citizenCountryCode == null} and {@code age == null}
    *      results in
    *      {@code 2010-09-07T09:27:20.697Z;VIDP-AT;SaferChat-AT;eFKN39cIZoDjRRZOKhYlTD3dcXE=;;4;}
    * @param timestamp The current date.
    * @param assertionIssuerId The identifier of the underlying S-PEPS/V-IDP instance.
    * @param spID The identifier of the service provider being served.
    * @param sha1OfEID A base64 encoded String of the SHA1 hash of the citizen's eIdentifier.
    * @param citizenCountryCode The citizen's home country code (according to ISO 3166-1 alpha-2).
    * @param citizenQAALevel The underlying QAA level.
    * @param age The citizen's age.
    * @param others An arbitrary number of further fields to be logged.
    */
   void log(Date timestamp, String assertionIssuerId, String spID, String sha1OfEID,
            String citizenCountryCode, Integer citizenQAALevel, Integer age, Object... other);

}
```

There are already several implementations available:

**`at.gv.egiz.moodle.eidconnector.logging.impl.CSVStatisticLogger`**

This implementation uses a CSV syntax for log entries. These entries are logged at DEBUG level using the default logging framework of the eID Connector (slf4j). In order to get CSV files the logging configuration (`logback.xml`) has to be configured in such a way that log from the CSVStatisticLogger are the only ones being logged into the CSV file.

e.g.

```
...
<appender name="STATISTICFILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
   <File>${catalina.base}/logs/statistics.csv</File>
   <encoder>
      <pattern>%m%n</pattern>
   </encoder>
   <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
      <maxIndex>9</maxIndex>
      <FileNamePattern>${catalina.base}/logs/statistics.csv.%i</FileNamePattern>
   </rollingPolicy>
   <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
      <MaxFileSize>10240KB</MaxFileSize>
   </triggeringPolicy>
</appender>
<logger name="at.gv.egiz.moodle.eidconnector.logging.impl.CSVStatisticLoggerImpl"
        level="DEBUG">
   <appender-ref ref="STATISTICFILE"/>
</logger>
...
```

**at.gv.egiz.moodle.eidconnector.logging.impl.DatabaseStatisticLogger**

This implementation uses a table named "statistic_log" for logging. This allows more sophisticated evaluation of log entries.

Important note: This logger implementation requires the configuration key "logger.database.enabled" within category "internal" to be set to true. This entry automatically creates the table (if not already done) and provides database access for the logger.

**at.gv.egiz.moodle.eidconnector.logging.impl.NOPStatisticLogger**

Using this implementation completely disables statistical logging.

**at.gv.egiz.moodle.eidconnector.logging.impl.SimpleStatisticLogger**

This logger just uses standard Java logging, such like

```
[DEBUG@24.09.2010 11:18:03]
at.gv.egiz.moodle.eidconnector.logging.impl.SimpleStatisticLogger:log:67 -
at.gv.egiz.moodle.eidconnector.logging.impl.SimpleStatisticLogger@1204425[timest
amp=Fri Sep 24 11:18:03 CEST
2010,assertionIssuerId=STORK,spID=https://www.stork-
kids.at/moodle19/auth/stork/index.php,sha1OfEID=iEt9+W/ZXpyoU1o/BpxHORVGSkw=,
citizenCountryCode=IS,citizenQAALevel=4,age=18]
```

This logging is more or less only human readable and can hardly be used for machine based analysis.

Note: This logger is used as a fallback logger in case of problems when using other logging implementations.

# 4 Moodle Registration

The eID Connector automatically shows a registration dialogue if the user just being authenticated has not yet been linked to a Moodle account.

There are two ways registration can be performed. One is to create a new Moodle account, the other one is to use an existing account.

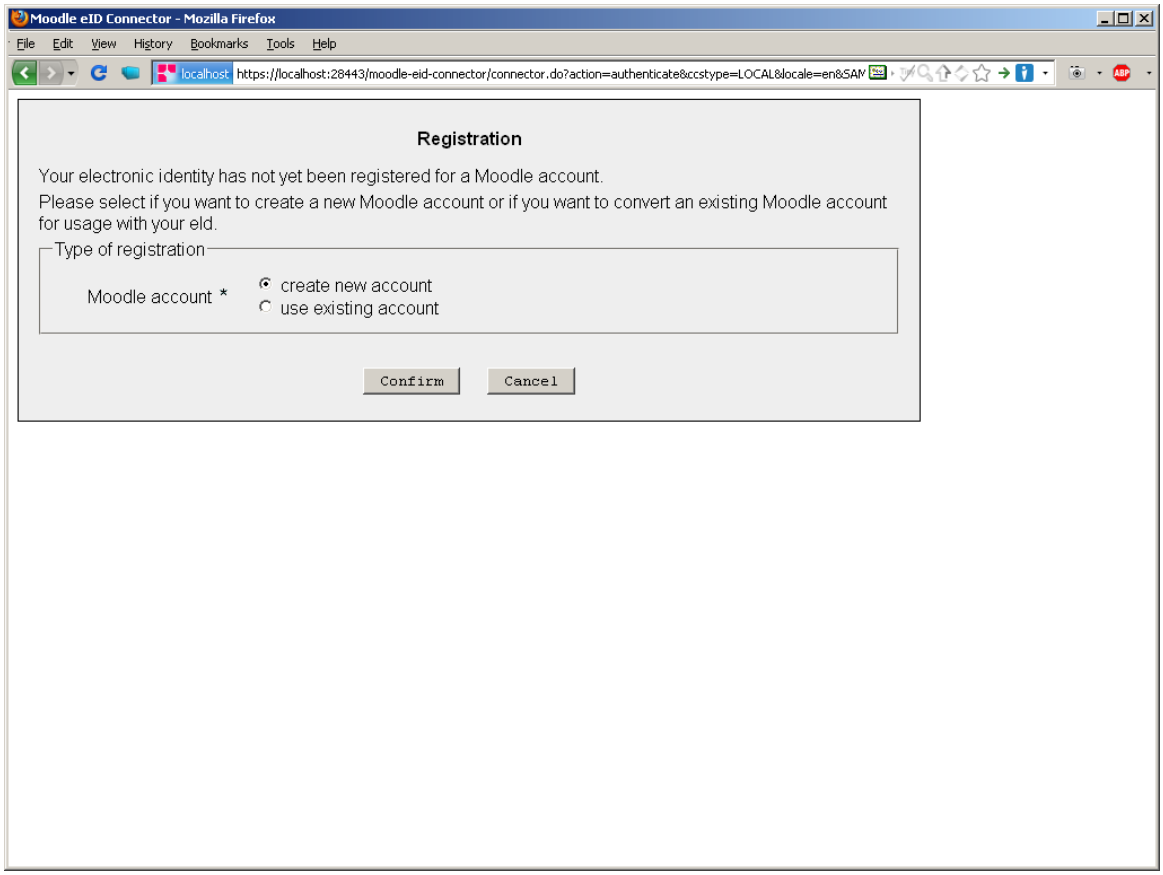


***Figure 8: Registration dialogue***

## 4.1 Creating a new Moodle account

If the user just being authenticated selects to create a new account, a new Moodle account (with a username derived from the eIdentifier (using a one-way hash function) of the type "stork" is created and the user is logged in.

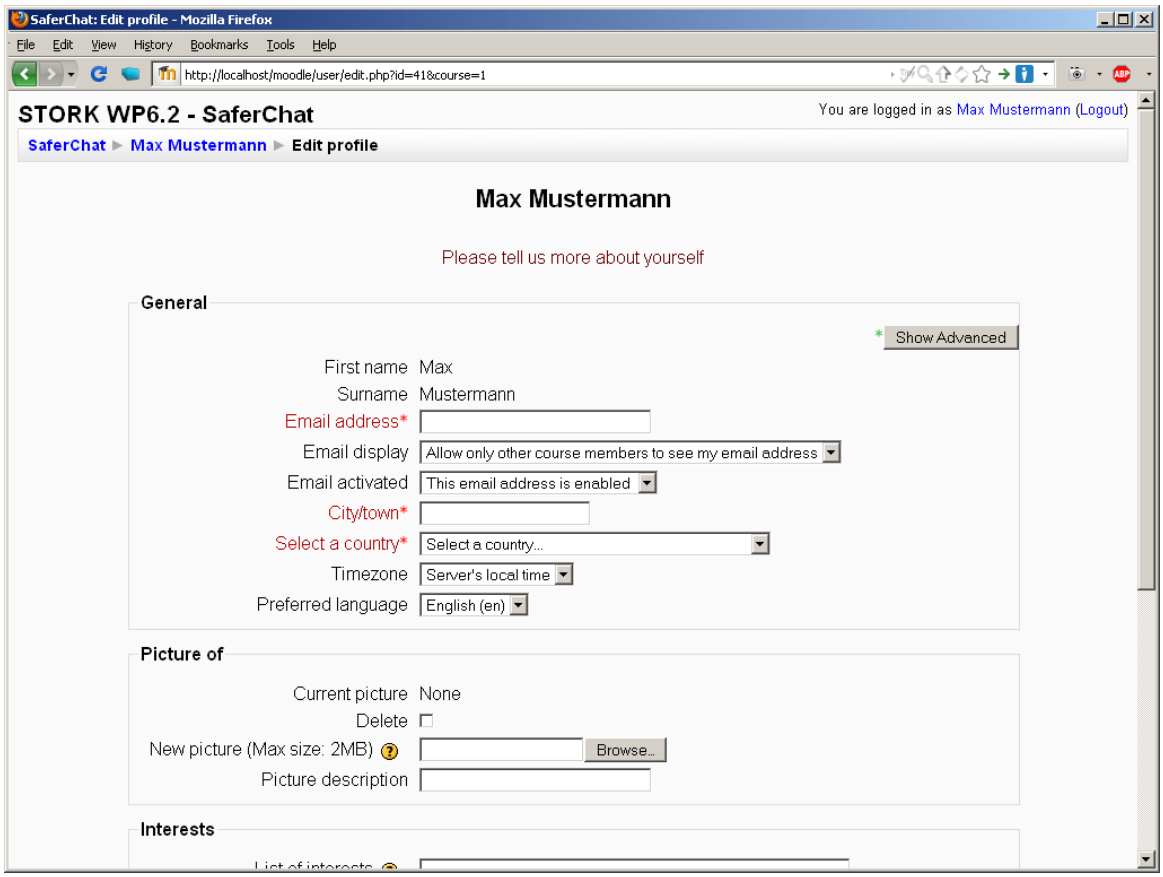


***Figure 9: Initial completion of account data after registration***

After the creation of the new account a Moodle dialogue is shown where the user can enter additional data (similar to a regular Moodle user authentication). Note that the fields of first name and surname are locked since the Moodle authentication plug-in has been configured accordingly (refer to section 3.1.1 for more details on that).

Once the form has been filled out the user is redirected to his personal page (*Figure 10*).

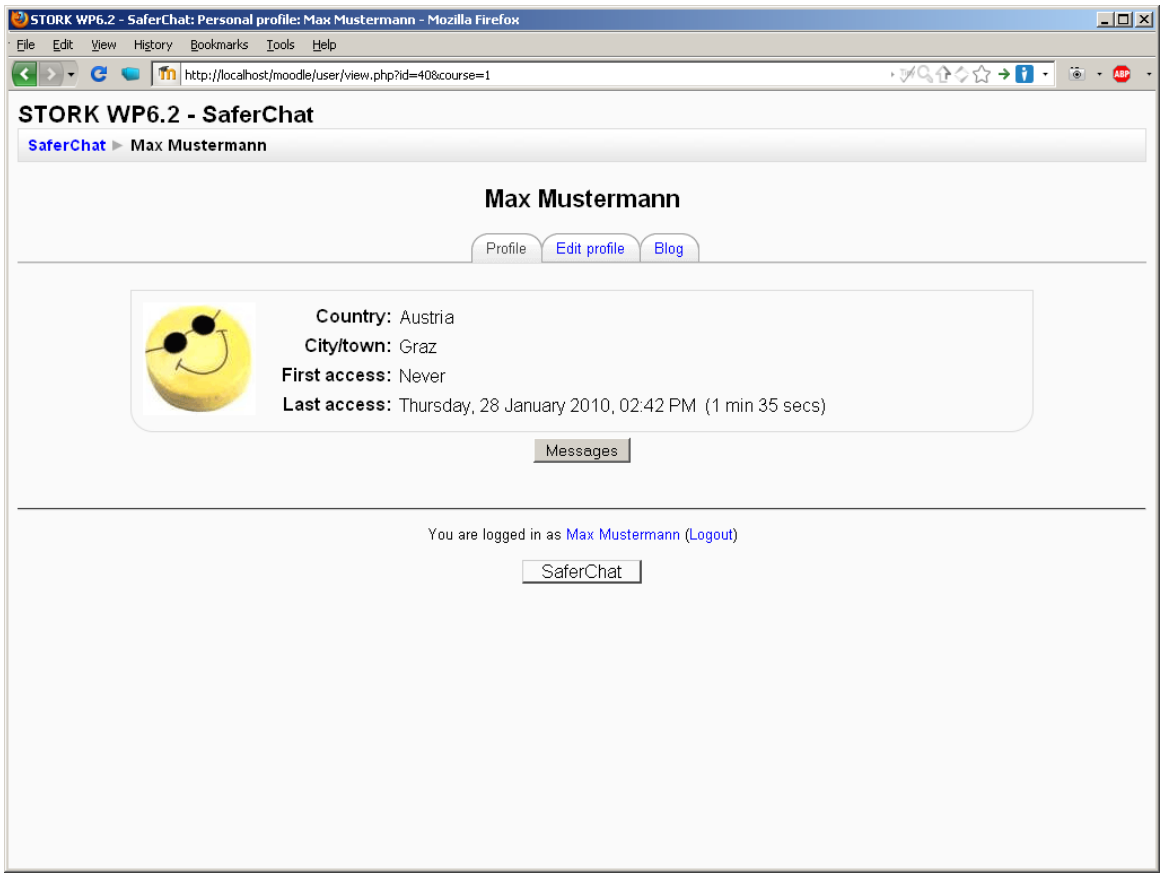


***Figure 10: Successful STORK authentication and registration***

## 4.2 Registering an existing Moodle account

If the user selects to use an existing Moodle account to be linked to his identity an extended registration dialogue is shown requesting the user to enter the username and password of his Moodle account.

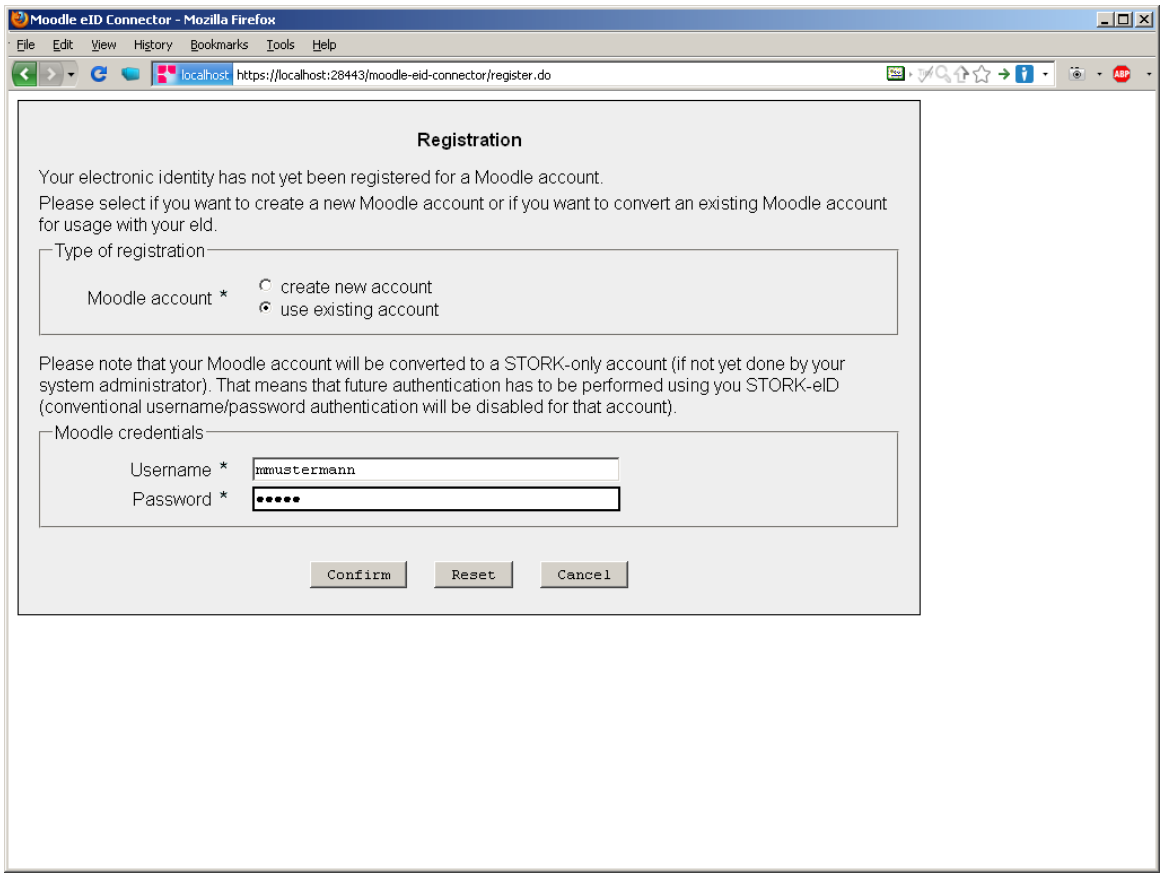


***Figure 11: Registration dialogue – using an existing Moodle account***

The username/password credentials are verified. If successfully verified, the account matching the credentials is converted into a stork account (type "stork) and the user is logged in.

Note that an account once converted into a STORK account cannot be used for conventional username/password authentication. Any future authentication has to be performed using STORK.

# 5 Moodle Authentication

From the user's perspective authentication is done performing the following steps:

1. The user enters the Moodle site. He is not logged in yet so he selects "Login".

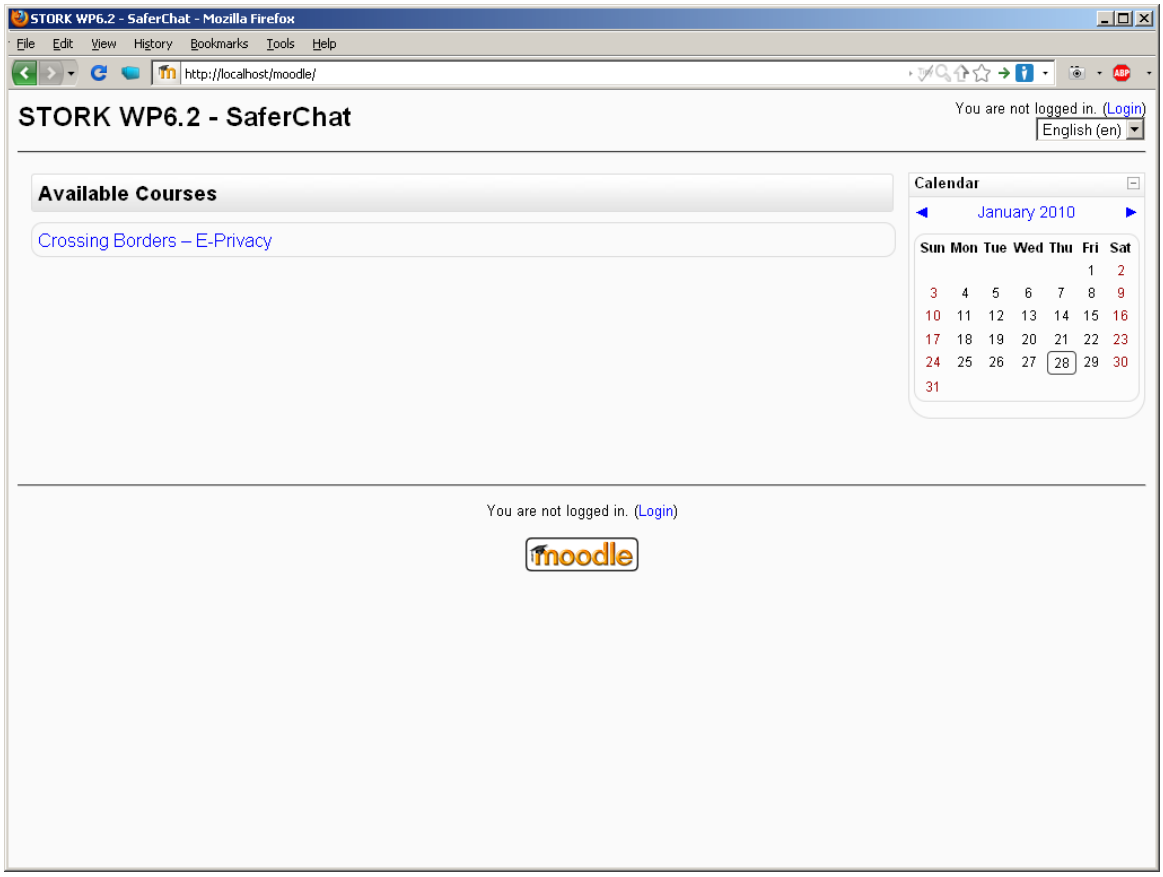


***Figure 12: Exemplary Moodle start page (user not logged in)***

2. The user does not select username/password authentication but “STORK eID Authentication”.


*Figure 13: Exemplary Moodle login page*

3. A list of countries being supported is shown. The user selects his home country.
4. The user is redirected to the authentication service of his country (PEPS) or he is requested to perform authentication using his middleware. *Figure 14* and *Figure 15* show authentication using the Austrian middleware as an example.

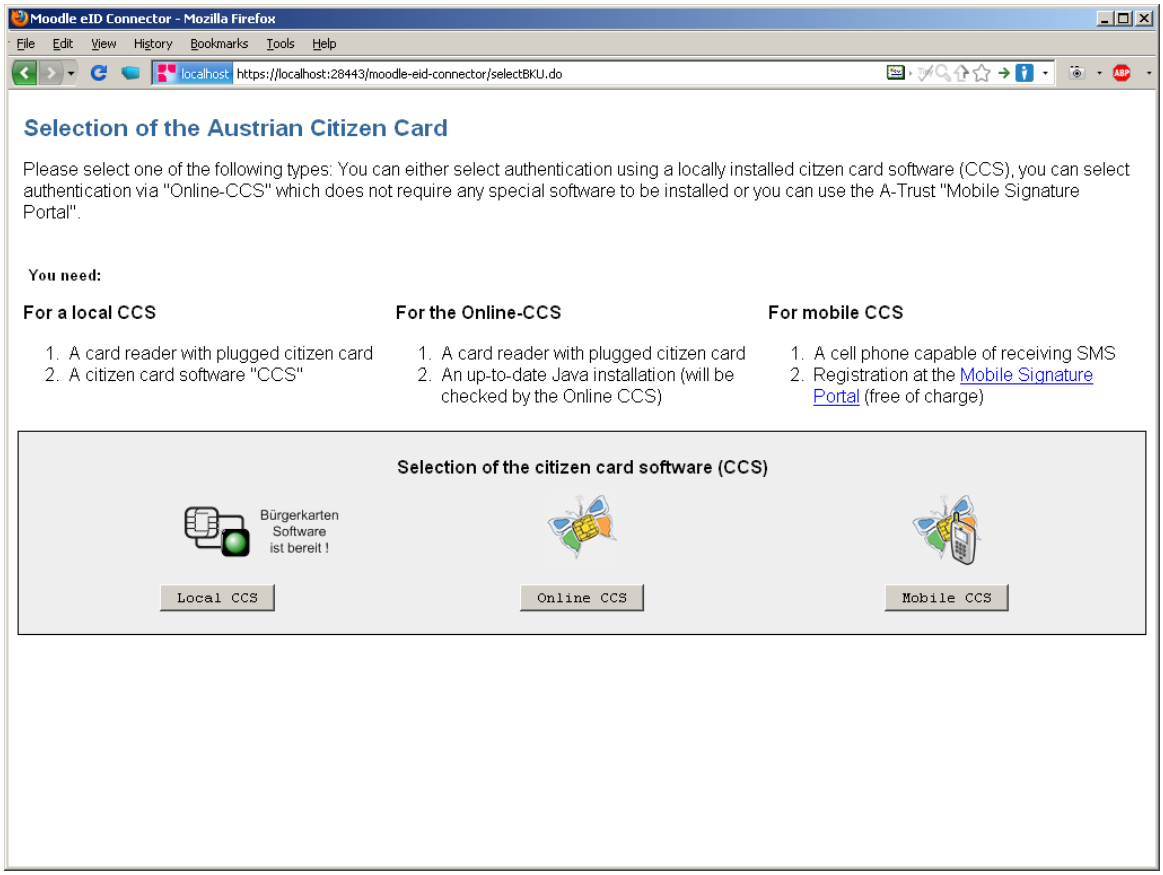


***Figure 14: Selection of Austrian Citizen Card***

5. The user performs authentication.

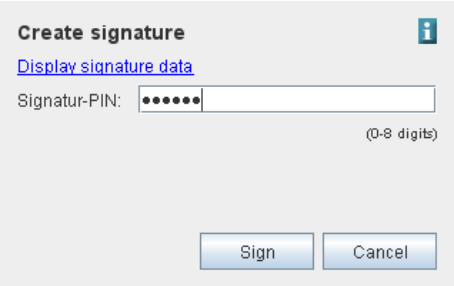


***Figure 15: Authentication using Austrian middleware***

6. Since the user has already been registered (just an assumption in this chapter) he is logged into Moodle. The user's age is temporary stored to his account's data. That data is only valid for the current session.

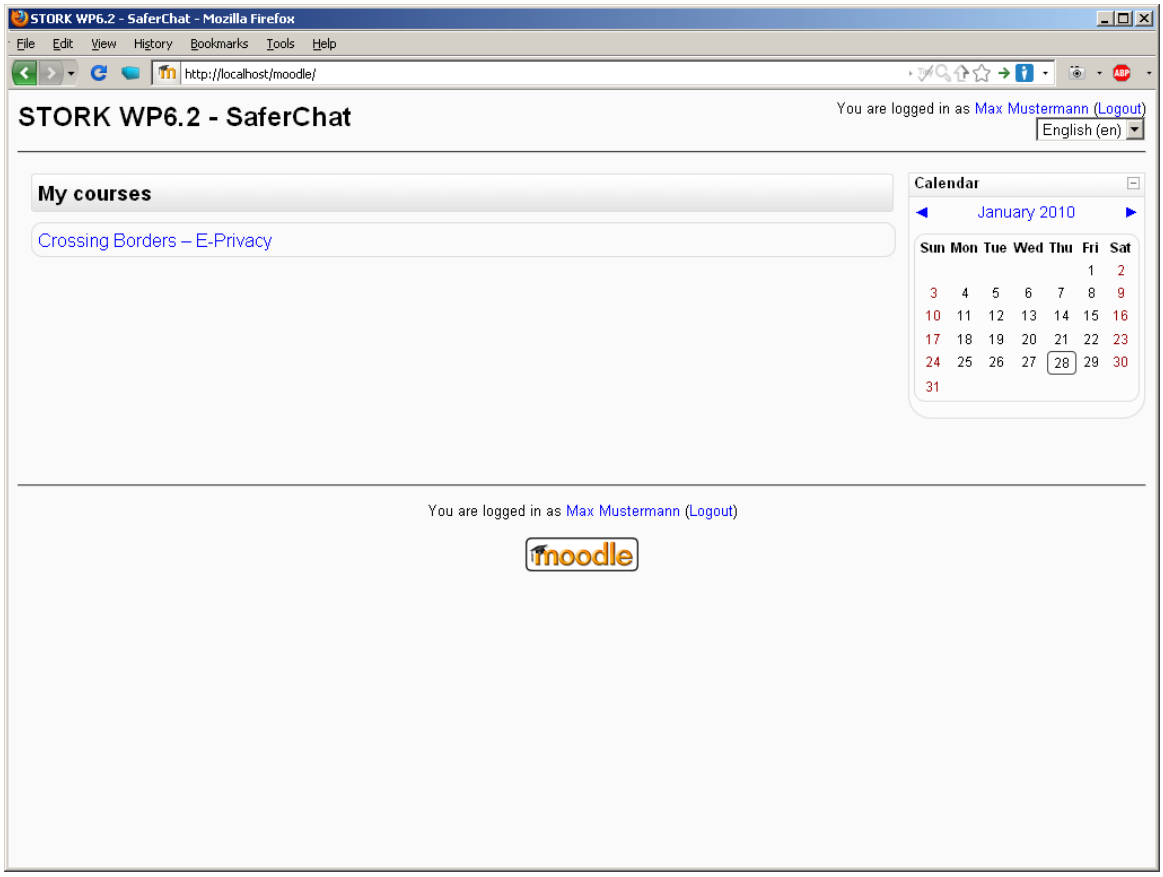


***Figure 16: Successful STORK login***

## 5.1 Entering a SaferChat Activity

Once authenticated the used has access to courses. Within a single course an arbitrary number of SaferChat activities can be set by the course creator. *Figure 17* shows a sample course with four different chat rooms.

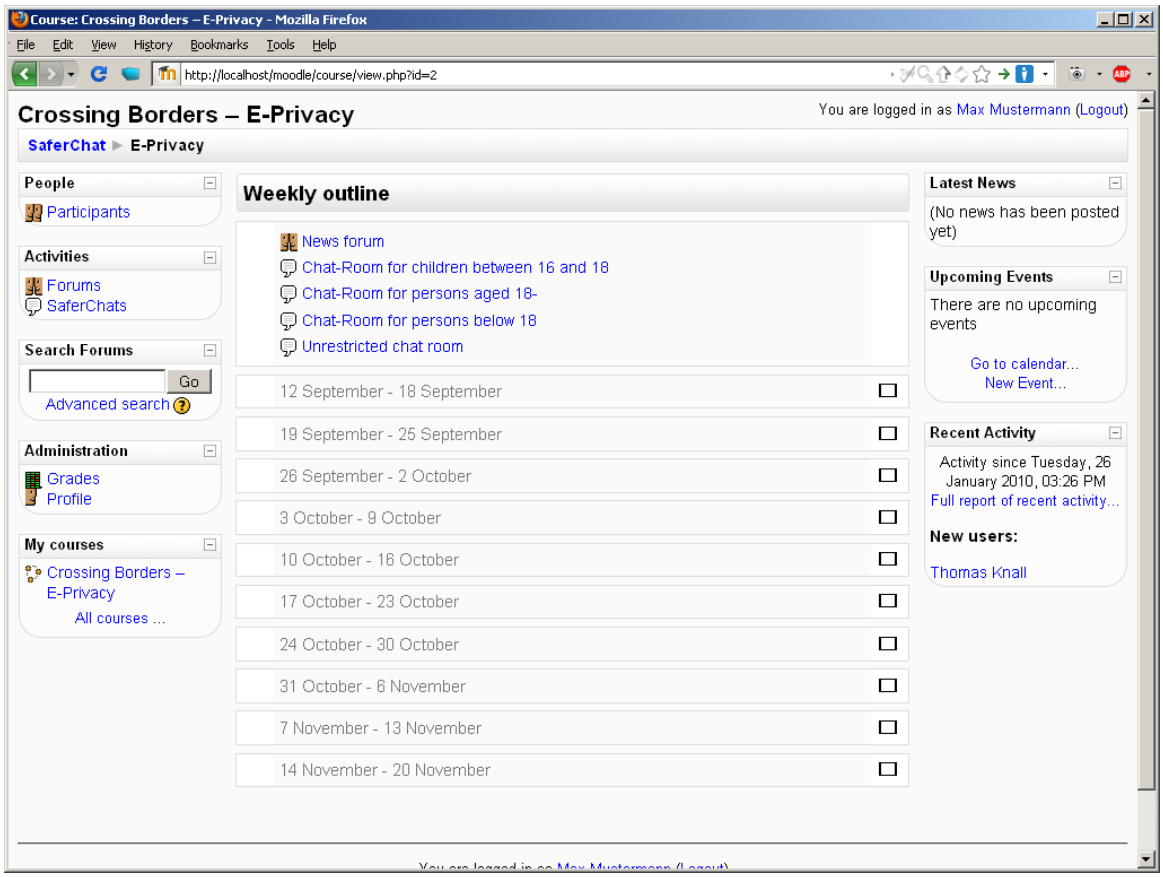


***Figure 17: Selecting a chat room of a course***

The user selects a SaferChat room. If his age does not match the age restrictions (if configured) for this chat room or the used QAA level was insufficient (if configured) access if denied (*Figure 18*) otherwise granted (*Figure 19*).

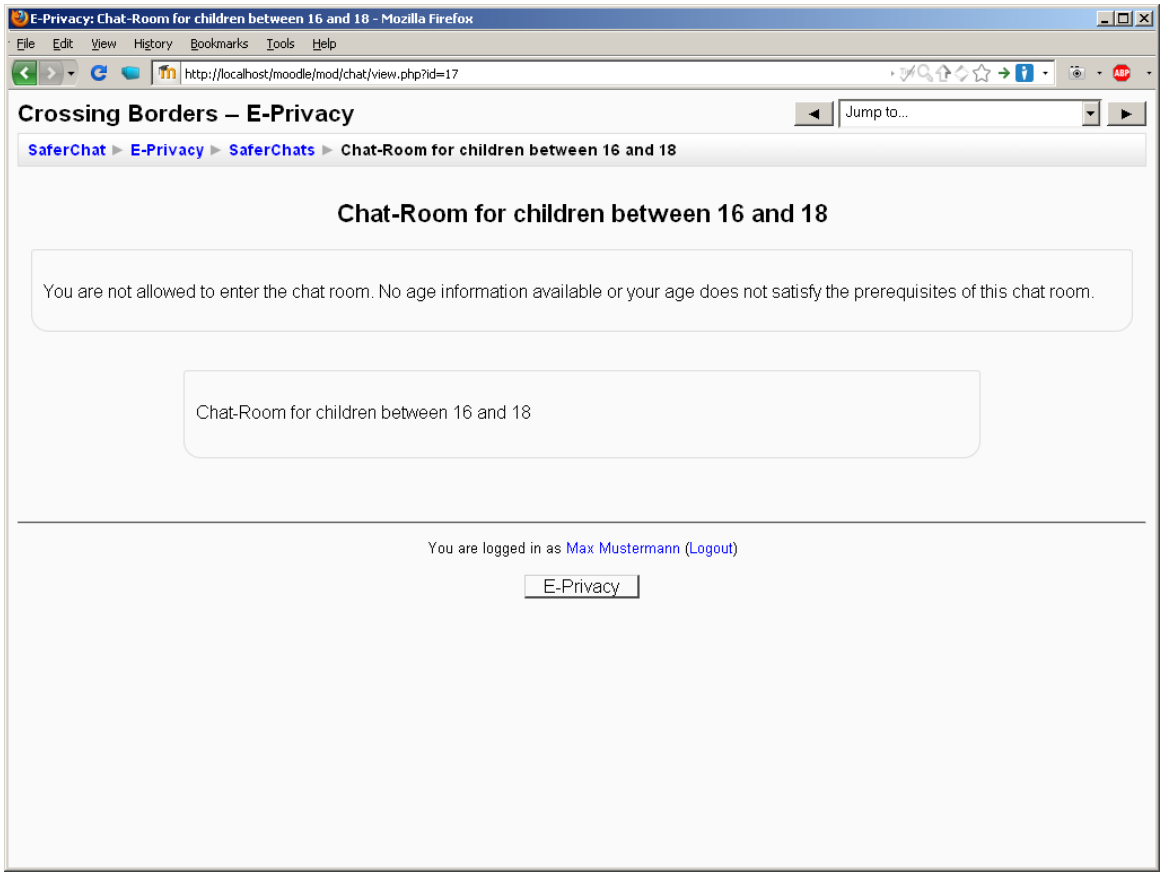


***Figure 18: Access to chat room denied***

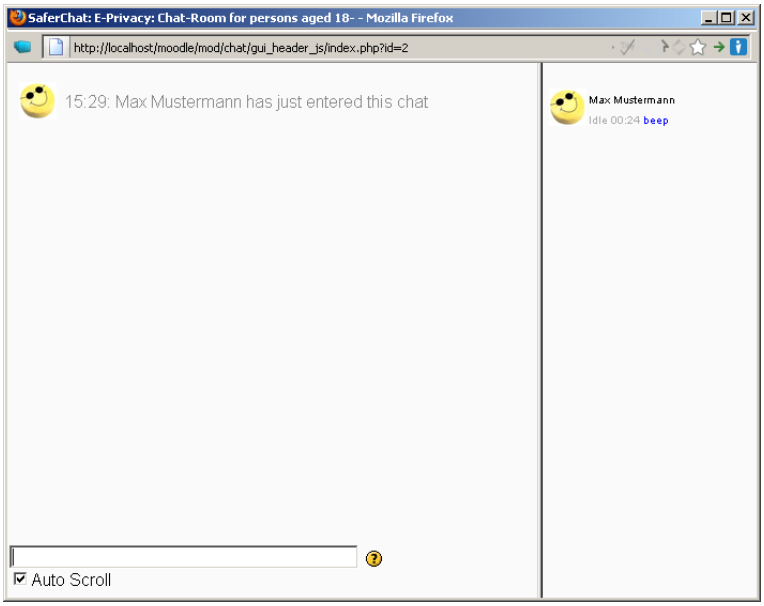


***Figure 19: Access to chat room granted***

# References


[1] Moodle.org, *Moodle developer documentation – Authentication plug-ins*, http://docs.moodle.org/en/Development:Authentication_plugins (2010-01-22)

[2] Moodle.org, *Moodle requirements*, http://docs.moodle.org/en/Installing_Moodle#Requirements (2010-01-22)

[3] Moodle.org, *Moodle Docs – Installing contributed modules of plug-ins*, http://docs.moodle.org/en/Installing_contributed_modules_or_plugins (2010-01-22)

[4] The Apache Software Foundation, *Apache Tomcat 6.0 – Documentation*, http://tomcat.apache.org/tomcat-6.0-doc/index.html (2010-01-25)

[5] Sun, *Java Servlet Specification*, version 2.4, http://java.sun.com/products/servlet/download.html (2010-01-25)

[6] Moodle.org, *Moodle Docs – Adding/editing a course*, http://docs.moodle.org/en/Adding/editing_a_course (2010-01-26)

[7] Joaquín Alcalde-Moraño, Jorge López Hernández-Ardieta, Adrian Johnston, Daniel Martinez, Bernd Zwattendorfer, *D5.8.1b Interface Specification*, v1.0 rev2

[8] International Organization for Standardization (ISO), *English country names and code elements*, http://www.iso.org/iso/english_country_names_and_code_elements (2010-01-28).

[9] IETF, RFC 2045, http://www.ietf.org/rfc/rfc2045.txt (2010-01-29)

[10] Moodle.org, *Moodle Docs – Password salting*, http://docs.moodle.org/en/Password_salting (2010-02-16)

[11] B. Hulsebosch, G. Lenzini, and H. Eertink; *STORK Deliverable 2.3 – Quality Authenticator Scheme*, final, 2009-03-03.