SEVENTH FRAMEWORK PROGRAMME
THEME 3
Information and Communication Technologies

Grant agreement for:
Collaborative project, Small and medium-scale focused research project (STREP)

Technical Report TR-TARWIS-USER-MANUAL:

TARWIS 4.0 User Manual v1.0

Project acronym: WISEBED
Project full title: Wireless Sensor Network Testbeds
Grant agreement no.: 224460

Responsible Partner: UBERN (Philipp Hurni, Gerald Wagenknecht, Markus Anwander, Torsten Braun)
Report Preparation Date: April 29, 2011

Contents

1 What Is TARWIS?

All over the world, researchers have set up small wireless sensor network testbeds for research purposes, in order to test and evaluate the real-world behavior of developed protocol mechanisms. A large number of testbeds have been put into operation, each with different equipment and testbed architecture design (e.g. MoteLab [6], Kansei [3], PowerBench [2], JAWS-DSN, DES-Testbed [1]). The popularity of wireless sensor networks is increasing, and many researchers are setting up and deploying their own new testbeds. Although each testbed may differ with respect to hardware and software, all wireless sensor network testbeds require common functionalities. As every shared resource, a testbed needs a notion of users, it requires support for reprogramming and reconfiguration of the nodes, provisions to debug and remotely reset sensor nodes in case of node failures as well as a solution for collecting and storing experimental data.

TARWIS targets at providing these functionalities independent from the node type and node operating system. The system has been designed to access and manipulate a testbed from within a website, in order to also let researchers access testbed resources remotely over the Internet, in order to share testbed resources with European research partners in a federation of testbeds. TARWIS hence relieves researchers setting up a sensor network testbed from the burden to implement their own scheduling and testbed management solutions. TARWIS has been incrementally developed during the first two years of the WISEBED [4] project by University of Bern, and has recently been demonstrated to the European sensor network research community [5].

TARWIS contains a number of different components, such as the *TARWIS Server*, the *TARWIS GUI*, the *Reservation System*, the *Identity Provider (IDP)* plus a web interface (*IDP Tools*), the *Service Provider (SP)*, and *Sensor Network Authorization System (SNA)* plus a web interface (*SNA GUI*). Fig. 1 shows the relation between the different components.

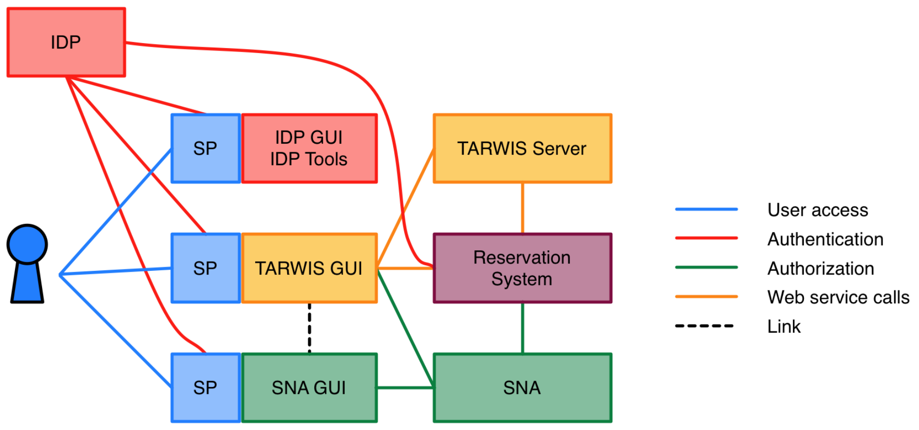


Figure 1: TARWIS Overview.

The core of the system build the TARWIS Server (as backend), the TARWIS GUI (as graphical user interface), and for node reservation the Reservation System. The protect the access to the web-based graphical user interface SPs are used. The authentication is done by the IDP. To manage the users a graphical user interface called IDP Tools us provided. The define the roles of the users within a testbed the SNA with the according GUI has to be used.

2 Installation of the IDP and SP

This Section describes the installation of the Identity Provider (IDP) and the Service Providers (SP).

2.1 Prerequisites

2.1.1 OpenSSL

Recommended Version 0.9.8, Debian Package: openssl The OpenSSL tools are used to handle the server certificates.

```
apt-get update
apt-get install openssl
```

2.1.2 NTP

```
apt-get install ntp-server

tzconfig
Do you want to change that? [n]: y
Number: 12
Name: [] UTC
date
Fri Feb 20 09:32:26 UTC 2009
```

WARNING: If this server is a virtual instance, ntp does not work. You have to synchronize Dom0, because Dom0 regulate the time for the virtual machines.

Example: Time synchronization for Lancaster

```
apt-get install ntpdate
ntpdate -u -b uk.pool.ntp.org
4 Mar 19:32:20 ntpdate[18770]: step time server 77.75.105.150 offset 5.017991
    sec
```

2.1.3 Apache 2.2 with mod_ssl and mod_proxy_ajp

Debian packages: apache2 The modules mod_ssl and mod_proxy_ajp are part of the package.

```
apt-get install apache2
# (apache2-mpm-worker apache2-utils apache2.2-common libapr1 libaprutil1 libpq4
    libsqlite3 -0)
```

2.1.4 cURL

Debian package: curl (optional, as an alternative to wget)

```
apt-get install curl
```

2.1.5 gnupg (GNU Privacy Guard) and gpgv

Debian packages: gnupg and gpgv (recommended, to verify the signature on the installed software)

```
apt-get install gnupg gpgv
```

2.1.6 JAVA 1.5

Make sure that the non-free repository is included in the apt sources (sources.list file or a file in directory /etc/apt/sources.list.d/).

```
echo "deb ftp://mirror.switch.ch/mirror/debian etch main contrib non-free" >> /
    etc/apt/sources.list
```

or

```
echo "deb http://ftp.us.debian.org/debian etch main contrib non-free" >> /etc/apt
    /sources.list
```

Install Java

```
apt-get update
apt-get install sun-java5-jdk
```

Note: for those who like vim with a basic syntax highlighting

```
apt-get install vim
echo "syntax on" > /root/.vimrc
```

To avoid conflicts with other Java virtual machines like kaffe or gcj, deinstalling them is highly recommended. You may also include the following lines in /etc/profile

```
vi /etc/profile

JAVA_HOME=/usr/lib/jvm/java-1.5.0-sun
export JAVA_HOME
```

Check if the correct Java version is included in the path:

```
source /etc/profile

java -version
java version "1.5.0_14"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_14-b03)
Java HotSpot(TM) Server VM (build 1.5.0_14-b03, mixed mode)
```

2.1.7 Maven

Maven is used to build CAS server and client. Currently, there's no Debian package for Maven 2 in the stable distribution. A manual installation is required on pure Debian/stable systems, which is shown below.

```
cd /opt
curl -O http://archive.apache.org/dist/maven/binaries/apache-maven-2.0.9-bin.zip
jar -xf apache-maven-2.0.9-bin.zip
rm apache-maven-2.0.9-bin.zip
ln -s /opt/apache-maven-2.0.9 /opt/apache-maven
chmod +x /opt/apache-maven/bin/mvn
```

Configure the proxy if necessary (search for ¡proxies¿: line 74 - 89). Here an example.

```
cd /opt/apache-maven
vi ./conf/settings.xml

<proxies>
  <proxy>
```

```
      <id>optional</id>
      <active>true</active>
      <protocol>http</protocol>
      <username></username>
      <password></password>
      <host>proxy.iam.unibe.ch</host>
      <port>8080</port>
      <nonProxyHosts>unibe.ch</nonProxyHosts>
    </proxy>

  </proxies>
```

Add the following lines to /etc/profile

```
vi /etc/profile

M2_HOME=/opt/apache-maven
export M2_HOME
M2=$M2_HOME/bin
PATH=$M2:\$PATH
export PATH
```

The environment variables become active in a new shell. Or just read in the new profile by the following command.

```
source /etc/profile
```

2.1.8 Shibboleth (Service Provider) build environment

Since the Shibboleth Service Provider is implemented in C/C++, some C/C++ build tools are required:

```
apt-get install gcc g++ make
```

2.1.9 external libraries (Service Provider)

The Shibboleth service provider is linked against some external libraries. The required header files and libraries are:

```
apt-get install libssl0.9.8 libssl-dev
apt-get install libcurl3 libcurl3-dev
apt-get install apache2-threaded-dev
```

2.2 Tomcat 5.5

```
apt-get install tomcat5.5
```

Configure JVM memory options and not to use the security manager. In /etc/default/tomcat5.5 (line 19) set the following variables:

```
vi /etc/default/tomcat5.5

CATALINA_OPTS="-Xms256M -Xmx512M -XX:MaxPermSize=512M -XX:-DisableExplicitGC -
    server"
TOMCAT5_SECURITY=no
```

The values for memory usage depend on the physical memory of the server. Set Xmx to 512MBytes minimum and XX:MaxPermSize to half of the available memory or 512MBytes minimal. Tomcat user is: tomcat55

In /etc/tomcat5.5/server.xml replace the AJP 1.3 Connector on port 8009: (line 105)

```
vi /etc/tomcat5.5/server.xml

  <Connector port="8009" address="127.0.0.1"
          enableLookups="false" redirectPort="443" protocol="AJP/1.3"
          tomcatAuthentication="false" />
```

Other connectors are not needed when Apache is run in front of Tomcat, so they should be commented out (i.e. the Connector for port 8180).

2.3 Shibboleth IdP 2.0 Installation

Remove XML/Xerces libraries that came with the Tomcat distribution from $CATALINA_HOME/common/endorsed

```
rm /usr/share/tomcat5.5/common/endorsed/xercesImpl.jar
rm /usr/share/tomcat5.5/common/endorsed/xml-apis
```

Get Shibboleth IdP 2.0.0

```
cd /opt
curl -O http://www.iam.unibe.ch/wisebed/shibboleth-idp-2.0.0-bin.zip
jar -xf shibboleth-idp-2.0.0-bin.zip
rm shibboleth-idp-2.0.0-bin.zip
```

Install the Shibboleth IdP package in /opt/shibboleth-idp-2.0.0 and the webapplication in $CATALINA_HOME/webapps/
Endorse libraries from the Shibboleth IdP tar ball in directory $CATALINA_HOME/common/endorsed

```
cd /opt/identityprovider
cp endorsed/*.jar /usr/share/tomcat5.5/common/endorsed/
```

Run the ant tasks to install the Shibboleth IdP software:

```
chmod 755 ant.sh
./ant.sh

Is this a new installation? Answering yes will overwrite your current configurat
    ion. [yes|no]
yes
Where should the Shibboleth Identity Provider software be installed? [default: /
    opt/shibboleth-idp-2.0.0]
/opt/shibboleth-idp-2.0.0
What is the hostname of the Shibboleth Identity Provider server? [default: idp.e
    xample.org]
ipd.example.org
A keystore is about to be generated for you. Please enter a password that will b
    e used to protect it.
SECRET-PASSWORD

(output omitted)

Building jar: /opt/shibboleth-idp-2.0.0/war/idp.war
BUILD SUCCESSFUL
```

Set symbolic links for your convenience. Link /etc/shibboleth to the shibboleth-idp configuration directory and /var/log/shibboleth to the shibboleth-idp log directory:

```
ln -s /opt/shibboleth-idp-2.0.0 /opt/shibboleth-idp
ln -s /opt/shibboleth-idp/conf /etc/shibboleth
ln -s /opt/shibboleth-idp/logs /var/log/shibboleth
```

Set the IDP_HOME environment variable:

```
export IDP_HOME=/opt/shibboleth-idp
```

You may also include the following line into your /etc/profile file:

```
vi /etc/profile

IDP_HOME=/opt/shibboleth-idp
export IDP_HOME
```

Set permissions and ownership in order to allow the tomcat55 user to access directories in $IDP_HOME

```
cd /opt/shibboleth-idp
chown -R tomcat55 logs metadata credentials
chmod 755 logs metadata
```

Create a context descriptor for the IdP web application in $CATALINA_HOME/conf/Catalina/localhost

```
cd /var/lib/tomcat5.5/
mkdir -p conf/Catalina/localhost
```

Create the context descriptor file /var/lib/tomcat5.5/conf/Catalina/localhost/idp.xml

```
vi /var/lib/tomcat5.5/conf/Catalina/localhost/idp.xml

<Context
    docBase="/opt/shibboleth-idp/war/idp.war"
    privileged="true"
    antiResourceLocking="false"
    antiJARLocking="false"
    unpackWAR="false" />
```

2.4 MySQL Server Installation

2.4.1 Installation

Install the Debian etch package for MySQL 5

```
 apt-get install mysql-server-5.0
```

By default, the mysql daemon only listens to localhost on IPv4. Set password for the root user in MySQL:

IMPORTANT: replace the SECRET-MySQL-ROOT-PASSWORD with a own password!

```
 /usr/bin/mysqladmin -u root password 'SECRET-MySQL-ROOT-PASSWORD'
```

2.4.2 IDP user data database

Create the IDP database and an IDP-Admin user:

IMPORTANT: replace the **two** SECRET-USER-DB-PASSWORD with a own password!

```
mysql -u root -p

SET NAMES 'utf8';
SET CHARACTER SET utf8;
CHARSET utf8;

USE mysql;
INSERT INTO user (Host,User,Password,Select_priv,Insert_priv,
  Update_priv,Delete_priv, Create_tmp_table_priv,Lock_tables_priv,Execute_priv)
  VALUES   ('localhost','idpadmin',PASSWORD('SECRET-USER-DB-PASSWORD'), 'Y','Y','
      Y','Y','Y','Y','Y');
FLUSH PRIVILEGES;

CREATE DATABASE IF NOT EXISTS IDPtools CHARACTER SET = utf8;
USE IDPtools;
GRANT ALL PRIVILEGES ON IDPtools.* TO 'idpadmin'@'localhost' IDENTIFIED BY '
    SECRET-USER-DB-PASSWORD';
QUIT
```

Check if the user idpadmin has been created with password secret-password:

```
mysql -u idpadmin -p
Enter password:'SECRET-USER-DB-PASSWORD'
Welcome to the MySQL monitor. Commands end with ; or \g.
[...]
quit;
```

Create tables:

```
curl -O http://www.iam.unibe.ch/wisebed/scripts/wisebed.sql
sed -i 's/idp.example.org/idp.example.org/g' wisebed.sql
mysql -u root -p < wisebed.sql
rm wisebed.sql
```

Define your main IDP-Administrator

```
mysql -u idpadmin -p
USE IDPtools;
INSERT INTO `idp_Users` (`idUser`, `idGroup`, `uniqueID`, `username`, `password`,
     `custom1`, `custom2`, `state`, `dateCreate`, `dateExpire`, `dateModify`, `
    dateLastLogin`)
VALUES ('1', '1', '000001@idp.example.org', 'USERNAME-WEB-INTERFACE-ADMIN', MD5('
    SECRET-WEB-INTERFACE-ADMIN-PASSWORD'), NULL, NULL, 'active', NOW(),
    '2032-09-30 08:38:35', NOW(), NULL);
```

idp.example.org replace it with your address (for example 000001@idp.unibe.ch) USERNAME replace it with the admin username (for example admin@unibe-ch)

Set the Last name and First name

```
INSERT INTO `idp_UserAttributes` (`idAttribute`, `idUser`, `value`) VALUES
    ('22', '1', 'surname');
INSERT INTO `idp_UserAttributes` (`idAttribute`, `idUser`, `value`) VALUES
    ('23', '1', 'givenname');
quit;
```

2.5 Install IDP-Admin

Install the Debian etch package for PHP 5 and PEAR DB to access MySQL

```
apt-get install libapache2-mod-php5
apt-get install php5-mysql
apt-get install php-db
```

Activate apache2 modul mod_rewrite

```
ln -s /etc/apache2/mods-available/rewrite.load /etc/apache2/mods-enabled/
```

Make sure PHP5 module is installed:

```
a2enmod php5
/etc/init.d/apache2 force-reload

cd /opt
curl -O http://www.iam.unibe.ch/wisebed/idpadmin.tgz
tar -zxf idpadmin.tgz
rm idpadmin.tgz
ln -s /opt/idpadmin/htdocs /var/www/idpadmin
```

Add your IDP URL and Admin mail address

```
cd /opt/idpadmin
find . -type f -exec sed -i 's/idp.example.org/idp.example.org/g' {} \;
find . -type f -exec sed -i 's/admin@example.org/admin@example.org/g' {} \;
```

Insert password for database (line 31) in /opt/idpadmin/libs/config.php

```
vi /opt/idpadmin/libs/config.php

define('DB_PASSWORD', ''SECRET-USER-DB-PASSWORD'');
```

Set Permissions

```
cd /opt/idpadmin/
chgrp www-data logs
chmod g+w logs
chgrp www-data import
chmod g+w import
chgrp -R www-data templates/compiled
chmod -R g+w templates/compiled
```

2.6 Install the java mysql connector

Install debian package apt-get install libmysql-java. mysql-connector-java.jar will be installed in /user/share/-java/:

```
apt-get install libmysql-java
```

Create a symbolic link to provide tomcat with the java mysql connector classes:

```
ln -s /usr/share/java/mysql-connector-java.jar /usr/share/tomcat5.5/common/lib/
```

2.7 CAS Server web application

Get and uncompress CAS server 3.2.1 from http://www.ja-sig.org/products/cas/.

```
cd /opt
curl -O http://www.ja-sig.org/downloads/cas/cas-server-3.2.1-release.zip
jar -xf cas-server-3.2.1-release.zip
rm cas-server-3.2.1-release.zip
```

Replace the WEB-INF

```
cd  /opt/cas-server-3.2.1/cas-server-webapp/src/main/
rm -rf webapp
curl -O http://www.iam.unibe.ch/wisebed/cas/webapp.tgz
tar -zxf webapp.tgz
rm webapp.tgz
```

The config file (/opt/cas-server-3.2.1/cas-server-webapp/src/main/webapp/WEB-INF/deployerConfigContext.xml) is now configured for a MySQL connection to the IDPtools database. You just have to adjust the password (line 148)

```
 vi /opt/cas-server-3.2.1/cas-server-webapp/src/main/webapp/WEB-INF/
     deployerConfigContext.xml

<property name="password">
     <value>SECRET-PASSWORD</value>
 </property>
```

Add (for instance at line 30) the following dependency to the maven config file (/opt/cas-server-3.2.1/cas-server-webapp/pom.xml):

```
vi /opt/cas-server-3.2.1/cas-server-webapp/pom.xml

  <dependency>
         <groupId>${project.groupId}</groupId>
       <artifactId>cas-server-support-jdbc</artifactId>
         <version>${project.version}</version>
    </dependency>
```

Now build the CAS server web application:

```
cd /opt/cas-server-3.2.1/cas-server-support-jdbc
mvn package
cd /opt/cas-server-3.2.1/cas-server-webapp
mvn package
```

Create a context descriptor for the CAS server web application in $CATALINA_HOME/conf/Catalina/localhost/
Create the context descriptor file /var/lib/tomcat5.5/conf/Catalina/localhost/cas.xml

```
 vi /var/lib/tomcat5.5/conf/Catalina/localhost/cas.xml

<Context
    docBase="/opt/cas-server-3.2.1/cas-server-webapp/target/cas.war"
    privileged="true"
    antiResourceLocking="false"
    antiJARLocking="false"
    unpackWAR="false" />
```

cas log

```
mkdir /var/log/cas
chown tomcat55  /var/log/cas
```

2.8 CAS Client Installation

```
cd /opt
wget http://www.ja-sig.org/downloads/cas-clients/cas-client-3.1.1-release.tar.gz
tar -zxf cas-client-3.1.1-release.tar.gz
rm cas-client-3.1.1-release.tar.gz
cd cas-client-3.1.1/cas-client-core
mvn package
```

Make the CAS client classes available to the Shibboleth IdP web application:

```
cp /opt/cas-client-3.1.1/cas-client-core/target/cas-client-core-3.1.1.jar /opt/
    identityprovider/lib/
```

2.9 Apache Configuration

Apache has to be configured with the modules mod_ssl for SSL support and mod_proxy_ajp to redirect requests to Tomcat. The Apache configuration usually takes place in files in the directory /etc/apache2/sites-available/

mod_ssl

Copy the idp.example.org.key to the directory /etc/ssl/private/ and idp.example.org.crt to the directory /etc/ssl/certs/.

```
cp idp.example.org.key /etc/ssl/private/
cp idp.example.org.crt /etc/ssl/certs/
```

Get the bundle with the accepted CA root certificates and place it into the directory /etc/ssl/.

```
cd  /etc/ssl/
curl -O  http://www.iam.unibe.ch/wisebed/metadata/ca-bundle.crt
```

Make sure the server listens on port 443 (and 8443 for AA connections) with the ”Listen” directive in /etc/apache2/ports.conf

```
vi /etc/apache2/ports.conf

Listen 443
Listen 8443
```

Remove default configuration

```
cd /etc/apache2/sites-enabled
unlink 000-default
```

Configure the virtual host on idp.example.org. Create a new configuration file in /etc/apache2/sites-available or adapt an existing one. For example, use /etc/apache2/sites-available/aai-logon

```
vi /etc/apache2/sites-available/aai-logon

<IfModule mod_ssl.c>
   <VirtualHost _default_:443>
        ServerName idp.example.org
   SSLEngine On
   SSLCipherSuite ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
   SSLCertificateFile              /etc/ssl/certs/idp.example.org.crt
   SSLCertificateKeyFile           /etc/ssl/private/idp.example.org.key
   SSLCertificateChainFile         /etc/ssl/certs/idp.example.org.crt
   SSLVerifyDepth                  10
```

```
    SSLOptions                          +StdEnvVars

          DocumentRoot  /var/www/

   </VirtualHost>
 </IfModule>

 <IfModule mod_proxy_ajp.c>
       ProxyRequests  Off

      <Proxy  ajp://localhost:8009>
            Allow  from  all
    </Proxy>

    ProxyPass  /idp  ajp://localhost:8009/idp  retry=5
    ProxyPass  /cas  ajp://localhost:8009/cas  retry=5

</IfModule>
```

Configure the virtual host for port 8443. Make a new configuration file in /etc/apache2/sites-available or adapt an existing one. For example, use /etc/apache2/sites-available/aai-aa. Uncomment the last ProxyPass line if CAS is going to be used.

```
vi  /etc/apache2/sites-available/aai-aa

 <IfModule mod_ssl.c>
   <VirtualHost _default_:8443>
   ServerName idp.example.org
 SSLEngine On
 SSLCipherSuite ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
 SSLCertificateFile                /etc/ssl/certs/idp.example.org.crt
 SSLCertificateKeyFile            /etc/ssl/private/idp.example.org.key
 SSLCertificateChainFile           /etc/ssl/certs/idp.example.org.crt
 SSLCACertificateFile              /etc/ssl/ca-bundle.crt
 SSLVerifyDepth   10
 SSLVerifyClient            optional_no_ca
 SSLOptions                -StdEnvVars +ExportCertData


      </VirtualHost>
  </IfModule>

 <IfModule mod_proxy_ajp.c>
    ProxyRequests  Off

    <Proxy  ajp://localhost:8009>
          Allow  from  all
      </Proxy>

      ProxyPass  /idp  ajp://localhost:8009/idp  retry=5
 </IfModule>
```

Enable both virtual hosts (aai-logon and aai-aa):

```
 a2ensite  aai-logon
 a2ensite  aai-aa
 apache2ctl  -t
 Syntax OK
```

Enable the ssl module.

```
a2enmod ssl
```

Module ssl installed; run /etc/init.d/apache2 force-reload to enable. Enable the ajp proxy module, the module mod_proxy has also to be enabled.

```
a2enmod proxy_ajp
apache2ctl -k restart
```

2.10 Shibboleth IDP 2.0 Configuration

2.10.1 Configure Shibboleth IdP

Copy the key and certificate used for signing to the /opt/shibboleth-idp/credentials/ directory. Make sure the tomcat user has read permissions for both the key and the certificate file.

```
cp idp.example.org.{key,crt} /opt/shibboleth-idp/credentials/
```

Remember to set appropriate ownership and permissions, notably for the file idp.example.org.key.

```
cd /opt/shibboleth-idp/credentials
chown tomcat55 idp.example.org.key
chgrp root idp.example.org.{key,crt}
chmod 440 idp.example.org.key
chmod 644 idp.example.org.crt
```

Move away the self-signed certificate generated from the installation procedure.

```
mv idp.crt idp-self-signed.crt
mv idp.key idp-self-signed.key
```

Use the certificate idp.example.org.crt for the IdP-to-SP communcation together with idp.example.org.key.

```
cd /opt/shibboleth-idp/credentials
ln -sf idp.example.org.crt idp.crt
ln -sf idp.example.org.key idp.key
```

Configure to use the EditNet federation metadata and the trusted root certificate in /opt/shibboleth-idp-2.0.0/conf/relying-party.xml. The entity identifier (https://idp.example.org/idp/shibboleth) of the IdP may be adapted also, if a different value than the default should be used.

Get the relying-party.xml from unibe.ch

```
cd /opt/shibboleth-idp-2.0.0/conf/
curl -O  http://www.iam.unibe.ch/wisebed/scripts/relying-party.xml
```

Replace idp.example.org with your IDP URL.

```
sed -i 's/idp.example.org/idp.example.org/g' relying-party.xml
```

Download the EdiNetaai specific attribute-resolver.xml file and adapt it (line 740).

```
cd /opt/shibboleth-idp/conf/
curl -O http://www.iam.unibe.ch/wisebed/scripts/attribute-resolver.xml
vi /opt/shibboleth-idp/conf/attribute-resolver.xml

  <resolver:DataConnector id="myMySQL" xsi:type="RelationalDatabase" xmlns="urn:
      mace:shibboleth:2.0:resolver:dc">
```

```
        <ApplicationManagedConnection jdbcDriver="com.mysql.jdbc.Driver"
              jdbcURL="jdbc:mysql://localhost/IDPtools" jdbcUserName="idpadmin"
                  jdbcPassword="SECRET-USER-DB-PASSWORD" />
        <QueryTemplate>
              <![CDATA[
                       select * from view_aa where principal = '\$requestContext.
                           principalName'
                ]]>
        </QueryTemplate>
    </resolver:DataConnector>
```

The Attribute Filter Policy file attribute-filter.xml allows the release of attributes to every test Service Provider within the EdiNet Federation. For more configuration details look at AttributeFilterPolicy.

```
cd /opt/shibboleth-idp/conf/
curl -O http://www.iam.unibe.ch/wisebed/scripts/attribute-filter.xml
```

Configure the CAS client filter for the Shiboleth IdP web application in /opt/identityprovider/build/WEB-INF/web.xml

```
cd /opt/identityprovider/build/WEB-INF/
curl -O http://www.iam.unibe.ch/wisebed/scripts/web.xml
```

Replace idp.example.org with your server name

```
sed -i 's/idp.example.org/idp.example.org/g' web.xml
```

Redeploy the Shibboleth IdP web application, responding no. Tomcat will reload the web application provided that the context descriptor points to the file /opt/identityprovider/war/idp.war (see the IdP deployment section for that).

```
cd /opt/identityprovider/
./ant.sh install

Buildfile: build.xml

install:
Is this a new installation? Answering yes will overwrite your current
    configuration. [yes|no]
no
Copying 1 file to /opt/shibboleth-idp-2.0.0/lib
JARs are never empty, they contain at least a manifest file
Building jar: /opt/shibboleth-idp-2.0.0/war/idp.war

BUILD SUCCESSFUL
```

2.10.2 Some Tests

Restart tomcat

```
/etc/init.d/tomcat5.5 restart
```

Restart apache

```
apache2ctl restart
```

In case of errors, check the log files:

```
tail -f /var/log/apache2/*.log
tail -f /var/log/tomcat5.5/*.log
```

Try to open the following URLs in your browser:

```
https://MY_IDP.SERVER.COM/cas/login
https://MY_IDP.SERVER.COM/idp/profile/Status
# (You can test your IDP-Admin credentials if you like (Message Log In
    Successful)
```

In case of errors, check also the IDP and CAS log files:

```
tail -f /var/log/shibboleth/*.log
tail -f /var/log/cas/caslog.log
tail -f /var/log/apache2/*.log
tail -f /var/log/tomcat5.5/*.log
```

2.11 Service Provider to protect IDPtools web interface

2.11.1 Source code download

```
export MYBUILD=/opt/shibsp2.0-build
mkdir \$MYBUILD

wget http://shibboleth.internet2.edu/downloads/log4shib/1.0.1/log4shib-1.0.1.tar
    .gz -P $MYBUILD
wget http://mirror.switch.ch/mirror/apache/dist/xerces/c/2/sources/xerces-c-
    src_2_8_0.tar.gz -P $MYBUILD
wget http://www.iam.unibe.ch/wisebed/xerces-c-src_2_8_0.tar.gz -P $MYBUILD
wget http://www.iam.unibe.ch/wisebed/xml-security-c-1.4.0.tar.gz -P $MYBUILD
wget http://www.iam.unibe.ch/wisebed/xmltooling-1.1.tar.gz -P $MYBUILD
wget http://www.iam.unibe.ch/wisebed/opensaml-2.1.tar.gz -P $MYBUILD
wget http://www.iam.unibe.ch/wisebed/shibboleth-sp-2.1.tar.gz -P $MYBUILD

for f in $MYBUILD/*.tar.gz; do tar -xzvf $f -C $MYBUILD; done
```

2.11.2 Build

```
export SHIB_HOME=/opt/shibboleth-sp-2.0/
export XERCESCROOT=$MYBUILD/xerces-c-src_2_8_0/
mkdir $SHIB_HOME
```

1. Log4Shib:

```
cd $MYBUILD/log4shib-1.0.1/
./configure --disable-static --disable-doxygen --prefix=$SHIB_HOME
make
make install
```

2. XercesC:

```
cd $MYBUILD/xerces-c-src_2_8_0/src/xercesc/
./runConfigure -p linux -r pthread -P $SHIB_HOME
make
env XERCESCROOT=\$XERCESCROOT make install
```

3. XML-Security:

```
cd $MYBUILD/xml-security-c-1.4.0
./configure --without-xalan --prefix=$SHIB_HOME
make
env XERCESCROOT=\$XERCESCROOT make install
```

4. XML-Tooling:

```
cd $MYBUILD/xmltooling-1.1/
./configure --with-log4shib=$SHIB_HOME --prefix=\$SHIB_HOME -C
make
make install
```

5. OpenSAML:

```
cd $MYBUILD/opensaml-2.1/
./configure --prefix=$SHIB_HOME --with-log4shib=\$SHIB_HOME -C
make
make install
```

6. Shibboleth Service Provider:

```
cd $MYBUILD/shibboleth-2.1/
./configure --with-saml=$SHIB_HOME --enable-apache-22 --with-log4shib=$SHIB_HOME
    --prefix=$SHIB_HOME -C
make
 make install
```

2.11.3 Install Shibboleth Apache module

Create the file /etc/apache2/mods-available/shib.load

```
vi /etc/apache2/mods-available/shib.load

# Load the shibboleth module
LoadModule mod_shib /opt/shibboleth-sp2/lib/shibboleth/mod_shib_22.so
```

Create the file /etc/apache2/mods-available/shib.conf

```
vi /etc/apache2/mods-available/shib.conf

# Global Configuration
# This is the XML file that contains all the global, non-apache-specific
# configuration. Look at this file for most of your configuration parameters.
ShibConfig /etc/shibboleth2/shibboleth2.xml

# Used for example logo and style sheet in error templates.
<IfModule mod_alias.c>
  <Location /shibboleth-sp>
   Allow from all
  </Location>
  Alias /shibboleth-sp/main.css /opt/shibboleth-sp2/share/doc/shibboleth/main.
     css
  Alias /shibboleth-sp/logo.jpg /opt/shibboleth-sp2/share/doc/shibboleth/logo.
     jpg
</IfModule>
```

Adjust the Apache configuration /etc/apache2/envvars

```
vi /etc/apache2/envvars

...
...
# This file is generated from envvars-std.in
#
export LD_LIBRARY_PATH=/opt/shibboleth-sp2/lib
```

Enable the Shibboleth Apache module:

```
a2enmod shib
Module shib installed; run /etc/init.d/apache2 force-reload to enable.
```

2.11.4 Install Shibboleth daemon

Define the current Shibboleth SP release Symlink the current installation:

```
ln -sf \$SHIB_HOME /opt/shibboleth-sp2
```

Prepare SP2 init script Copy the distribution init script:

```
cp \$SHIB_HOME/etc/shibboleth/shibd-debian /etc/init.d/shibd
```

Adjust the init script /etc/init.d/shibd

```
vi /etc/init.d/shibd

PATH=/sbin:/bin:/usr/sbin:/usr/bin
DESC="Shibboleth 2 daemon"
NAME=shibd
SHIB_HOME=/opt/shibboleth-sp2/
SHIBSP_CONFIG=/etc/shibboleth2/shibboleth2.xml
LD_LIBRARY_PATH=/opt/shibboleth-sp2/lib
DAEMON=/opt/shibboleth-sp2/sbin/shibd
SCRIPTNAME=/etc/init.d/$NAME
PIDFILE=/var/run/$NAME.pid
```

Install the init script:

```
chmod +x /etc/init.d/shibd
update-rc.d shibd defaults
```

WARNING: This step is only needed if you do a fresh installation. In case of an update, these steps will overwrite the existing configuration files, which may not be what you want! Copy the configuration files:

```
mkdir /etc/shibboleth2
cp $SHIB_HOME/etc/shibboleth/attribute-map.xml /etc/shibboleth2/
cp $SHIB_HOME/etc/shibboleth/attribute-policy.xml /etc/shibboleth2/
cp $SHIB_HOME/etc/shibboleth/native.logger /etc/shibboleth2/
cp $SHIB_HOME/etc/shibboleth/shibboleth2.xml /etc/shibboleth2/
cp $SHIB_HOME/etc/shibboleth/shibd.logger /etc/shibboleth2/
cp $SHIB_HOME/etc/shibboleth/syslog.logger /etc/shibboleth2/
```

Prepare logging directory

```
mkdir -p /var/log/shibboleth-sp/
touch /var/log/shibboleth-sp/native.log
chgrp www-data /var/log/shibboleth-sp/native.log
```

```
chmod g+w /var/log/shibboleth-sp/native.log
touch /opt/shibboleth-sp-2.0/var/log/shibboleth/shibd.log
touch /opt/shibboleth-sp-2.0/var/log/shibboleth/transaction.log
ln -s /opt/shibboleth-sp-2.0/var/log/shibboleth/shibd.log /var/log/shibboleth-sp
    /
ln -s /opt/shibboleth-sp-2.0/var/log/shibboleth/transaction.log /var/log/
    shibboleth-sp/
```

Configuration shibboleth main configuration file

```
cd /etc/shibboleth2
mv shibboleth2.xml shibboleth2.xml.original
curl -O http://www.iam.unibe.ch/wisebed/scripts/shibboleth2.xml
sed -i 's/sp.example.org/idp.example.org/g' shibboleth2.xml
```

Check the certificate paths

```
ls /etc/ssl/private/idp.example.org.key
ls /etc/ssl/certs/idp.example.org.crt
```

Change in /etc/shibboleth2/shibboleth2.xml at line 207 the supportContact with your email address

```
vi /etc/shibboleth2/shibboleth2.xml

supportContact="admin@sp.example.org"
```

Attribute handling Get the attribute-map.xml and attribute-policy.xml

```
cd /etc/shibboleth2
curl -O http://www.iam.unibe.ch/wisebed/scripts/attribute-map.xml
curl -O http://www.iam.unibe.ch/wisebed/scripts/attribute-policy.xml
```

2.12 Tests

(Re)start shibd

```
/etc/init.d/shibd restart
```

Check logfiles:

```
tail -f /var/log/shibboleth-sp/*
```

Now open the URL:

```
https://idp.example.org/idpadmin/
```

- Press the Login Button. This forwards you to https://idp.example.org/idpadmin/admin/.
- The shibboleth Service Provider forwards you the WAYF Server (IDPtools is protected by '/opt/idpadmin/htdocs/admin/.htaccess')
- Select your IDP in the List
- Now the IDP is addressed. The cas-client forwards you to the cas-server, which asks you for your credentials.
- Enter your main IDP-Administrator username and password.

- Now you should see a You don't have permission to access this site notice.
- add Install to your URL -¿ https://idp.example.org/idpadmin/admin/Install
- Now you should be the main 'IDP-Administrator with full authorization
- Create one or more sub-groups
- Create users
- You give users admin authorization for the sub-groups

3 Installation of the SNA

3.1 SNA installation

Webserver:

```
apt-get install libapache2-mod-php5 php5 php5-common php5-curl php5-dev php5-gd
    php5-mysql php5-mhash
```

MySQL

```
apt-get install mysql-server-5.0 mysql-common
/usr/bin/mysqladmin -u root password 'SECRET-MySQL-ROOT-PASSWORD'
```

3.2 Get SNAportal Source Code

```
cd /opt
wget http://www.iam.unibe.ch/wisebed/sna-portal-1.0.5.tgz
tar -zxf sna-portal-1.0.5.tgz
chown -R www-data:www-data sna-portal-1.0.5
```

3.3 MySQL database setup

```
mysqladmin -u root -p create snaportal

mysql -u root -p

GRANT DELETE, INSERT, SELECT, UPDATE
      ON snaportal.*
      TO snaportal@localhost  IDENTIFIED BY 'wisebed';

quit;

cd /opt/sna-portal-1.0.5/db
mysql -u root -p --database=snaportal < sna-portal-db.sql
```

3.4 Apache configuration

```
vi /etc/apache2/sites-available/sna-portal

# define an Alias for the AAIportal
#
Alias /sna-portal  "/opt/sna-portal-1.0.5/web"

# configure the directory with the AAIportal web application
#
<Directory "/opt/sna-portal-1.0.5/web">
     DirectoryIndex index.php
     #
     # IMPORTANT ! Make sure we can override PHP configuration
     # and Shibboleth authentication settings in .htaccess files
     # in the web directory
     #
     AllowOverride All
     Order allow,deny
     Allow from all
</Directory>

a2ensite sna-portal
/etc/init.d/apache2 reload
```

3.5 SNA configuration

The file /etc/SNA.conf specifies settings for the per-site local authorization tool SNA (Sensor Network Authorization Tool). This file is given in the top-level directory and
a) needs to be moved to /etc/SNA.conf and be made readable for the users www-data and root (as root, run ”chmod 644 /etc/SNA.conf)

```
cd TARWIS_4.0
cp SNA.conf /etc
```

b) needs to be **ADAPTED** for **YOUR PARTICULAR SITE**

Then, go on and **ALTER THIS FILE** set your particular settings there:

```
SNA_GUI_URL=https://<yourhost>/sna-portal                            # the webpage where SNA GUI is accessible over HTTP
SNA_AUTHORIZATION_URL=https://<yourhost>/webservices/authorization.php  # webservice url for authorization
GET_UID_URL=https://<yourhost>/portal/getUID.php

SUPPORT_EMAIL=somebody@youruniversity.whereveryouare                 # the email adress of the person responsible
                                                                     # for the wisebed testbed
SUPPORT_URL=http://www.youruniversity.whereveryouare/whereveryouwantthistopointto.html
                                                                     # a webpage, e.g. of the responsible person
SNA_MYSQL_PASS=wisebed                                               # the mysql password
AAI_PORTAL_SMTP=mail.iam.unibe.ch                                    # the smpt relay server used for automated mail sending
```

then, go to the SNA directory and set up the mysql db

```
cd /opt/sna-portal-1.0.5
sh resetDB.sh
```

You will be prompted to enter your mysql password (again) then, go to /opt/sna-portal-1.0.5 again and run setConfig.sh. This program sets the configuration values of /etc/SNA.conf to the tool at /opt/sna-portal-1.0.5

```
cd /opt/sna-portal-1.0.5
sh setConfig.sh
```

3.6 Enable webservice

```
cp -r /opt/sna-portal-1.0.5/webservices /var/www/
chown -R www-data:www-data /var/www/webservices
```

3.7 Login

```
https://<DOMAIN>/sna-portal/admin.php

User: portaladmin
Pass: wisebed
```

4 User Administration

4.1 Manage Groups

The following example (cf. Fig 2) illustrates a possible scenario for an university with different faculties and departments.

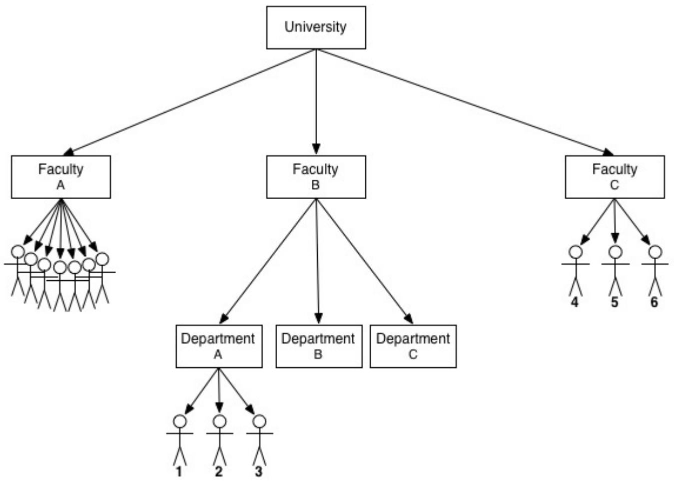


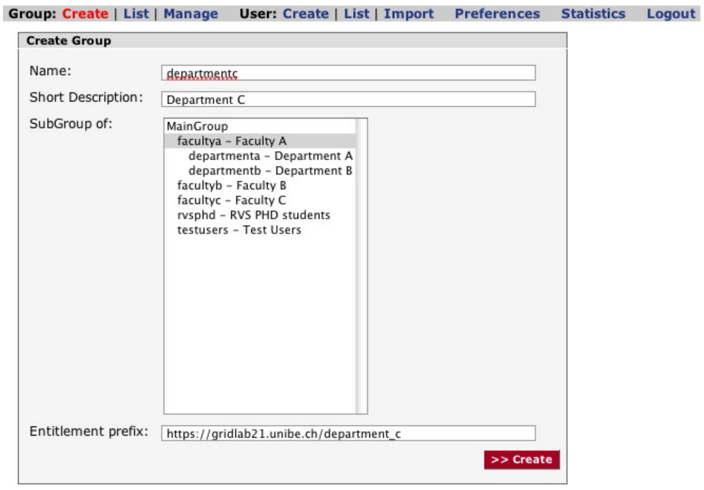
Figure 2: User administration: manage groups.

To create a group, first, login to the IDPtools web-interface (as main admin). Choose menu *Group: Create*, as shown in Fig. 3. To create Department C as a child of the Faculty A, choose Faculty A in the list. Fill in *Name*, a *Short Description* and an *Entitlement Prefix*. Now press the *Create* button.

Now fill in the Helpdesk information and press the Save button, as shown in Fig 4. In this menu it is possible to:

- Invite an user as administrator for this group and all its sub groups (later in this tutorial)
- Change the helpdesk information
- Change the mail templates for user notifications
- Enabling and disabling this group

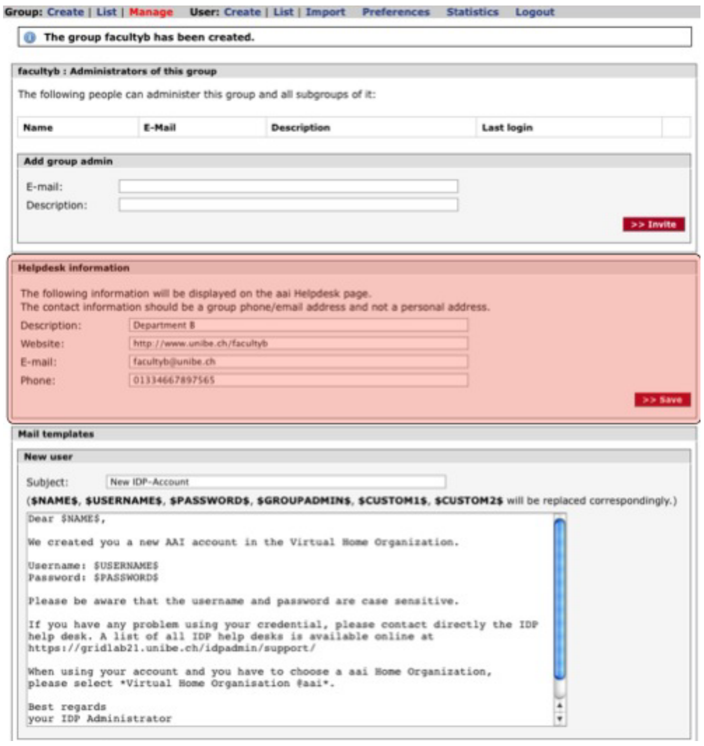
After creating all faculties and departments click on *Group: List* to get an overview, as shown in Fig. 5.


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout
>> Create

Create Group

Name:	departmentc
Short Description:	Department C
SubGroup of:	MainGroup facultya - Faculty A departmenta - Department A departmentb - Department B facultyb - Faculty B facultyc - Faculty C rvsphd - RVS PHD students testusers - Test Users
Entitlement prefix:	https://gridlab21.unibe.ch/department_c

Figure 3: User administration: create group.


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout
>> Invite
>> Save

The group facultyb has been created.

facultyb : Administrators of this group

The following people can administer this group and all subgroups of it:

Name	E-Mail	Description	Last login

Add group admin

E-mail:	
Description:	

Helpdesk information

The following information will be displayed on the aai Helpdesk page.
The contact information should be a group phone/email address and not a personal address.

Description:	Department B
Website:	http://www.unibe.ch/facultyb
E-mail:	facultyb@unibe.ch
Phone:	01334667897565

Mail templates

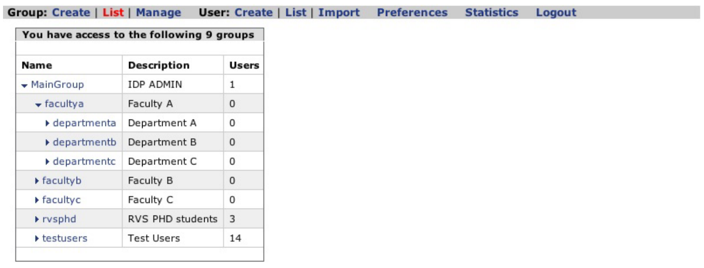
New user

Subject: New IDP-Account

(**$NAME$**, **$USERNAME$**, **$PASSWORD$**, **$GROUPADMIN$**, **$CUSTOM1$**, **$CUSTOM2$** will be replaced correspondingly.)

```
Dear $NAME$,

We created you a new AAI account in the Virtual Home Organization.

Username: $USERNAME$
Password: $PASSWORD$

Please be aware that the username and password are case sensitive.

If you have any problem using your credential, please contact directly the IDP
help desk. A list of all IDP help desks is available online at
https://gridlab21.unibe.ch/idpadmin/support/

When using your account and you have to choose a aai Home Organization,
please select *Virtual Home Organisation @aai*.

Best regards
your IDP Administrator
```

Figure 4: User administration: manage group.


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout

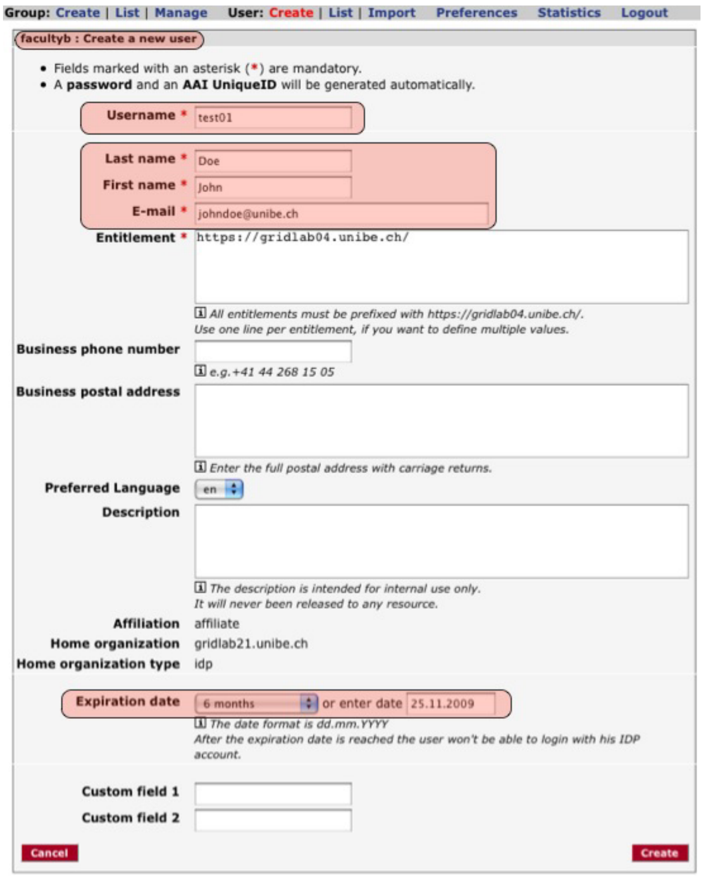
You have access to the following 9 groups

Name	Description	Users
MainGroup	IDP ADMIN	1
facultya	Faculty A	0
departmenta	Department A	0
departmentb	Department B	0
departmentc	Department C	0
facultyb	Faculty B	0
facultyc	Faculty C	0
rvsphd	RVS PHD students	3
testusers	Test Users	14

Figure 5: User administration: list groups.

4.2 Manage Users

Now we create some users. Select in the menu *Group: List* the group the user should be added. Then click on *User: Create*. Now you see the selected group in the upper right corner (cf. Fig. 6).


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout
Cancel
Create

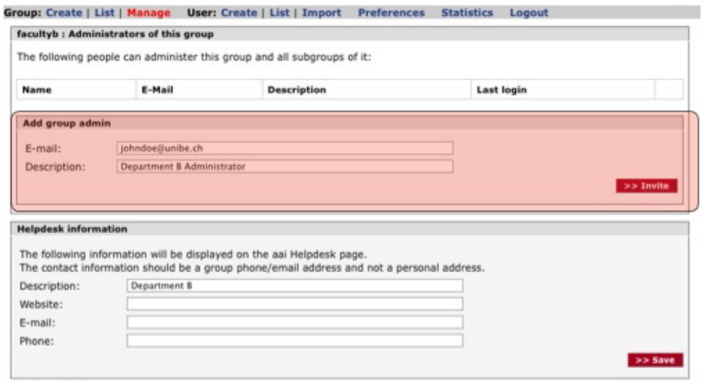
facultyb : Create a new user

- Fields marked with an asterisk (*) are mandatory.
- A **password** and an **AAI UniqueID** will be generated automatically.

Username *	test01
Last name *	Doe
First name *	John
E-mail *	johndoe@unibe.ch
Entitlement *	https://gridlab04.unibe.ch/ *All entitlements must be prefixed with https://gridlab04.unibe.ch/. Use one line per entitlement, if you want to define multiple values.*
Business phone number	*e.g.+41 44 268 15 05*
Business postal address	*Enter the full postal address with carriage returns.*
Preferred Language	en
Description	*The description is intended for internal use only. It will never been released to any resource.*
Affiliation	affiliate
Home organization	gridlab21.unibe.ch
Home organization type	idp
Expiration date	6 months or enter date 25.11.2009 *The date format is dd.mm.YYYY After the expiration date is reached the user won't be able to login with his IDP account.*
Custom field 1	
Custom field 2	

Figure 6: User administration: create user.

Because it is not feasible for the main administrator to do the whole work, delegate the subgroups to other administrators. To add an administrator select the corresponding group in *Group: List* and the click on *Group: Manage*.


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout
>> Invite
>> Save

facultyb : Administrators of this group

The following people can administer this group and all subgroups of it:

Name	E-Mail	Description	Last login

Add group admin

E-mail:	johndoe@unibe.ch
Description:	Department B Administrator

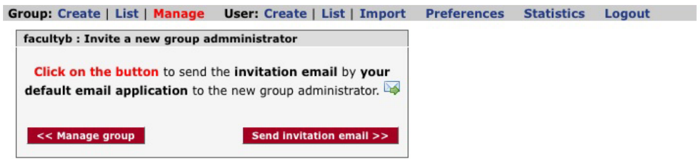
Helpdesk information

The following information will be displayed on the aai Helpdesk page.
The contact information should be a group phone/email address and not a personal address.

Description:	Department B
Website:	
E-mail:	
Phone:	

Figure 7: User administration: add administrator.

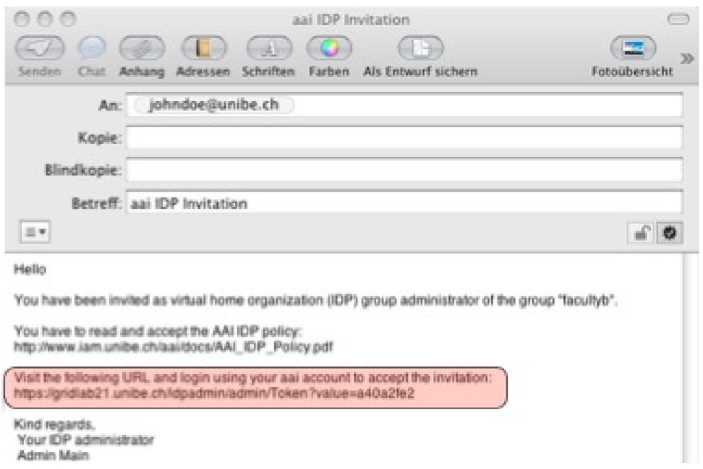
To invite the further administrator, send him an email, as shown in Fig. 8 and 9.


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout
<< Manage group
Send invitation email >>

facultyb : Invite a new group admministrator

Click on the button to send the **invitation email** by **your default email application** to the new group administrator.

Figure 8: User administration: invite administrator.



An: johndoe@unibe.ch
Kopie:
Blindkopie:
Betreff: aai IDP Invitation

Hello

You have been invited as virtual home organization (IDP) group administrator of the group "facultyb".

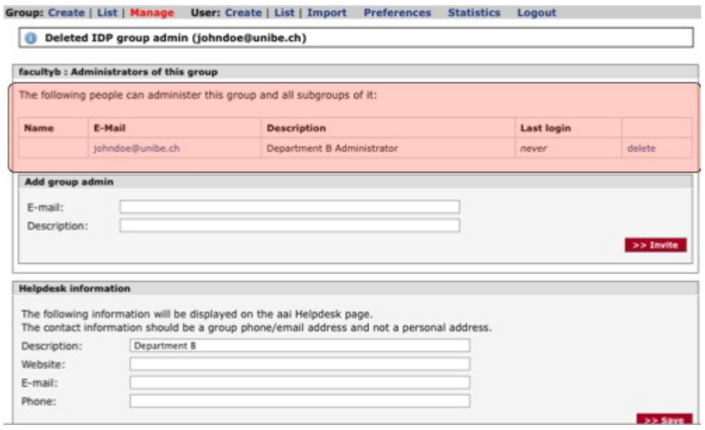
You have to read and accept the AAI IDP policy:
http://www.iam.unibe.ch/aai/docs/AAI_IDP_Policy.pdf

Visit the following URL and login using your aai account to accept the invitation:
https://gridlab21.unibe.ch/idpadmin/admin/Token?value=a40a2fe2

Kind regards,
Your IDP administrator
Admin Main

Figure 9: User administration: invitation email.

Now the user opens the URL inside the email. The URL is protected by shibboleth, so the user has to log in with its AAI login. After successful login he is automatically added as administrator for the selected group (cf. Fig.10).


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout
>> Invite
>> Save

Deleted IDP group admin (johndoe@unibe.ch)

facultyb : Administrators of this group

The following people can administer this group and all subgroups of it:

Name	E-Mail	Description	Last login	
	johndoe@unibe.ch	Department B Administrator	*never*	delete

Add group admin

E-mail:
Description:

Helpdesk information

The following information will be displayed on the aai Helpdesk page.
The contact information should be a group phone/email address and not a personal address.

Description: Department B
Website:
E-mail:
Phone:

Figure 10: User administration: admin is defined for this group.

To list and manage current user, click on *User: List*. There you get an overview about the users in the currently selected group. Groups can be selected in the *Group: List Menu*. In this menu you can manage the users:

- Edit attributes (name, email, expiration date, ...)
- Reset password.
- Expire a user immediately
- Delete a user

IMPORTANT: IT IS NOT POSSIBLE TO REMOVE AN USER FROM THE DATABASE. For security and auditing reasons (legal issues), every deleted unique ID remains in the database to ensure that no other new user can get the same unique ID.

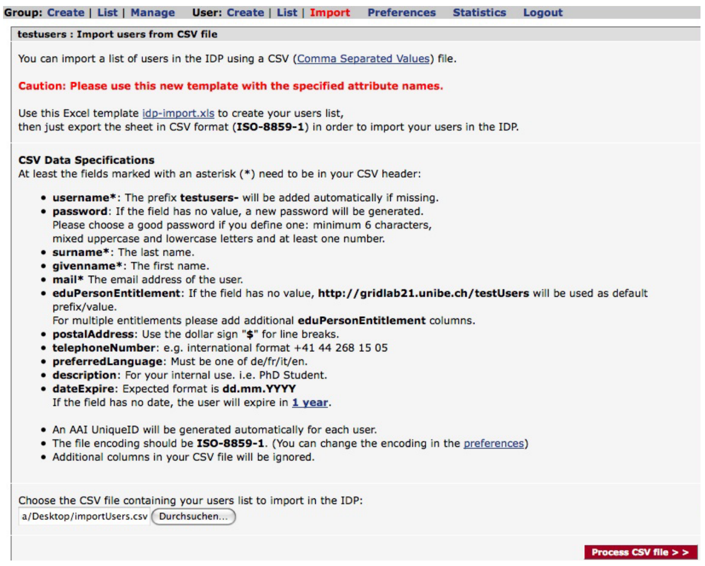
It is also possible to import users with a file formated with the CSV format (ISO-8859-1). Goto to the menu *User: Import* to get further information. The fields marked with an asterisk (*) have to be in your CSV header.

Here an example. We try to import 14 users:

```
username , password , surname , givenname , mail , postalAddress , telephoneNumber , preferredLanguage , description , dateExpire
user1 , pass1 , surname1 , givenname1 , user1@example.com, street  1,+41  44  268  01  05,en , user  1,31.12.2009
```

```
user2 , pass2 , surname2 , givenname2 , user2@example.com, street  2,+41  44  268  01  05,en , user  2,31.12.2009
user3 , pass3 , surname3 , givenname3 , user3@example.com, street  1,+41  44  268  02  05,en , user  3,31.12.2009
user4 , pass4 , surname4 , givenname4 , user4@example.com, street  1,+41  44  268  04  05,en , user  4,31.12.2009
user5 , pass5 , surname5 , givenname5 , user5@example.com, street  1,+41  44  268  05  05,en , user  5,31.12.2009
user6 , pass6 , surname6 , givenname6 , user6@example.com, street  1,+41  44  268  06  05,en , user  6,31.12.2009
user7 , pass7 , surname7 , givenname7 , user7@example.com, street  1,+41  44  268  07  05,en , user  7,31.12.2009
user8 , pass8 , surname8 , givenname8 , user8@example.com, street  1,+41  44  268  08  05,en , user  8,31.12.2009
user9 , pass9 , surname9 , givenname9 , user9@example.com, street  1,+41  44  268  09  05,en , user  9,31.12.2009
user10 , pass10 , surname10 , givenname10 , user10@example.com, street  1,+41  44  268  10  05,en , user  10,31.12.2009
user11 , pass11 , surname11 , givenname11 , user11@example.com, street  1,+41  44  268  11  05,en , user  11,31.12.2009
user12 , pass12 , surname12 , givenname12 , user12@example.com, street  1,+41  44  268  12  05,en , user  12,31.12.2009
user13 , pass13 , surname13 , givenname13 , user13@example.com, street  1,+41  44  268  13  05,en , user  13,31.12.2009
user14 , pass14 , surname14 , givenname14 , user14@example.com, street  1,+41  44  268  14  05,en , user  14,31.12.2009
```

As you can see, the password is clear text. If you enter no password, a password will be generated for every user. Select the csv-file and upload it (cf. Fig. 11).


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout

testusers : Import users from CSV file

You can import a list of users in the IDP using a CSV (Comma Separated Values) file.

Caution: Please use this new template with the specified attribute names.

Use this Excel template idp-import.xls to create your users list,
then just export the sheet in CSV format (**ISO-8859-1**) in order to import your users in the IDP.

CSV Data Specifications
At least the fields marked with an asterisk (*) need to be in your CSV header:

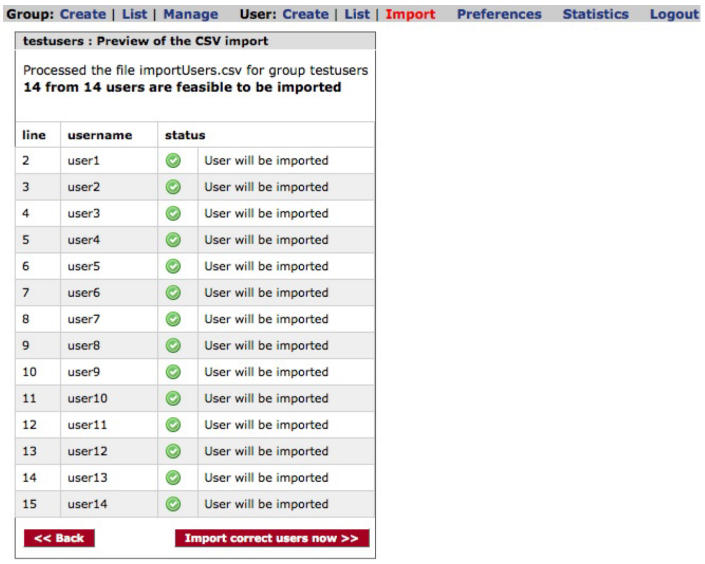
- **username***: The prefix **testusers-** will be added automatically if missing.
- **password**: If the field has no value, a new password will be generated.
  Please choose a good password if you define one: minimum 6 characters,
  mixed uppercase and lowercase letters and at least one number.
- **surname***: The last name.
- **givenname***: The first name.
- **mail*** The email address of the user.
- **eduPersonEntitlement**: If the field has no value, **http://gridlab21.unibe.ch/testUsers** will be used as default prefix/value.
  For multiple entitlements please add additional **eduPersonEntitlement** columns.
- **postalAddress**: Use the dollar sign "**$**" for line breaks.
- **telephoneNumber**: e.g. international format +41 44 268 15 05
- **preferredLanguage**: Must be one of de/fr/it/en.
- **description**: For your internal use. i.e. PhD Student.
- **dateExpire**: Expected format is **dd.mm.YYYY**
  If the field has no date, the user will expire in **1 year**.

- An AAI UniqueID will be generated automatically for each user.
- The file encoding should be **ISO-8859-1**. (You can change the encoding in the preferences)
- Additional columns in your CSV file will be ignored.

Choose the CSV file containing your users list to import in the IDP:
a/Desktop/importUsers.csv

Figure 11: User administration: import users.

Now we get an overview with the message that all 14 users are feasibly to be imported (cf. Fig. 12).


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout

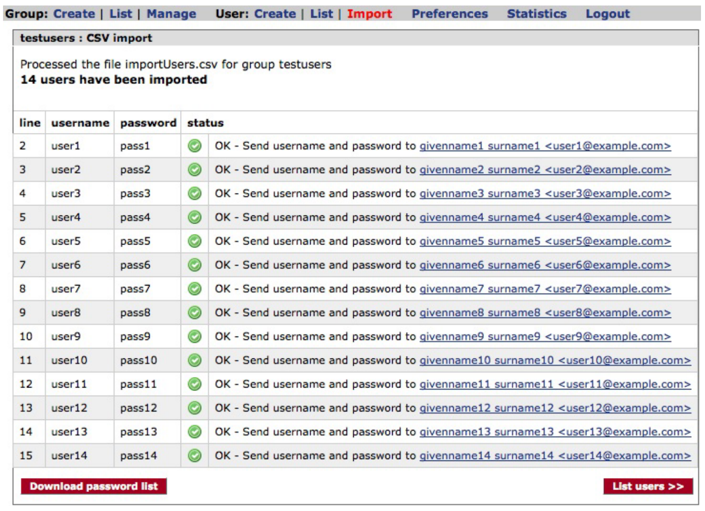
testusers : Preview of the CSV import

Processed the file importUsers.csv for group testusers
14 from 14 users are feasible to be imported

line	username	status
2	user1	✓ User will be imported
3	user2	✓ User will be imported
4	user3	✓ User will be imported
5	user4	✓ User will be imported
6	user5	✓ User will be imported
7	user6	✓ User will be imported
8	user7	✓ User will be imported
9	user8	✓ User will be imported
10	user9	✓ User will be imported
11	user10	✓ User will be imported
12	user11	✓ User will be imported
13	user12	✓ User will be imported
14	user13	✓ User will be imported
15	user14	✓ User will be imported

Figure 12: User administration: import users.

Press button *Import correct users now*. Now you can see an overview about the 8 imported users. If you press the *Download password list* button, as shown in Fig. 13 you will get a list about all given respectively automatic created passwords.


Group: Create | List | Manage User: Create | List | Import Preferences Statistics Logout
Download password list
List users >>

testusers : CSV import

Processed the file importUsers.csv for group testusers
14 users have been imported

line	username	password	status
2	user1	pass1	✓ OK - Send username and password to givenname1 surname1 <user1@example.com>
3	user2	pass2	✓ OK - Send username and password to givenname2 surname2 <user2@example.com>
4	user3	pass3	✓ OK - Send username and password to givenname3 surname3 <user3@example.com>
5	user4	pass4	✓ OK - Send username and password to givenname4 surname4 <user4@example.com>
6	user5	pass5	✓ OK - Send username and password to givenname5 surname5 <user5@example.com>
7	user6	pass6	✓ OK - Send username and password to givenname6 surname6 <user6@example.com>
8	user7	pass7	✓ OK - Send username and password to givenname7 surname7 <user7@example.com>
9	user8	pass8	✓ OK - Send username and password to givenname8 surname8 <user8@example.com>
10	user9	pass9	✓ OK - Send username and password to givenname9 surname9 <user9@example.com>
11	user10	pass10	✓ OK - Send username and password to givenname10 surname10 <user10@example.com>
12	user11	pass11	✓ OK - Send username and password to givenname11 surname11 <user11@example.com>
13	user12	pass12	✓ OK - Send username and password to givenname12 surname12 <user12@example.com>
14	user13	pass13	✓ OK - Send username and password to givenname13 surname13 <user13@example.com>
15	user14	pass14	✓ OK - Send username and password to givenname14 surname14 <user14@example.com>

Figure 13: User administration: import users.

In the menu *Preferences* you can modify several preferences, such expiration date, attributes, email settings, and more. In the menu *Statistics* you get an overview about the statistics and the possibility to maintain the database.

5 Administration of User Roles and Actions

In this Section the administration of user roles according to the local testbed is described. First, the administrator has to login with its AAI login and register for a role (e.g. TARWIS User), as shown in Section 7.2. In the next step, the AAI login of the user has to get the *Portal Administrator* role in the SNA, thus it can administrate the roles of all other AAI users.

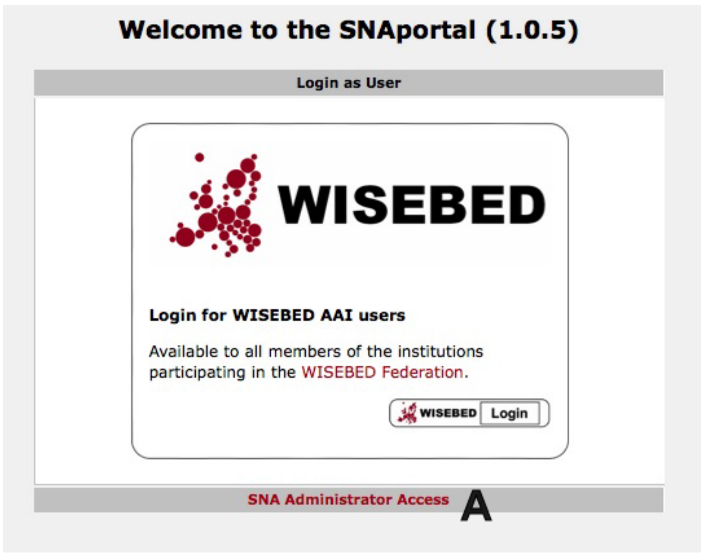
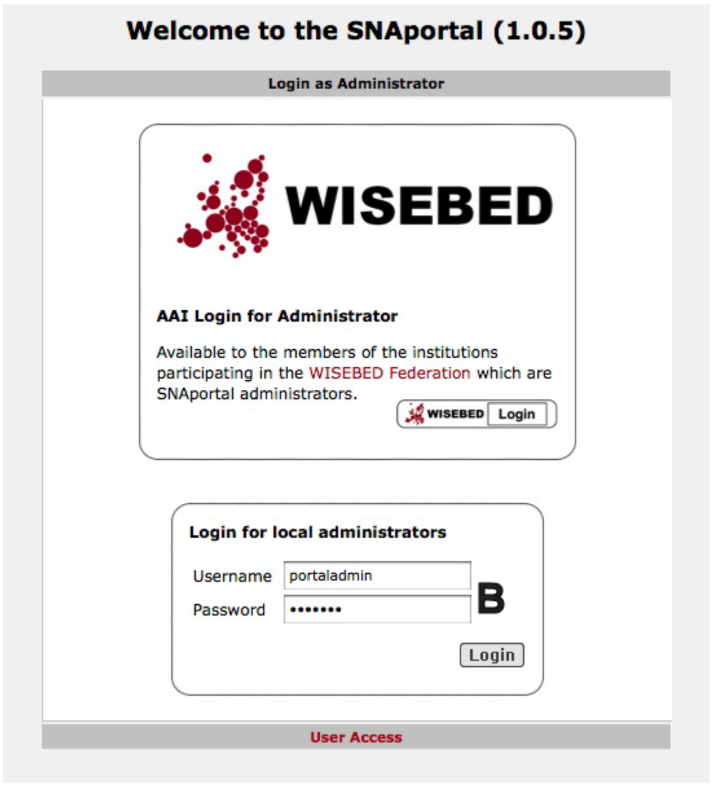

Welcome to the SNAportal (1.0.5)
Login as User
WISEBED
Login for WISEBED AAI users
Available to all members of the institutions participating in the WISEBED Federation.
WISEBED Login
SNA Administrator Access
A

Figure 14: User role administration: login as user portaladmin.

(A) First, the administrator has to login as the user *portaladmin* (cf. Fig. 14). This has to be done only once, because before the first use, no AAI user has the *Portal Administrator* rights (which are necessary to administrate the roles of all other AAI users.

(B) The administrator has to enter the credentials for the *portaladmin* (user=portaladmin, password=wisebed), as shown in Fig. 15.


Welcome to the SNAportal (1.0.5)

Login as Administrator

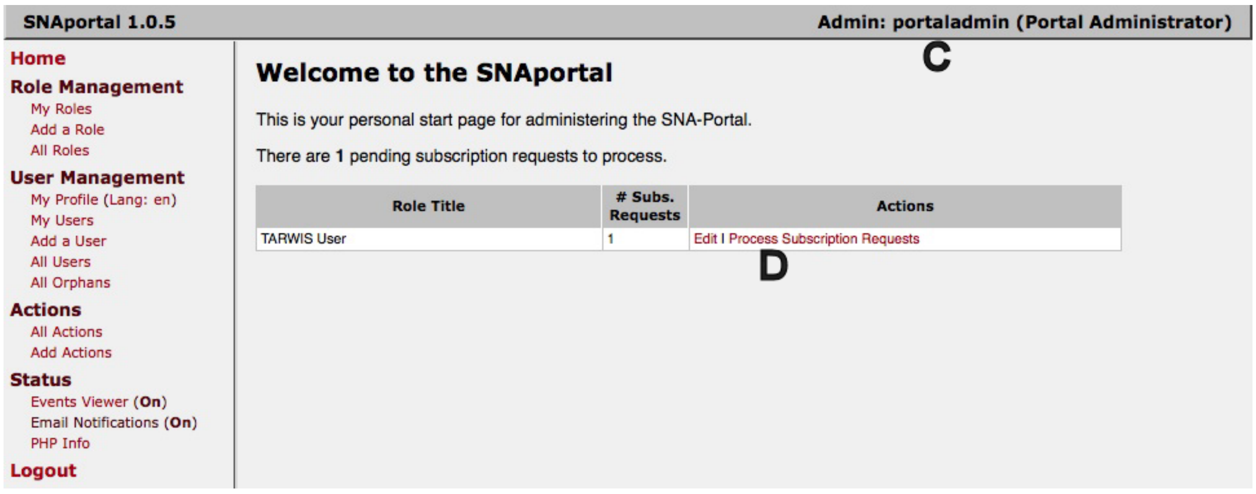
AAI Login for Administrator

Available to the members of the institutions participating in the WISEBED Federation which are SNAportal administrators.

Login for local administrators

Username: portaladmin
Password: •••••••

Figure 15: User role administration: portaladmin credentials.


SNAportal 1.0.5 — Admin: portaladmin (Portal Administrator)

Welcome to the SNAportal

This is your personal start page for administering the SNA-Portal.

There are 1 pending subscription requests to process.

Role Title	# Subs. Requests	Actions
TARWIS User	1	Edit I Process Subscription Requests

Figure 16: User role administration: administrator menu.

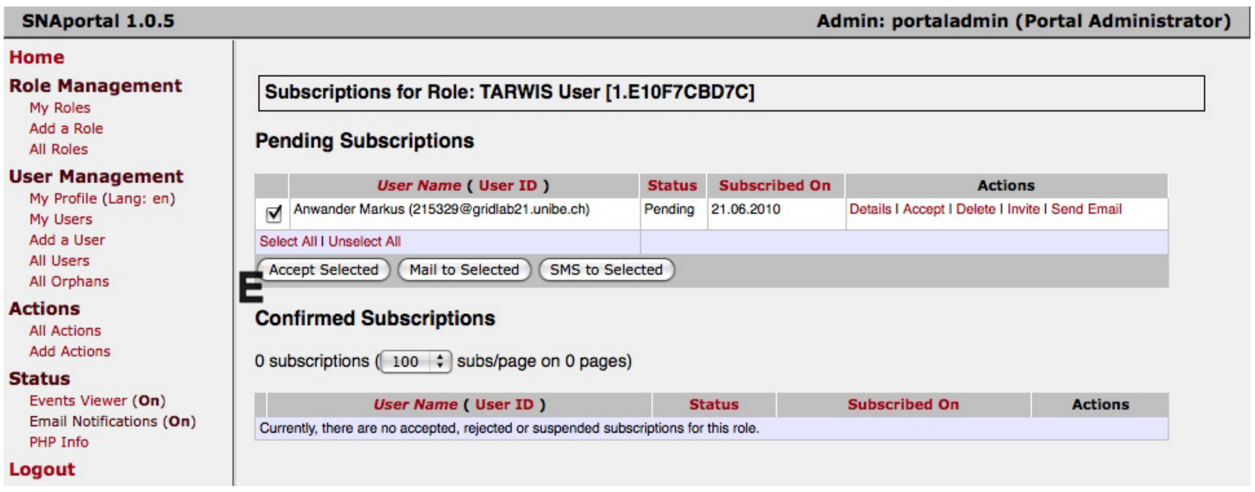
(C) Now, the user is logged in as user *portaladmin* (cf. Fig. 16).

(D) The next step it to edit the own pending subscription as TARWIS user (clicking on *Process Subscription Requests*) and

(E) accepting the request (cf. Fig. 17). This has nothing to do with the own role as *Portal Administrator*.

(F) In the next step, the administrator gets the *Portal Administrator* privilege. After clicking on *All Users*, choosing the own AAI user, clicking on *Edit*

(G) the properties of the AAI user are shown and can be edited (cf. Fig. 18).



Subscriptions for Role: TARWIS User [1.E10F7CBD7C]

Pending Subscriptions

	User Name (User ID)	Status	Subscribed On	Actions
✓	Anwander Markus (215329@gridlab21.unibe.ch)	Pending	21.06.2010	Details | Accept | Delete | Invite | Send Email

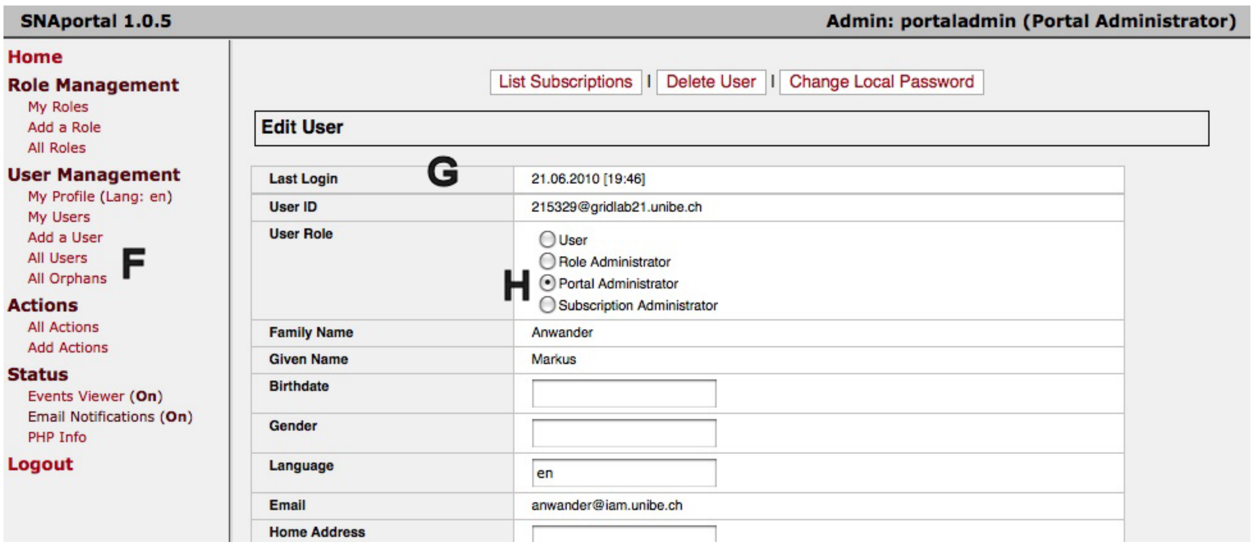
Confirmed Subscriptions

0 subscriptions (100 subs/page on 0 pages)

User Name (User ID)	Status	Subscribed On	Actions
Currently, there are no accepted, rejected or suspended subscriptions for this role.			

Figure 17: User role administration: accept own role as TARWIS user.

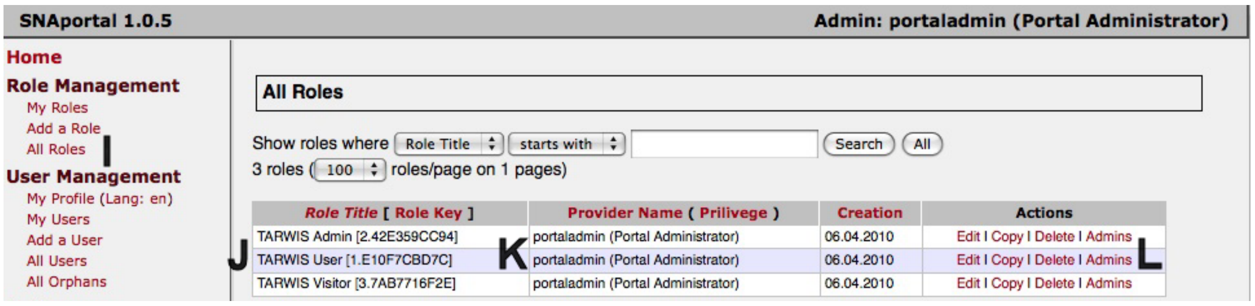
(H) After choosing *Portal Administrator* and clicking on *Save*, the AAI login of the administrator has the *Portal Administrator* privilege and can administrate the roles of all AAI users.



Edit User

Last Login	21.06.2010 [19:46]
User ID	215329@gridlab21.unibe.ch
User Role	User Role Administrator [✓ Portal Administrator] Subscription Administrator
Family Name	Anwander
Given Name	Markus
Birthdate	
Gender	
Language	en
Email	anwander@iam.unibe.ch
Home Address	

Figure 18: User role administration: change role to portaladmin.



All Roles

Show roles where Role Title starts with

3 roles (100 roles/page on 1 pages)

Role Title [Role Key]	Provider Name (Prilivege)	Creation	Actions
TARWIS Admin [2.42E359CC94]	portaladmin (Portal Administrator)	06.04.2010	Edit | Copy | Delete | Admins
TARWIS User [1.E10F7CBD7C]	portaladmin (Portal Administrator)	06.04.2010	Edit | Copy | Delete | Admins
TARWIS Visitor [3.7AB7716F2E]	portaladmin (Portal Administrator)	06.04.2010	Edit | Copy | Delete | Admins

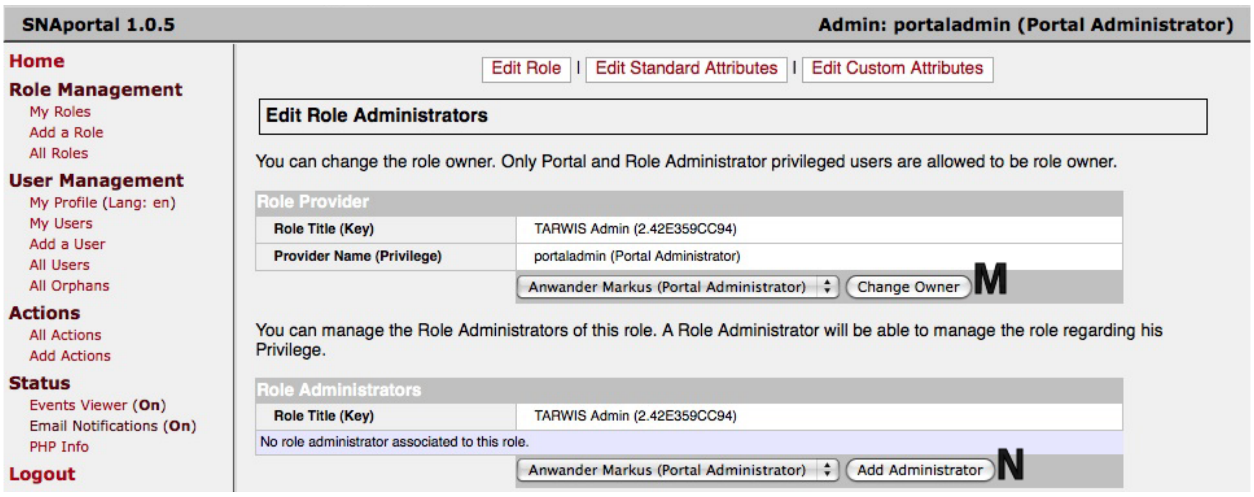
Figure 19: User role administration: all roles.

(I) By clicking on *All Roles*, as shown in Fig. 19,

(J) all roles can be seen (TARWIS Admin, TARWIS User, TARWIS Visitor are pre-configured),

(K) and the administrator of the roles. Currently it is the user portaladmin (we logged in as portaladmin, see step **(B)**).

(L) By clicking on *Admins*, we change it to the own AAI user (in step **(H)**, we gave the own AAI user the *Portal Administrator* privilege.


SNAportal 1.0.5 — Admin: portaladmin (Portal Administrator)

Home
Role Management: My Roles, Add a Role, All Roles
User Management: My Profile (Lang: en), My Users, Add a User, All Users, All Orphans
Actions: All Actions, Add Actions
Status: Events Viewer (On), Email Notifications (On), PHP Info
Logout

Edit Role | Edit Standard Attributes | Edit Custom Attributes

Edit Role Administrators

You can change the role owner. Only Portal and Role Administrator privileged users are allowed to be role owner.

Role Provider	
Role Title (Key)	TARWIS Admin (2.42E359CC94)
Provider Name (Privilege)	portaladmin (Portal Administrator)
	Anwander Markus (Portal Administrator) — Change Owner

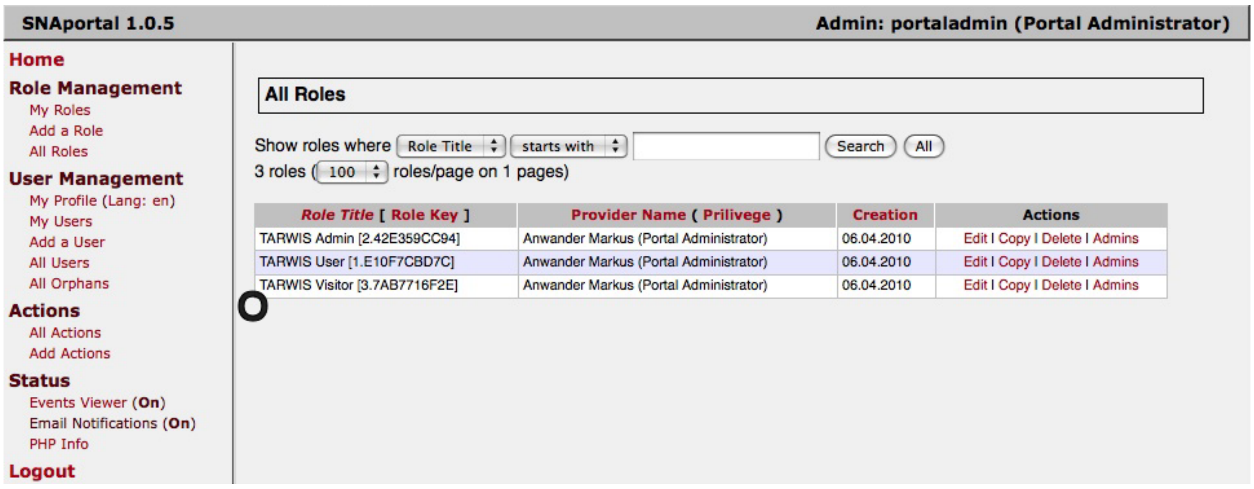
You can manage the Role Administrators of this role. A Role Administrator will be able to manage the role regarding his Privilege.

Role Administrators	
Role Title (Key)	TARWIS Admin (2.42E359CC94)
No role administrator associated to this role.	
	Anwander Markus (Portal Administrator) — Add Administrator

Figure 20: User role administration: change role owner and add role administrator.

(M) Now, the role provider (owner of the role) has to be changed from the initial portaladmin user to the own AAI user (in the current example *Markus Anwander*), as shown in Fig. 20.

(N) The own AAI user should also be added as a role administrator.


SNAportal 1.0.5 — Admin: portaladmin (Portal Administrator)

Home
Role Management: My Roles, Add a Role, All Roles
User Management: My Profile (Lang: en), My Users, Add a User, All Users, All Orphans
Actions: All Actions, Add Actions
Status: Events Viewer (On), Email Notifications (On), PHP Info
Logout

All Roles

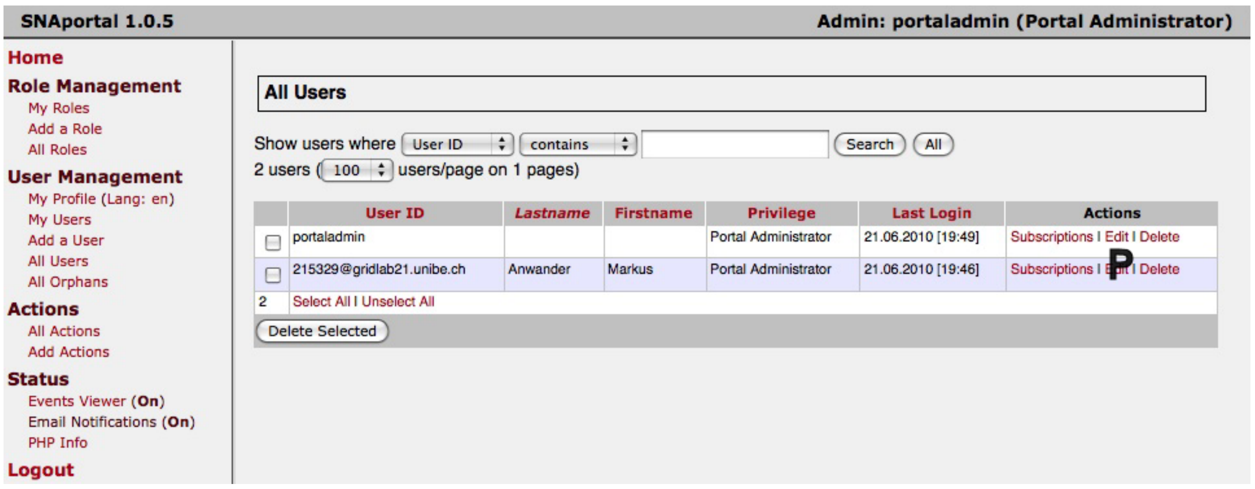
Show roles where Role Title starts with [ ] Search All
3 roles (100 roles/page on 1 pages)

Role Title [Role Key]	Provider Name (Prilivege)	Creation	Actions
TARWIS Admin [2.42E359CC94]	Anwander Markus (Portal Administrator)	06.04.2010	Edit I Copy I Delete I Admins
TARWIS User [1.E10F7CBD7C]	Anwander Markus (Portal Administrator)	06.04.2010	Edit I Copy I Delete I Admins
TARWIS Visitor [3.7AB7716F2E]	Anwander Markus (Portal Administrator)	06.04.2010	Edit I Copy I Delete I Admins

Figure 21: User role administration: allowed to administrate all three user roles.

(O) We do this for all three roles (TARWIS Admin, TARWIS User, and TARWIS Visitor). Now, the administrator has the *Portal Administrator* privilege and can administrate all three roles (cf. Fig 21).

(P) As last step, the default password of the user *portaladmin* has to be changed (cf. Fig. 22), by clicking on *Edit*. The default password can be changed using *Change Local Password*.



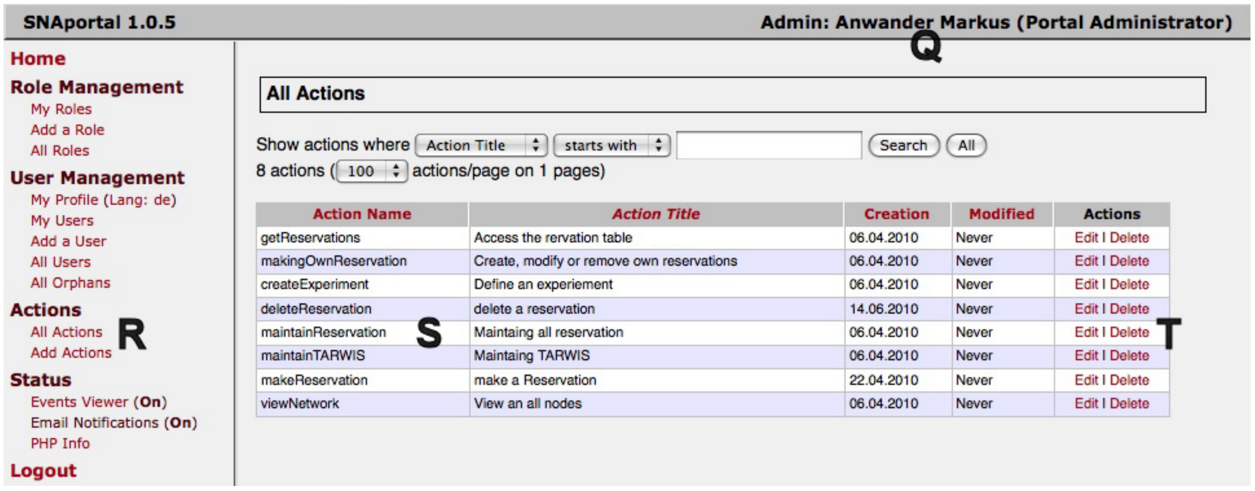
All Users

Show users where User ID contains

2 users (100 users/page on 1 pages)

	User ID	*Lastname*	Firstname	Privilege	Last Login	Actions
☐	portaladmin			Portal Administrator	21.06.2010 [19:49]	Subscriptions I Edit I Delete
☐	215329@gridlab21.unibe.ch	Anwander	Markus	Portal Administrator	21.06.2010 [19:46]	Subscriptions I Edit I Delete
2	Select All I Unselect All					

Figure 22: User role administration: change initial password of user portaladmin.



All Actions

Show actions where Action Title starts with

8 actions (100 actions/page on 1 pages)

Action Name	*Action Title*	Creation	Modified	Actions
getReservations	Access the rervation table	06.04.2010	Never	Edit I Delete
makingOwnReservation	Create, modify or remove own reservations	06.04.2010	Never	Edit I Delete
createExperiment	Define an experiement	06.04.2010	Never	Edit I Delete
deleteReservation	delete a reservation	14.06.2010	Never	Edit I Delete
maintainReservation	Maintaing all reservation	06.04.2010	Never	Edit I Delete
maintainTARWIS	Maintaing TARWIS	06.04.2010	Never	Edit I Delete
makeReservation	make a Reservation	22.04.2010	Never	Edit I Delete
viewNetwork	View an all nodes	06.04.2010	Never	Edit I Delete

Figure 23: User role administration: all actions.

(Q) Now, the administrator (currently as user portaladmin) has to be logged out and again logged in as administrator with its own AAI login (cf. Fig 23).

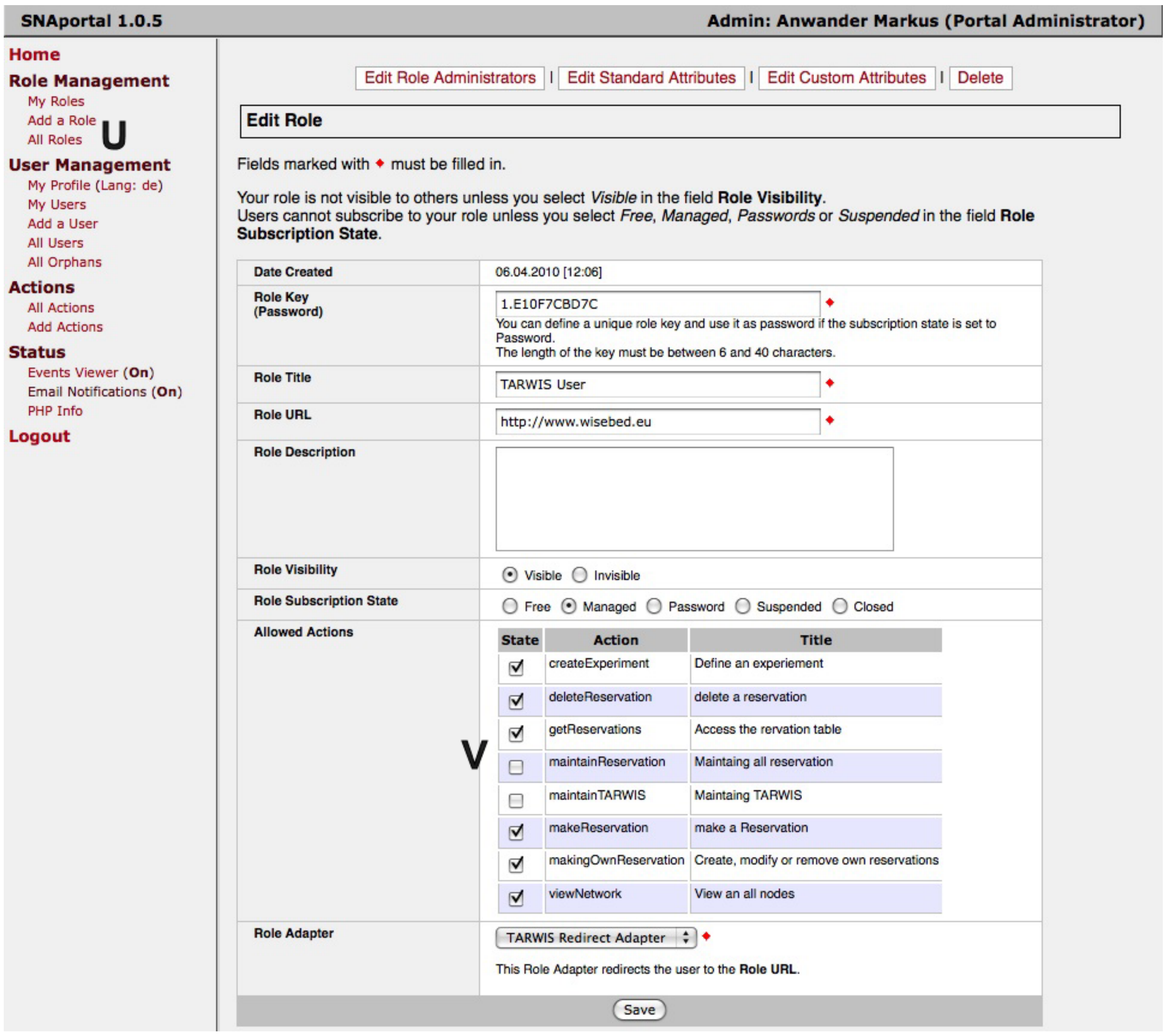
(R) By clicking on *All Actions*, the administrator can see all possible actions.

(S) Every action corresponds to a webservice supported by TARWIS and the Reservation System.

(T) Actions can be edited and deleted.

(U) By clicking on *All Roles* and afterwards on *Edit*, a role can be edited (cf. Fig 24).

(V) The important thing is the correlation between the actions and a role. In case of the role *TARWIS User*, the actions *createExperiment*, *deleteReservation*, *getReservations*, *makeReservation*, *makingOwnReservation*, and *viewNetwork* are allowed.



Edit Role

Fields marked with ♦ must be filled in.

Your role is not visible to others unless you select *Visible* in the field **Role Visibility**.
Users cannot subscribe to your role unless you select *Free*, *Managed*, *Passwords* or *Suspended* in the field **Role Subscription State**.

Date Created	06.04.2010 [12:06]
Role Key (Password)	1.E10F7CBD7C ♦ You can define a unique role key and use it as password if the subscription state is set to Password. The length of the key must be between 6 and 40 characters.
Role Title	TARWIS User ♦
Role URL	http://www.wisebed.eu ♦
Role Description	
Role Visibility	[✓ Visible] Invisible
Role Subscription State	Free [✓ Managed] Password Suspended Closed
Allowed Actions	(see below)
Role Adapter	TARWIS Redirect Adapter ♦ This Role Adapter redirects the user to the **Role URL**.

State	Action	Title
✓	createExperiment	Define an experiment
✓	deleteReservation	delete a reservation
✓	getReservations	Access the rervation table
☐	maintainReservation	Maintaing all reservation
☐	maintainTARWIS	Maintaing TARWIS
✓	makeReservation	make a Reservation
✓	makingOwnReservation	Create, modify or remove own reservations
✓	viewNetwork	View an all nodes

Figure 24: User role administration: all roles.

6 Installation of TARWIS Server, TARWIS GUI and Reservation System

This section is a guideline to set up the current version of TARWIS (Version 4.0) and further a RS-API (Reservation-System) compliant perl-based and Web-Service accessible Reservation System.

6.1 Folder Contents

After having checked out TARWIS_4.0 from the subversion repository

```
svn co https://svn.itm.uni-luebeck.de/wisebed/TARWIS/TARWIS_4.0/
```

Make sure you have the following subfolders in your directory: ”TARWISServer” (the testbed management system core logic) ”portal” (the php-based WebGUI) ”RS” (the Reservation System) ”sna-portal-1.0.5” (the Sensor Network Authorization tool)

6.2 Installation of php/perl libraries

We assume you are running a fresh Debian Lenny distro on your portal server, as requested by the Interface Description and Hardware requirements Document of December 2008.

The install script ”install.sh” in the ”TARWISServer” directory installs the necessary php/perl libraries as well as other programs and utilites necessary to run the portal server software for Debian Etch. The script has to be run as root user

REMINDER: If you use another distro, you have to search all the packages that ”install.sh” for your particular linux distro. If you do not have all the software packages listed in install.sh, the setup of the TARWIS prototype will inevitably fail.

Go to the directory ./TARWISServer and run

```
cd TARWISServer
sh install.sh
```

Please follow the installation instructions attentively and install failed packets manually (e.g. by using perl’s packet manager cpan)

6.3 TARWIS.conf

TARWIS reads the essential configuration parameters from the file TARWIS.conf. This file is given in the top-level directory and

a) needs to be moved to /etc/TARWIS.conf and be made readable for the users www-data and root (as root, run ”chmod 644 /etc/TARWIS.conf)
b) needs to be **ADAPTED** for **YOUR PARTICULAR SITE**

We briefly go through the variables in the TARWIS Server configuration file (/etc/TARWIS.conf)

```
####################################
# TARWIS Server Configuration File #
####################################

# VARIABLES RELATED TO THE TARWIS SERVER COMPONENT

ORGANIZATION_NAME=University of BERN  # obviously, write which is your university affiliation
ORGANIZATION_ACRONYM=UBERN            # use the ACRONYM used by the WISEBED Proposal
TESTBED_NAME=UBERNTestbed             # find a fancy name for your Testbed
MYSQL_DATABASE_NAME=wsn_info          # do not change this entry
MYSQL_DATABASE_HOST=130.92.66.182     # the host of the database, either bind it to the ip of eth0 or use localhost
MYSQL_DATABASE_USER=wisebeduser       # please use the user ”wisebeduser”
MYSQL_DATABASE_PASSWORD=wisebed       # use a password for the user ”wisebeduser”
                                      # the script ”createWisebedMySQLUser.sh” sets the password
                                      # ”wisebed” — you are free to change it

TARWIS_DIRECTORY=/opt/TARWISServer/   # the directory where the TARWIS server and the correspondign code is
SESSION_MANAGEMENT_SERVICE_URL=http://130.92.66.182:3000 # the SESSION MANAGEMENT SERVICE URL
CONTROLLER_SERVICE_URL=http://130.92.66.182:3025         # the URL of the CONTROLLER SERVICE

TARWIS_GUI_URL=https://gridlab23.unibe.ch/portal/TARWIS  # where TARWIS GUI is accessible over HTTP
SOAP_TIMEOUT=20                       # how many seconds the SOAP call is allowed to last until the response (deprecated)
FLASH_ATTEMPTS=2                      # how many times the system shall attempt to flash the nodes
WAIT_PERIOD_BEFORE_FLASH_RETRY=50     # how long TARWIS shall wait before reattempting to flash ndoes with negative flash status
WAIT_PERIOD_AFTER_REBOOT=20           # how long it takes a node to reboot after a hard-reset (UBERN ~60s)
                                      # This time only specifies how long TARWIS maximally waits for the asynchronous status replies.
                                      # If replies return earlier, TARWIS continues of course

TOPOLOGY_UPDATE_INTERVAL=0            # how often is ”getNeighborhood” called for every neighbor? (CURRENTLY DISABLED)
PROPERTY_UPDATE_INTERVAL=0            # how often is ”getPropertyValueOf” called for every capability?
ALIVE_STATE_UPDATE_INTERVAL=0         # how often is ”areNodesAlive” called?
ALIVE_STATE_REQUEST_MAXIMUM_RESPONSE_TIME=20 # how long shall be waited for the reply?

DEFAULT_IMAGE_PATH=/opt/TARWISServer/noExperiment.ihex
                                      # specify the location of the default image that will be flashed to every node after every
                                           experiment
DEFAULT_IMAGE_PLATFORM=Contiki        # specify the platform of this default image

USE_JAX_IN_BACKEND=0                  # Do we use JAVA JAX in the backend? If yes, we need a workaround
DEBUG_MODE=0                          # WITH DEBUG_MODE, the entire SOAP-TRACE stuff is written into the .err files


############################################
# VARIABLES RELATED TO THE WEB-BASED GUI #
############################################
```

```
# variables to specify the coordinate system that is projected onto the image containing the
# map of the sensor network testbed (x, y, z in pixel and a in degree)

MAP_SIZE_X=390                      # check out "map_parameters.pdf"
MAP_SIZE_Y=172                      # to see how parameters x,y,z and angle are
MAP_SIZE_Z=455                      # specified
MAP_ANGLE=45

MAP_REFRESH_INTERVAL=30000          # refreshing interval of the map in the Experiment Monitoring Section (in ms)
MAP_PATH=../pics/maps               # path of the images of the maps relative to the folder named "TARWIS" inside "portal"
MAP_FILE_NAME=3D_nobuilding.png     # file name of the map file (default template)

MAX_DISPLAYABLE_NODES=200           # specify the max. number of nodes that are being loaded
SCALE_NODES_IN_GUI=0                # specify whether to scale the nodes to positions in between [0,100] in the GUI
SMTP_RELAY=asterix.unibe.ch         # the smtp relay that is used to send the notification emails
UPDATE_LINES_TARWIS_GUI=10          # how many lines of output the GUI shall update
```

6.4 MYSQL USER setup

TARWIS needs MYSQL to be present at the portal server. You need to know the password for your root mysql user. If you have not defined a password for the root mysql user.
Proceed as follows: run "resetDB.sh" to set up the table definitions.
"resetDB.sh" first prompts you to set up the root mysql user password.

Enter "y" to set the root mysql user password now. Please remember this password!

If you have ALREADY set your root mysql user password, enter "n" when the script asks you this question.

(make sure you are inside the directory TARWISServer when running this script)

```
cd TARWISServer
sh resetDB.sh
```

"resetDB.sh" reads in the table definitions given in wisebed.sql and sets up the database schema.

The script "createWisebedMySQLUser.sh" generates a new MySQL user with username="wisebeduser" and password="wisebed".

Please run the script and enter the password of the root mysql user (!) in order to set up the "wisebeduser"

```
cd TARWISServer
sh createWisebedMySQLUser.sh
```

6.5 TARWISServer setup

Move the directory "TARWISServer" somewhere and make sure you specify the location correctly in the TARWIS_DIRECTORY variable in /etc/TARWIS.conf The directory specified in TARWIS_DIRECTORY variable must correctly point to your location of the directory TARWISServer.

We suggest you use /opt/TARWISServer, but you are free to change it.

```
cd TARWIS_4.0
cp -r ./TARWISServer /opt
```

Then, go to the TARWISServer folder and run the script "change_URL_in _WSDL_files.sh"

```
cd /opt/TARWISServer/
sh change_URL_in_WSDL_files.sh
```

6.6 WebGUI-System setup

Move the directory "portal" to the apache folder that is accessible from the outside, (most likely /var/www/).

If you already have something there, remove this directory first (rm -rf /var/www/portal) !

```
cd TARWIS_4.0
cp -r ./portal/ /var/www
```

Change the ownership of /var/www/portal such that php/apache can access all files

```
chown -R www-data:www-data /var/www/portal
```

Test if the access to this directory works with your browser (go to https://¡your_portal_server_url¿/portal/TARWIS). When you access the directory, you should be first prompted to the Shibboleth. Login Interface, and then get to the overview screen of the Testbed Management System of TARWIS.

This step requires the Shibboleth Service Provider to be installed on this machine (check ”ps aux — grep shibd” if the daemon is running). If you do not have this software installed yet, check out the Section 2 in this document how to do that.

6.6.1 Logo

Every partner should display its logo in the top right corner. This facilitates the identification of the portal that is currently being displayed.

For this reason, go to

```
cd /var/www/portal/TARWIS/pics
```

and edit the file logo.jpg

Please do not change the size of the file as this impacts on the layout of the size. Rather resize your logo instead.

6.7 RS Setup

TARWIS reads the essential configuration parameters for the reservation process from the file /etc/RS.conf. This file is given in the top-level directory and
a) needs to be moved to /etc/RS.conf and be made readable for the users www-data and root (as root, run ”chmod 644 /etc/RS.conf)
b) needs to be **ADAPTED** for **YOUR PARTICULAR SITE**

We briefly go through the variables of the RS configuration file (/etc/RS.conf):

```
RESERVATION_SYSTEM_SERVICE_URL=http://130.92.66.182:3100      # the url where this service runs
RESERVATION_SYSTEM_DIRECTORY=/opt/RS/                         # the directory of the perl-based RS
URN_PREFIX=gridlab21.unibe.ch                           # the hostname of the machine where the RS is accessible from the outside

RS_MYSQL_DATABASE_NAME=rs_info                          # do not change this entry
RS_MYSQL_DATABASE_HOST=localhost                        # It's the name of the database containing all the tables
                                                        # the host of the database, in most cases "localhost"
RS_MYSQL_DATABASE_USER=wisebeduser                      # please use the user "wisebeduser"
RS_MYSQL_DATABASE_PASSWORD=wisebed                      # use a password for the user "wisebeduser"
                                                        # the script resetDB.sh sets the password
                                                        # "wisebed" - you are free to change it
```

Then, proceed with the RS subfolder:
Move the directory ”RS” somewhere and make sure you specify the location correctly in the RESERVATION_SYSTEM_DIRECTORY variable in /etc/RS.conf. The directory specified in the variable must correctly point to your location of your directory containing the Reservation System

We suggest you use /opt/RS, but you are free to change it.

```
cd TARWIS_4.0
cp -r RS /opt
```

Before starting the Reservation System for the first time, run ”resetDB.sh” in this folder. This will set up the database for reservations:

```
cd /opt/RS
sh resetDB.sh
```

Then, go to the TARWISServer folder and run the script ”change_URL_in_WSDL_files.sh”

```
cd /opt/RS/
sh change_URL_in_WSDL_files.sh
```

To start the Reservation System, type:

```
cd /opt/RS
sh runReservationSystem.sh
```

6.8 DUMMY_TESTBED_IMPLEMENTATION

In order to test the ”cleanliness” and ”correctness”, as well as the ”language-independence” of the WebService Interfaces, we implemented a so-called ”DUMMY-Implementation” of these interfaces, essentially those that need to be implemented by each partner. The implementation of these interfaces mimick a ”dummy network”, and this implementation correctly talks to the web-services interfaces of the TARWISServer.

In this release of TARWIS, you can find a subdirectory called ”DUMMY_TESTBED_IMPLEMENTATION” in the directory TARWISServer.

6.9 Testing the Installation Setup Progress

We test the installation setup progress using the ”generic” TARWISServer on top of the ”DUMMY_TESTBED_IMPLEMEN

Make sure you have installed ruby and all the necessary libraries using install.sh (check point 1). If you have correctly set the parameters in /etc/TARWIS.conf, and changed the ip addresses in the WSDL Files, you can start the DUMMY_TESTBED_IMPLEMENTATION that then starts serving the WSNService WebService.

```
cd /opt/TARWISServer/DUMMY_TESTBED_IMPLEMENTATION/
sh runDummyTestbedImplementation.sh
```

After that, you can start the TARWISServer, which should then successfully start and use those interfaces. You can then go to the WebGUI in order to check if the GUI operates properly and if you can schedule an experiment and let it execute.

```
cd /opt/TARWISServer
sh runTARWISServer.sh
```

If you start an experiment by using TARWIS on top of the DUMMY_TESTBED_IMPLEMENTATION, it apparently will do nothing useful but only interact with the dummy-nodes, but it will illustrate you the interaction between the Testbed Implementation and the TARWISServer, hence give you an idea how to use WebServices to interconnect your particular Testbed Implementation with the Testbed Management System TARWIS.

You can then inspect the code in DUMMY_TESTBED_IMPLEMENTATION to start working on the code to interconnect your particular testbed to TARWIS.

6.10 Stopping/Killing/Cleaning Up

In oder to stop the TARWISServer, there's a script called "killall.sh" that kills all the perl processes of TARWISServer.

```
cd /opt/TARWISServer
sh killall.sh
```

To clean up all .out and .err files (TARWISServer redirects STDOUT and STDERR to *.out and *.err files for debugging purposes), run "cleanOutFilesErrorFiles.sh"

```
cd /opt/TARWISServer
sh cleanOutFilesErrorFiles.sh
```

To stop the DUMMY_TESTBED_IMPLEMENTATION Webservices, go to the folder DUMMY_TESTBED_IMPLEMENT
and run

```
cd /opt/TARWISServer/DUMMY_TESTBED_IMPLEMENTATION
sh killall.sh
```

6.11 Init-script

We use a init-script for starting and stopping all wisebed-related testbed software. You can re-use this script if you wish, it's contained in the subfolder INIT of the TARWIS_4.0 folder

6.12 Icons for your nodes

Put a small icon-sized image of type png to your /var/www/portal/TARWIS folder and it will be used for the testbed map.

The name of the picture must be the same as the name used in the ¡nodeType¿ tag of your testbed WiseML description
/var/www/portal/TARWIS/pics/¡YOUR_TYPE¿.png

7 Before You Start Your Experiment

First, the user has to be member of the WISEBED federation. To get a WISEBED login, the user has to ask the administrator of its home organization (which is a partner in the WISEBED federation). Every partner is responsible for its own users!

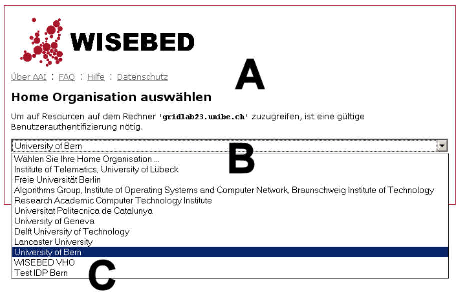
7.1 User Login

To access the TARWIS GUI the user needs to be authenticated using its WISEBED login. By accessing the TARWIS GUI with a web browser the user is directed to the two-step login process (cf. Fig. 25 and 26).

Step 1:

(A) First, the user is redirected to the WAYF (Where Are You From) server.

(B) There, the user has to select its home organization (e.g. *University of Bern*).


WISEBED

Über AAI : FAQ : Hilfe : Datenschutz

Home Organisation auswählen

Um auf Resourcen auf dem Rechner 'gridlab23.unibe.ch' zuzugreifen, ist eine gültige Benutzerauthentifizierung nötig.

University of Bern

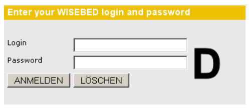
Wählen Sie Ihre Home Organisation ...
Institute of Telematics, University of Lübeck
Freie Universität Berlin
Algorithms Group, Institute of Operating Systems and Computer Network, Braunschweig Institute of Technology
Research Academic Computer Technology Institute
Universitat Politecnica de Catalunya
University of Geneva
Delft University of Technology
Lancaster University
University of Bern
WISEBED VHO
Test IDP Bern

Figure 25: Login: select home organization.

(C) If the user has no own home organization, it can select a so called virtual home organization (in this example it is *WISEBED VHO*).

Step 2:

(D) Now, the user is directed to the login dialog (cf. Fig. 26) of its own home organization and has to enter its credentials (from its WISEBED login).


Enter your WISEBED login and password

Login
Password

Figure 26: Login: enter the credentials.

7.2 User Roles

After login the user is redirected to the TARWIS GUI. To perform experiments on the testbed the user requires access rights, a so called *User Role*. It is not enough to be part of the WISEBED federation! On the *Welcome* site, the user can see its own roles (cf. Fig. 27).

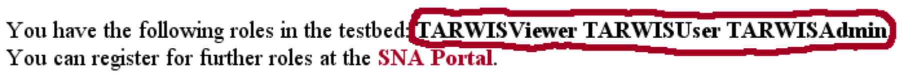
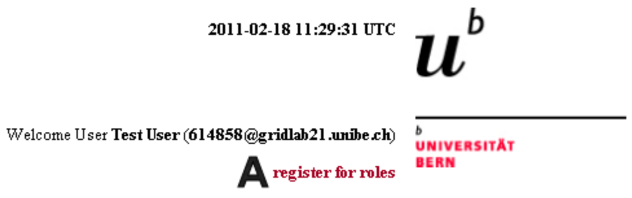

You have the following roles in the testbed: TARWISViewer TARWISUser TARWISAdmin
You can register for further roles at the SNA Portal.

Figure 27: User roles: current roles.

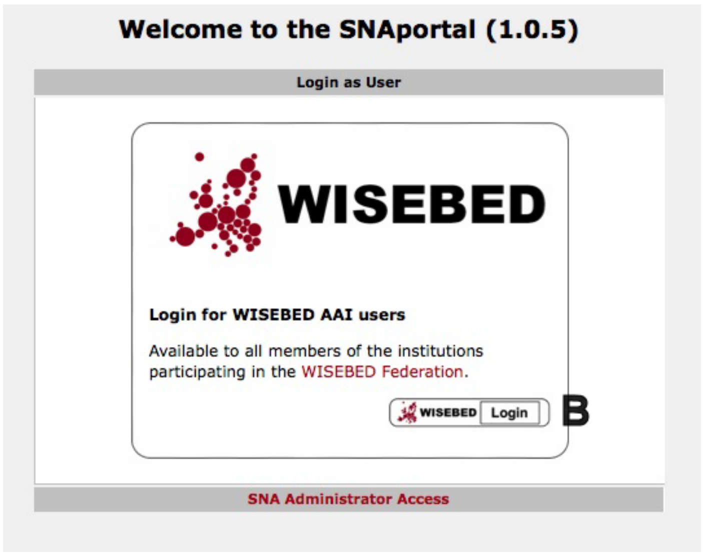
Possible roles are *TARWIS Viewer*, *TARIWS User*, and *TARWIS Admin*. A TARWIS Viewer is allowed to monitor public experiments and download theirs results. It is not allowed to reserve sensor nodes and perform own experiments. A TARWIS User can reserve sensor nodes nodes from the testbed and perform experiments. A TARWIS Admin can perform all administrative tasks.


2011-02-18 11:29:31 UTC

Welcome User Test User (614858@gridlab21.unibe.ch)

Figure 28: User roles: register for roles.

User roles can be obtained by the following steps:


Welcome to the SNAportal (1.0.5)

Login as User

Login for WISEBED AAI users

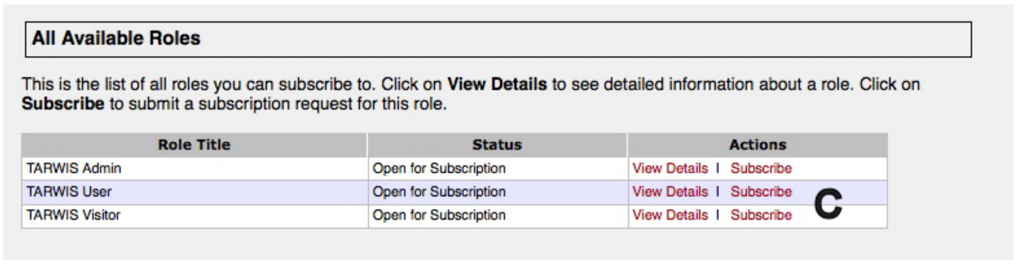
Available to all members of the institutions participating in the WISEBED Federation.

Figure 29: User roles: SNA portal login.

(A) By clicking on *register for roles* on the right top of the site (cf. Fig. 28), the user can register itself for roles.

(B) The user is directed to the SNA (Sensor Network Authorization) portal, shown in Fig. 29, where the user has to login again (using its wisebed login).

(C) The user can now subscribe for a role (cf. Fig. 30).


All Available Roles

This is the list of all roles you can subscribe to. Click on **View Details** to see detailed information about a role. Click on **Subscribe** to submit a subscription request for this role.

Role Title	Status	Actions
TARWIS Admin	Open for Subscription	View Details | Subscribe
TARWIS User	Open for Subscription	View Details | Subscribe
TARWIS Visitor	Open for Subscription	View Details | Subscribe

Figure 30: User roles: available roles.

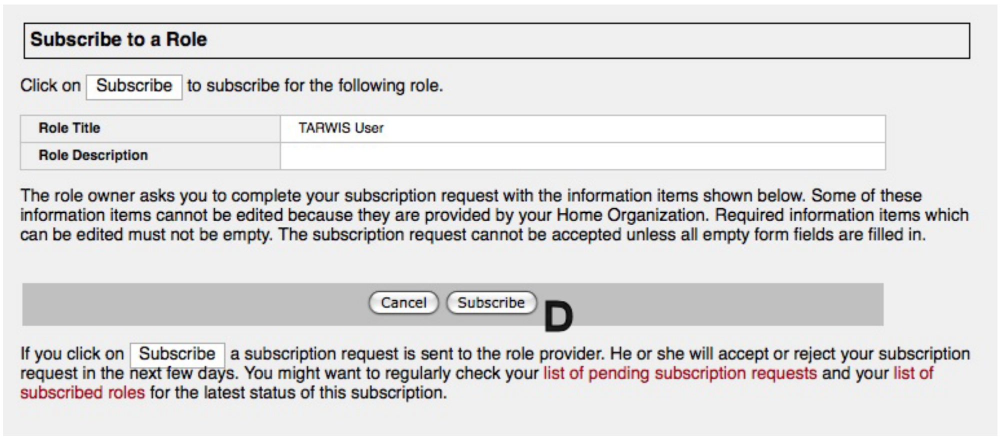
(D) By clicking on the *Subscribe* button (cf. Fig. 31), the role administrator will get a notification about the request. The user gets informed by email, when it is accepted or not accepted for the requested role.

(E) By clicking on *List My Roles* in the left menu, the user can see its roles, including the status (cf. Fig. 32).

(F) The user can also *Unsubscribe* for a role.

7.3 Binary Code Image for Experiments

The behavior of the sensor nodes only depends on the software running on the nodes. This software is developed by the user. There is no pre-installed software.



Subscribe to a Role

Click on Subscribe to subscribe for the following role.

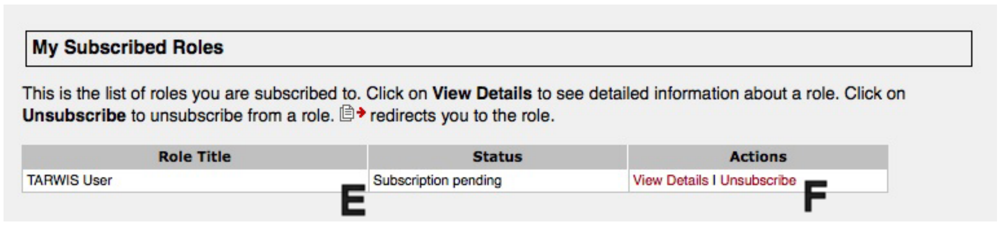
Role Title	TARWIS User
Role Description	

The role owner asks you to complete your subscription request with the information items shown below. Some of these information items cannot be edited because they are provided by your Home Organization. Required information items which can be edited must not be empty. The subscription request cannot be accepted unless all empty form fields are filled in.

Cancel Subscribe

If you click on Subscribe a subscription request is sent to the role provider. He or she will accept or reject your subscription request in the next few days. You might want to regularly check your list of pending subscription requests and your list of subscribed roles for the latest status of this subscription.

Figure 31: User roles: subscribe for a selected role.



My Subscribed Roles

This is the list of roles you are subscribed to. Click on **View Details** to see detailed information about a role. Click on **Unsubscribe** to unsubscribe from a role. redirects you to the role.

Role Title	Status	Actions
TARWIS User	Subscription pending	View Details | Unsubscribe

Figure 32: User roles: list of subscriptions.

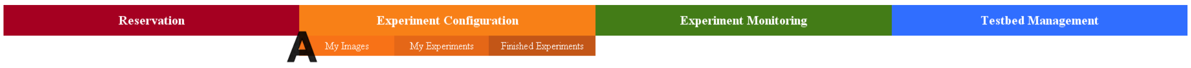
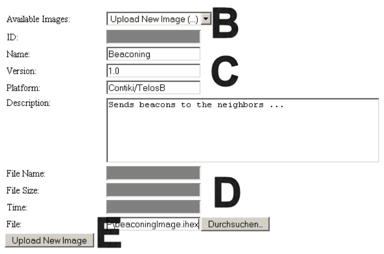


Figure 33: Image: menu.



Available Images:	Upload New Image (...)
ID:	
Name:	Beaconing
Version:	1.0
Platform:	Contiki/TelosB
Description:	Sends beacons to the neighbors ...
File Name:	
File Size:	
Time:	
File:	P:\beaconingImage.ihex

Figure 34: Image: upload.

(A) By clicking on the *Experiment Configuration* → *My Images* tab (cf. Fig. 33), the user can upload its software to the TARWIS GUI

(B) Selecting *Upload New Image (...)*, the user can upload its developed binary code image (cf. Fig 34).

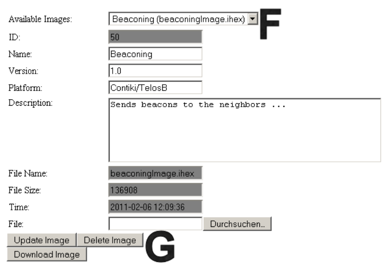
(C) Afterwards, the user can enter the name, version, platform, and description of the image.

(D) Then, the user selects the file of the image.

(E) Finally, the user finish the form by clicking on the *Upload New Image* button. The image is now uploaded to the TARWIS GUI and can be used for the experiments (see Section 8.2).

(F) By selecting a previously uploaded image, the user can update the name, version, description, and the platform (cf. Fig. 35).

(G) By clicking on the according button, the user can update, delete or download the image.



Available Images:	Beaconing (beaconingImage.ihex)
ID:	50
Name:	Beaconing
Version:	1.0
Platform:	Contiki/TelosB
Description:	Sends beacons to the neighbors ...
File Name:	beaconingImage.ihex
File Size:	136908
Time:	2011-02-06 12:09:36
File:	

Figure 35: Image: update.

8 How to Perform an Experiment on TARWIS?

This Section describes the steps how a user can perform its experiment. It starts with the nodes reservation, and afterwards the experiment configuration. Finally it describes the experiment monitoring and downloading the experiment results.

8.1 Node Reservation

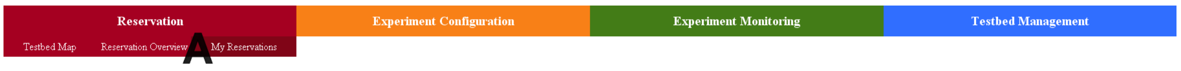
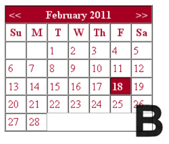


Figure 36: Reservation: menu.

(A) By clicking on the *Reservation → Reservation Overview* tab (cf. Fig. 36) the user can reserve nodes of the underlying testbed using the schedule sheet



Su	M	T	W	Th	F	Sa
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28					

Figure 37: Reservation: calendar sheet.

(B) First, the user can select the day, when its experiment should performed (cf. Fig. 37). The current day is preselected.

(C) The available sensor nodes are listed on the left side of the schedule sheet (cf. Fig. 38). Different types of sensor nodes are separated by a white line.

(D) On the top of the schedule sheet the 24 hours of the day are listed (in UTC) and divided into 15 minute slots.

(E) The state of the nodes is depicted by the color of the sheet. Available slots are colored green, blocked ones red and own reservations blue. Slots in the past are gray colored.

(F) To reserve sensor nodes for an experiment the user can select nodes in the green area. To achieve this you can either click&drag to select a rectangle (nodes vs. time) and double-click on a node to remove this node from the selected nodes of the rectangle. Or the user can select (single-click) the first and the last time slots.

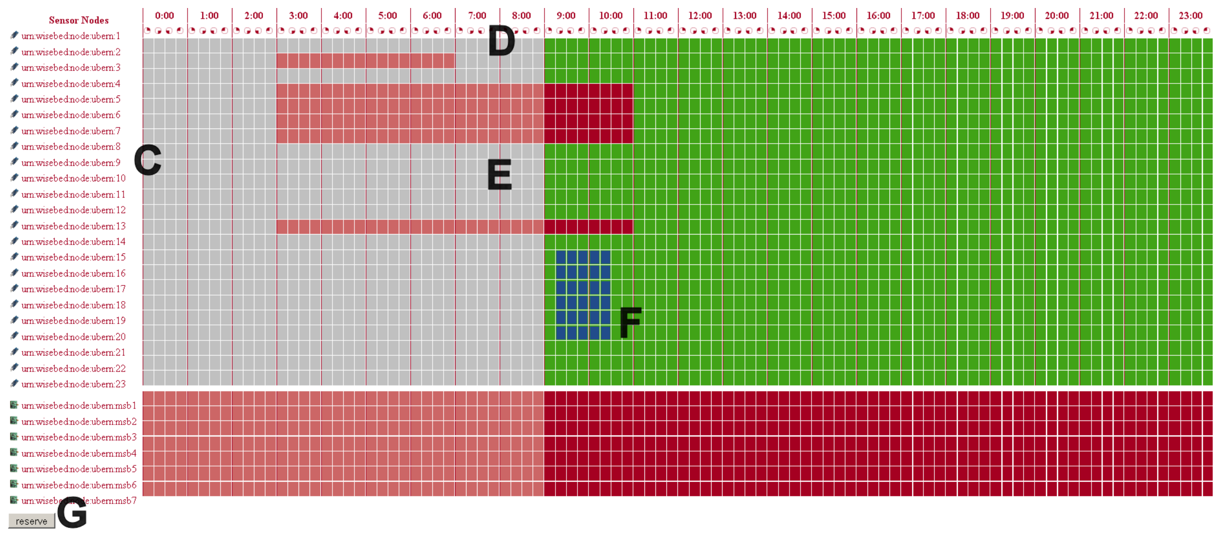

Sensor Nodes
0:00
1:00
2:00
3:00
4:00
5:00
6:00
7:00
8:00
9:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
urn:wisebed:node:ubern:1
urn:wisebed:node:ubern:2
urn:wisebed:node:ubern:3
urn:wisebed:node:ubern:4
urn:wisebed:node:ubern:5
urn:wisebed:node:ubern:6
urn:wisebed:node:ubern:7
urn:wisebed:node:ubern:8
urn:wisebed:node:ubern:9
urn:wisebed:node:ubern:10
urn:wisebed:node:ubern:11
urn:wisebed:node:ubern:12
urn:wisebed:node:ubern:13
urn:wisebed:node:ubern:14
urn:wisebed:node:ubern:15
urn:wisebed:node:ubern:16
urn:wisebed:node:ubern:17
urn:wisebed:node:ubern:18
urn:wisebed:node:ubern:19
urn:wisebed:node:ubern:20
urn:wisebed:node:ubern:21
urn:wisebed:node:ubern:22
urn:wisebed:node:ubern:23
urn:wisebed:node:ubern:msb1
urn:wisebed:node:ubern:msb2
urn:wisebed:node:ubern:msb3
urn:wisebed:node:ubern:msb4
urn:wisebed:node:ubern:msb5
urn:wisebed:node:ubern:msb6
urn:wisebed:node:ubern:msb7
reserve
C
D
E
F
G

Figure 38: Reservation: overview.

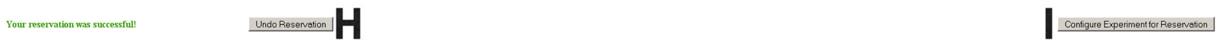

Your reservation was successful!
Undo Reservation
H
I
Configure Experiment for Reservation

Figure 39: Reservation: undo reservation or configure experiment.

(G) By click on the *reserve* button on the bottom of the page the reservation is finished.

(H) Afterwards it is depicted, if the reservations was successful or not. Now, the user can either undo the reservation (button *Undo Reservation*) or

(I) go directly to the experiment configuration tab, clicking the button *Configure Experiment for Reservation* (cf. Fig. 39).

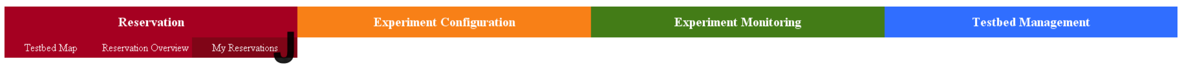
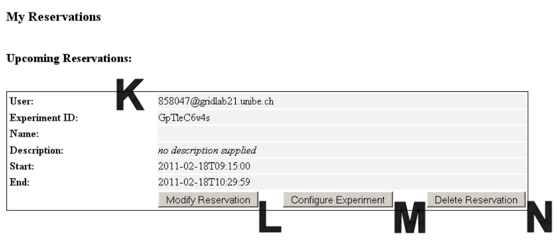


Figure 40: Reservation: menu.

(J) By clicking on the *Reservation* → *My Reservations* tab, the user can find its own reservations (cf. Fig. 40).



My Reservations

Upcoming Reservations:

User:	858047@gridlab21.unibe.ch
Experiment ID:	GpTleC6v4s
Name:	
Description:	*no description supplied*
Start:	2011-02-18T09:15:00
End:	2011-02-18T10:29:59

Figure 41: Reservation: my reservations.

(K) The unique ID of the experiment owner, the ID of the experiment, the name and description (if available), and the start- and end-time of the experiment are displayed.

(L) The user can modify the reservation (button *Modify Reservation*) or

(M) can configure the experiment (button *Configure Experiment*) or

(N) delete the reservation (button *Delete Experiment*), see Section 8.2.

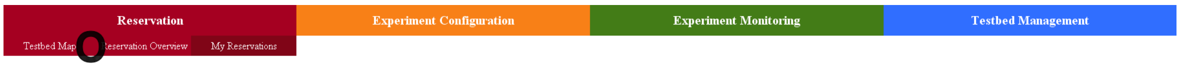
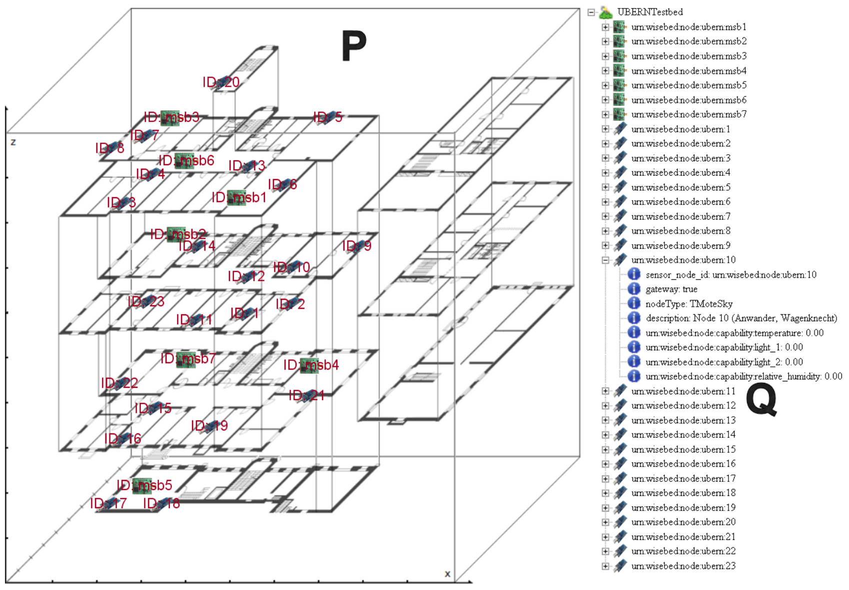


Figure 42: Reservation: menu.

(O) By clicking on the *Reservation* → *Testbed Map* tab (cf. 42),

(P) the user can find the map of the positions of all nodes of the testbed (cf. Fig. 43).

(Q) On the right side, there are additional information about the nodes (ID, type, description, capabilities).



```
UBERNTestbed
  urn:wisebed:node:ubern:msb1
  urn:wisebed:node:ubern:msb2
  urn:wisebed:node:ubern:msb3
  urn:wisebed:node:ubern:msb4
  urn:wisebed:node:ubern:msb5
  urn:wisebed:node:ubern:msb6
  urn:wisebed:node:ubern:msb7
  urn:wisebed:node:ubern:1
  urn:wisebed:node:ubern:2
  urn:wisebed:node:ubern:3
  urn:wisebed:node:ubern:4
  urn:wisebed:node:ubern:5
  urn:wisebed:node:ubern:6
  urn:wisebed:node:ubern:7
  urn:wisebed:node:ubern:8
  urn:wisebed:node:ubern:9
  urn:wisebed:node:ubern:10
    sensor_node_id: urn:wisebed:node:ubern:10
    gateway: true
    nodeType: TMoteSky
    description: Node 10 (Anwander, Wagenknecht)
    urn:wisebed:node:capability:temperature: 0.00
    urn:wisebed:node:capability:light_1: 0.00
    urn:wisebed:node:capability:light_2: 0.00
    urn:wisebed:node:capability:relative_humidity: 0.00
  urn:wisebed:node:ubern:11
  urn:wisebed:node:ubern:12
  urn:wisebed:node:ubern:13
  urn:wisebed:node:ubern:14
  urn:wisebed:node:ubern:15
  urn:wisebed:node:ubern:16
  urn:wisebed:node:ubern:17
  urn:wisebed:node:ubern:18
  urn:wisebed:node:ubern:19
  urn:wisebed:node:ubern:20
  urn:wisebed:node:ubern:21
  urn:wisebed:node:ubern:22
  urn:wisebed:node:ubern:23
```

Figure 43: Reservation: testbed map.

8.2 Experiment Configuration

After the user has reserved nodes for certain time slots it has to configure the experiment. Configuration of an experiment includes binary code images and configuration commands for the sensor nodes, number of runs which the experiment should be performed, and additional information such as experiment description.

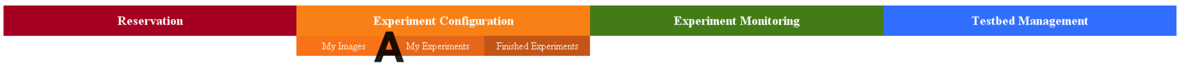
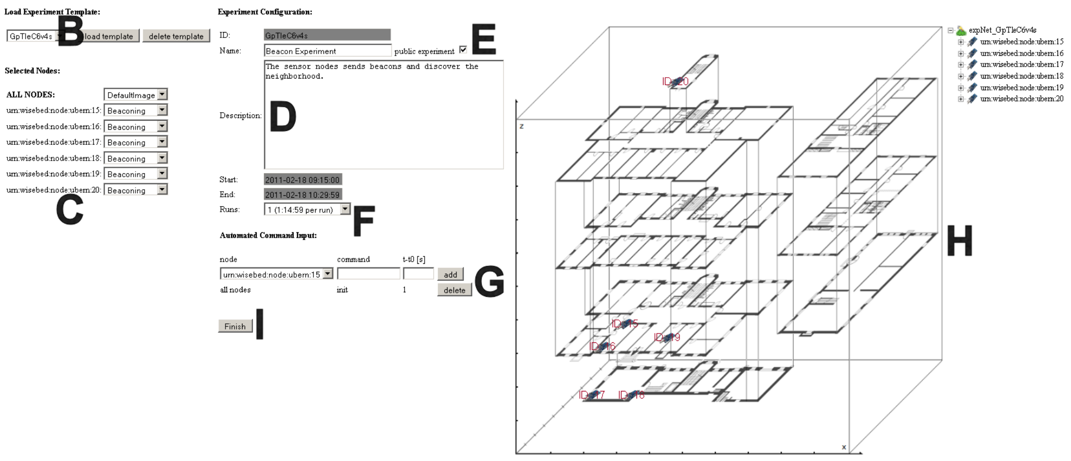


Figure 44: Configuration: menu.

(A) By clicking on the *Configure Experiment for Reservation* button (cf. **(H)** in Fig. 39) after reservation or via the *Experiment Configuration* → *My Experiments* tab (cf. Fig. 44 the user can configure its experiments using the dialog shown in Fig. 45.


Load Experiment Template:

GpTleC6v4s | load template | delete template

Selected Nodes:

ALL NODES:	DefaultImage
urn:wisebed:node:ubern:15:	Beaconing
urn:wisebed:node:ubern:16:	Beaconing
urn:wisebed:node:ubern:17:	Beaconing
urn:wisebed:node:ubern:18:	Beaconing
urn:wisebed:node:ubern:19:	Beaconing
urn:wisebed:node:ubern:20:	Beaconing

Experiment Configuration:

ID: GpTleC6v4s
Name: Beacon Experiment public experiment ☑
Description: The sensor nodes sends beacons and discover the neighborhood.
Start: 2011-02-18 09:15:00
End: 2011-02-18 10:29:59
Runs: 1 (1:14:59 per run)

Automated Command Input:

node	command	t-t0 [s]	
urn:wisebed:node:ubern:15			add
all nodes	init	1	delete

Finish

expNet_GpTleC6v4s
urn:wisebed:node:ubern:15
urn:wisebed:node:ubern:16
urn:wisebed:node:ubern:17
urn:wisebed:node:ubern:18
urn:wisebed:node:ubern:19
urn:wisebed:node:ubern:20

Figure 45: Configuration: image, description, number of runs, automated commands.

(B) First, the user can use a experiment template. This includes the images of the selected nodes, the description, the number of runs and the set of automated commands. Using a template could be utile, if the user wants to perform a series of similar experiments.

(C) The user can choose its binary code which will be uploaded to the selected sensor nodes. It can choose one image for all nodes or different images for different sensor nodes.

(D) A *Name* and a *Description* can be entered optionally by the user.

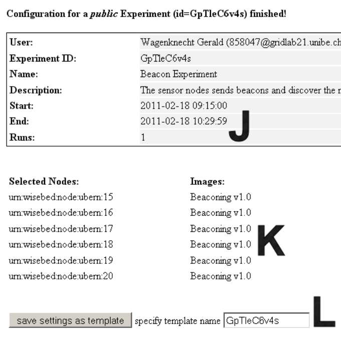
(E) If the user checks the *public experiment* checkbox, the experiment can be monitored by every TARWIS User and TARWIS Viewer.

(F) The user can select the number of runs, how often the experiment should be repeated. The time of a run is divided through reserved time. After a run the nodes are reseted and the experiment starts again.

(G) To configure the nodes or control the experiment the user may add commands which are transmitted to the sensor nodes at the chosen time. The commands are transmitted to the sensor nodes using the serial interface and have to be interpreted and executed by the operating system on the sensor node. A command can be send to all or only to selected sensor nodes.

(H) On the right side, the map with the selected sensor nodes and additional information (such as (ID, type, description, capabilities) are displayed.

(I) Clicking on the *Finish* button finish the configuration sheet.


Configuration for a *public* Experiment (id=GpTleC6v4s) finished!

User:	Wagenknecht Gerald (858047@gridlab21.unibe.ch)
Experiment ID:	GpTleC6v4s
Name:	Beacon Experiment
Description:	The sensor nodes sends beacons and discover the neighborhood.
Start:	2011-02-18 09:15:00
End:	2011-02-18 10:29:59
Runs:	1

Selected Nodes:	**Images:**
urn:wisebed:node:ubern:15	Beaconing v1.0
urn:wisebed:node:ubern:16	Beaconing v1.0
urn:wisebed:node:ubern:17	Beaconing v1.0
urn:wisebed:node:ubern:18	Beaconing v1.0
urn:wisebed:node:ubern:19	Beaconing v1.0
urn:wisebed:node:ubern:20	Beaconing v1.0

save settings as template specify template name GpTleC6v4s

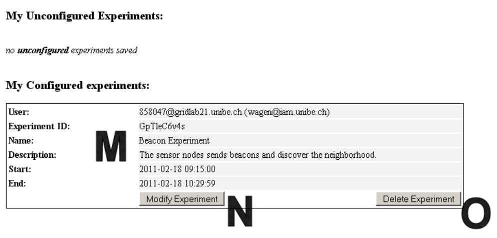
Figure 46: Configuration: configured experiment.

(J) After finishing, the configuration data of the experiment are depicted (cf. Fig. 46), like experiment ID, name and description of the experiment, start and end time and number of runs, and

(K) the selected sensor nodes with the selected images.

(L) Furthermore, the user can save the current configuration into a template for re-using with further experiments.

By clicking on *Experiment Configuration* → *My Experiments* (cf. Fig. 44 the user can finds its experiments, the unconfigured ones and the configured ones (cf. Fig. 47).


My Unconfigured Experiments:

*no **unconfigured** experiments saved*

My Configured experiments:

User:	858047@gridlab21.unibe.ch (wagen@iam.unibe.ch)
Experiment ID:	GpTleC6v4s
Name:	Beacon Experiment
Description:	The sensor nodes sends beacons and discover the neighborhood.
Start:	2011-02-18 09:15:00
End:	2011-02-18 10:29:59

Figure 47: Configuration: my experiments.

(M) The user can find again the configuration data of all its experiments, like experiment ID, name and description of the experiments, and start and end time as well.

(N) To modify the experiment configuration, the user can press the *Modify Experiment* button.

(O) To delete the experiment configuration, the user can press the *Delete Experiment* button.

8.3 Experiment Monitoring

A configured experiment is performed during the reserved time slots. The user can monitor its own experiments (or public experiments). It can follow the output of the sensor nodes. If necessary, the user can send commands to the sensor nodes or reset the sensor nodes (if it is owner of the experiment). It is possible that two or more experiments running in parallel on the testbed and the user can switch between them.

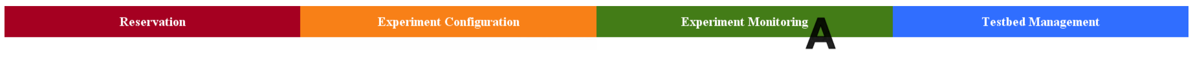


Figure 48: Monitoring: menu.

(A) By clicking on the *Experiment Monitoring* tab the user can monitor the experiments (cf. Fig. 48).

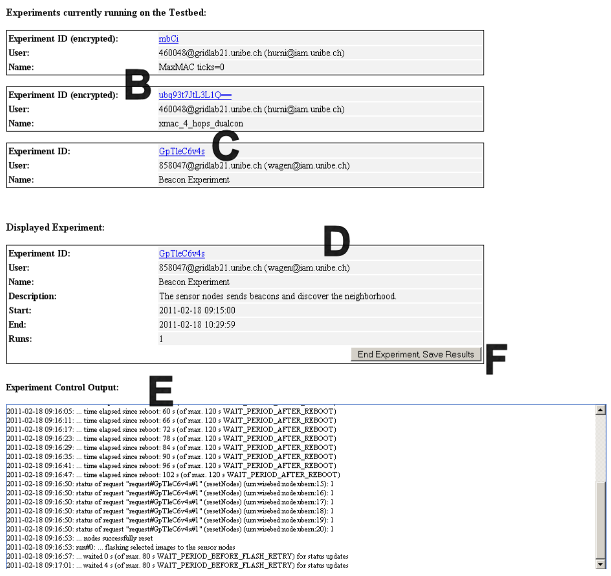
(B) On top of the site (cf. Fig. 49), all running experiments are listed including the *Experiment ID*, the owning *User*, and the *Name* of the experiment.

(C) By clicking on the experiment ID, the user choose the experiment it wants to monitor.

(D) This experiment is listed with additional information such as experiment description, start and end time, and number of runs.

(E) Also displayed is the control output, which includes, e.g., status of flashing the images on the sensor nodes.

(F) By clicking on the *End Experiment, Save Result* button, the user cancel the experiment before the regular end. The user will get an email with the zipped experiment results.



Experiments currently running on the Testbed:

Experiment ID (encrypted):	mbCi
User:	460048@gridlab21.unibe.ch (hurni@iam.unibe.ch)
Name:	MaxMAC ticks=0

Experiment ID (encrypted):	ubq93t7JtL3L1Q==
User:	460048@gridlab21.unibe.ch (hurni@iam.unibe.ch)
Name:	xmac_4_hops_dualcon

Experiment ID:	GpTleC6v4s
User:	858047@gridlab21.unibe.ch (wagen@iam.unibe.ch)
Name:	Beacon Experiment

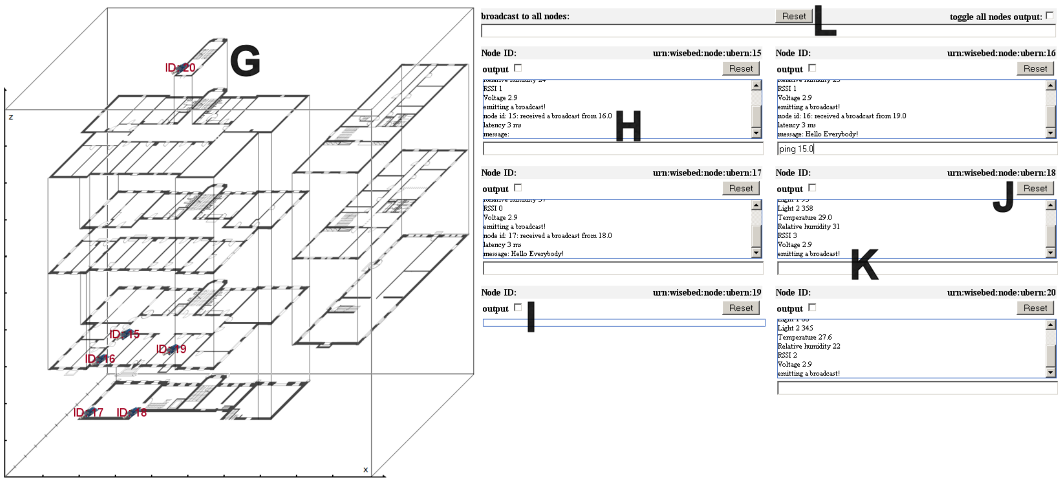
Displayed Experiment:

Experiment ID:	GpTleC6v4s
User:	858047@gridlab21.unibe.ch (wagen@iam.unibe.ch)
Name:	Beacon Experiment
Description:	The sensor nodes sends beacons and discover the neighborhood.
Start:	2011-02-18 09:15:00
End:	2011-02-18 10:29:59
Runs:	1

Experiment Control Output:

```
2011-02-18 09:16:05: ... time elapsed since reboot: 60 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:11: ... time elapsed since reboot: 66 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:17: ... time elapsed since reboot: 72 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:23: ... time elapsed since reboot: 78 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:29: ... time elapsed since reboot: 84 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:35: ... time elapsed since reboot: 90 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:41: ... time elapsed since reboot: 96 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:47: ... time elapsed since reboot: 102 s (of max. 120 s WAIT_PERIOD_AFTER_REBOOT)
2011-02-18 09:16:50: status of request "request#GpTleC6v4s#1" (resetNodes) (urn:wisebed:node:ubern:15): 1
2011-02-18 09:16:50: status of request "request#GpTleC6v4s#1" (resetNodes) (urn:wisebed:node:ubern:16): 1
2011-02-18 09:16:50: status of request "request#GpTleC6v4s#1" (resetNodes) (urn:wisebed:node:ubern:17): 1
2011-02-18 09:16:50: status of request "request#GpTleC6v4s#1" (resetNodes) (urn:wisebed:node:ubern:18): 1
2011-02-18 09:16:50: status of request "request#GpTleC6v4s#1" (resetNodes) (urn:wisebed:node:ubern:19): 1
2011-02-18 09:16:50: status of request "request#GpTleC6v4s#1" (resetNodes) (urn:wisebed:node:ubern:20): 1
2011-02-18 09:16:53: ... nodes successfully reset
2011-02-18 09:16:53: run#0: ... flashing selected images to the sensor nodes
2011-02-18 09:16:57: ... waited 0 s (of max. 80 s WAIT_PERIOD_BEFORE_FLASH_RETRY) for status updates
2011-02-18 09:17:01: ... waited 4 s (of max. 80 s WAIT_PERIOD_BEFORE_FLASH_RETRY) for status updates
```

Figure 49: Monitoring: switching between parallel experiments.



broadcast to all nodes: **toggle all nodes output:**

Node ID: **urn:wisebed:node:ubern:15**

```
RSSI 1
Voltage 2.9
emitting a broadcast!
node id: 15: received a broadcast from 16.0
latency 3 ms
message:
```

Node ID: **urn:wisebed:node:ubern:16**

```
RSSI 1
Voltage 2.9
emitting a broadcast!
node id: 16: received a broadcast from 19.0
latency 3 ms
message: Hello Everybody!
```

ping 15.0

Node ID: **urn:wisebed:node:ubern:17**

```
RSSI 0
Voltage 2.9
emitting a broadcast!
node id: 17: received a broadcast from 18.0
latency 3 ms
message: Hello Everybody!
```

Node ID: **urn:wisebed:node:ubern:18**

```
Light 2 358
Temperature 29.0
Relative humidity 31
RSSI 3
Voltage 2.9
emitting a broadcast!
```

Node ID: **urn:wisebed:node:ubern:19**

Node ID: **urn:wisebed:node:ubern:20**

```
Light 2 345
Temperature 27.6
Relative humidity 22
RSSI 2
Voltage 2.9
emitting a broadcast!
```

Figure 50: Monitoring: experiment output and map.

On the bottom of the site the output of the sensor nodes of the chosen experiment is displayed as shown in Fig. 50.

- **(G)** On the left side the nodes' connectivity on the node map is displayed, as soon as nodes transmits packets and discover each other.
- **(H)** On the right side an output window including a *Reset* button and a command line is displayed for each sensor node used in the experiment.
- **(I)** The output window can be switched off for performance and clearness reasons by clicking in the *output* checkbox.
- **(J)** If the user notices that a node misbehaves (e.g., is stuck in an endless loop or similar), it can reset the node using the *Reset* button.
- **(K)** Furthermore the user can send commands to the sensor nodes using the command line. The set of commands which can be used is the same as for the control of a local physical testbed and depends on the operating system on the sensor nodes.
- **(L)** It is also possible to reset all nodes with one click and send a command parallel to all sensor nodes.

8.4 Finishing Experiments

After finishing (or canceling) an experiment, all results are stored in the designated TARWIS database. The experiments' results and further information about the experiment are stored using WiseML (Wireless Sensor Network Markup Language) as described in the next Section.

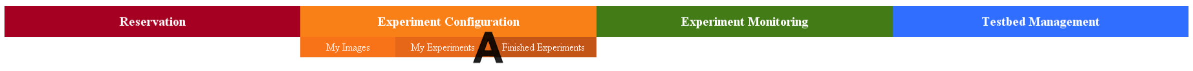
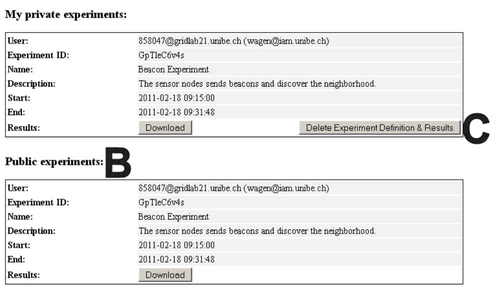


Figure 51: Configuration: menu.

(A) The finished experiments can be found in the *Experiment Configuration* → *Finished Experiments* tab (cf. Fig. 51).



My private experiments:

User:	858047@gridlab21.unibe.ch (wagen@iam.unibe.ch)
Experiment ID:	GpTleC6v4s
Name:	Beacon Experiment
Description:	The sensor nodes sends beacons and discover the neighborhood.
Start:	2011-02-18 09:15:00
End:	2011-02-18 09:31:48
Results:	Download Delete Experiment Definition & Results

Public experiments:

User:	858047@gridlab21.unibe.ch (wagen@iam.unibe.ch)
Experiment ID:	GpTleC6v4s
Name:	Beacon Experiment
Description:	The sensor nodes sends beacons and discover the neighborhood.
Start:	2011-02-18 09:15:00
End:	2011-02-18 09:31:48
Results:	Download

Figure 52: Configuration: finished experiments.

(B) On this site (cf. Fig. 52) all users' experiments are listed as well as all public experiments. The user can download the experiment results of its own experiments and of the public experiments

(C) For its own experiment, the user can delete the definitions and the results.

9 Data Acquisition and Representation

TARWIS integrates the WiseML (Wireless Sensor Network Markup Language) for several purposes. On one side, it uses WiseML for reading and parsing the necessary information about it's underlying *Network definition*. Furthermore, it uses WiseML for storing and generating the output of the *Experiment log and debug traces* in a common defined format.

Network definition: in order to read the network resources (node type, sensors, positions, etc), TARWIS calls the getNetwork() function of the SessionManagementService API, and retrieves a WiseML document listing the entire network endowment. It uses the retrieved positions to display the nodes of the network in the network graph. Listing 1 lists one *instantiation* of a node entry. The node type and endowment are described in the *defaults* section.

Listing 1: Node entry in SessionManagementService of Univ. of Bern testbed

```
<node id="urn:wisebed:node:ubern:1">
<position>
<x>69</x>
<y>20</y>
<z>52</z>
 </position>
<gateway>true </gateway>
<description>Node 1 – Office 205 (2nd Floor)</description>
 </node>
```

Experiment log and debug traces: As soon as an experiment is scheduled and configured, the TARWIS *ControllerService* retrieves experiment output (e.g. debug information, sensor values) over the *receive* function and stores it to the TARWIS internal database.

As soon as the experiment time has expired, the nodes are reflashed with a default image, and the network is prepared for the subsequent experiment. Every output of the finishing experiment is exported by TARWIS to a WiseML-file, zipped and saved to the TARWIS database. This WiseML-file hence comprises all important information about an experiment run, e.g., where the experiment took place geographically, what kind of nodes were used, what their sensor endowment was, and much more. Storing all this experiment-related information in one WiseML file offers many advantages, besides the possibility to easily use it for post-experiment analysis. As it defines essentially all crucial information of an experiment, it further allows to make the experiment data public to other research partners in a common well-defined language, giving them the opportunity to repeat the same or similar experiment, e.g. trying to improve the results. Hence, having integrated WiseML into the Testbed Management System inherently pushes research on wireless sensor networks one crucial step towards transparency and repeatability of sensor network experimentation.

Listing 2: Excerpt from a TARWIS-generated Experiment Trace

```
<wiseml> [...]
<trace id="experiment_UBERN_uniqueID_23453323">
  [...]
  <timestamp>3605.164612</timestamp>
  <node id="urn:wisebed:node:ubern:9">
  <position>
      <x>85</x>
      <y>80</y>
      <z>52</z>
   </position>
  <data key="textOutput">latency 15 ms</data>
   </node>
  <timestamp>3605.164612</timestamp>
  <node id="urn:wisebed:node:ubern:9">
  <position>
      <x>85</x>
      <y>80</y>
      <z>52</z>
   </position>
  <data key="textOutput">Light 1 202</data>
   </node>
  [...]
</trace>
[...]
</wiseml>
```

The WiseML code sample in Listing 2 lists two trace events retrieved in a small experiment at the University of Bern testbed. For each output line, one can determine the exact time (within the precision of some few milliseconds) relative to the experiment start time (c.f. the *timestamp* tag), the position of the node (hence, with mobile nodes, the node movement can also be captured) and the output itself. The WiseML-file generated by TARWIS can therefore describe to a very high degree what has happened at a certain time during the experiment.

10 TARWIS Testbed Management

To perform testbed administration tasks the user has to be a TARWIS Admin. How to become a TARWIS Admin is described in Section 5.

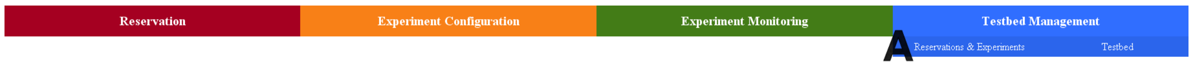


Figure 53: Testbed management: menu.

(A) By clicking on the *Testbed Management → Reservations and Experiments* tab, the administrator can modify the reservations and experiments (cf. Fig. 54).

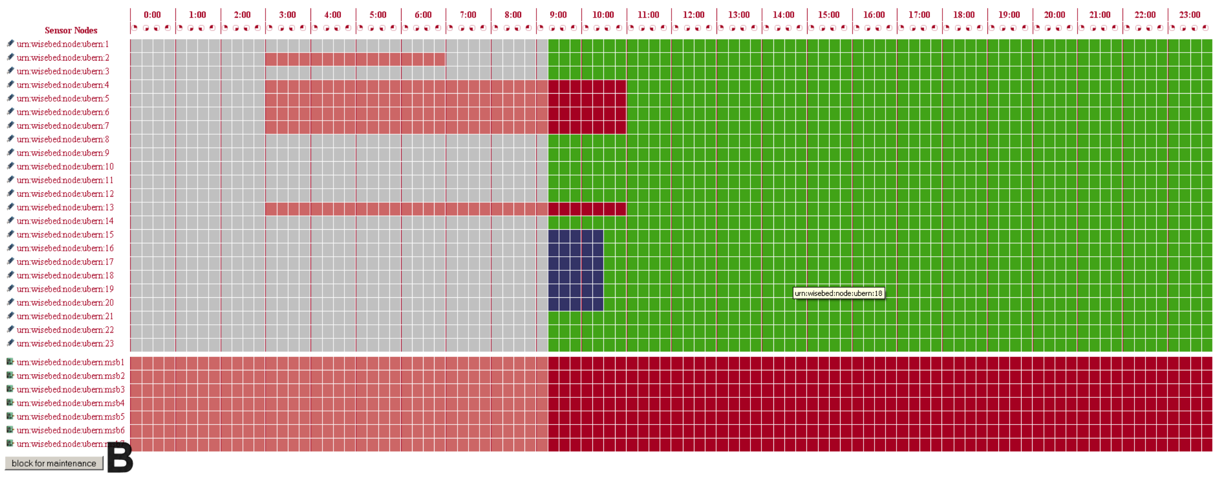

Sensor Nodes
0:00
1:00
2:00
3:00
4:00
5:00
6:00
7:00
8:00
9:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
urn:wisebed:node:ubern:1
urn:wisebed:node:ubern:2
urn:wisebed:node:ubern:3
urn:wisebed:node:ubern:4
urn:wisebed:node:ubern:5
urn:wisebed:node:ubern:6
urn:wisebed:node:ubern:7
urn:wisebed:node:ubern:8
urn:wisebed:node:ubern:9
urn:wisebed:node:ubern:10
urn:wisebed:node:ubern:11
urn:wisebed:node:ubern:12
urn:wisebed:node:ubern:13
urn:wisebed:node:ubern:14
urn:wisebed:node:ubern:15
urn:wisebed:node:ubern:16
urn:wisebed:node:ubern:17
urn:wisebed:node:ubern:18
urn:wisebed:node:ubern:19
urn:wisebed:node:ubern:20
urn:wisebed:node:ubern:21
urn:wisebed:node:ubern:22
urn:wisebed:node:ubern:23
urn:wisebed:node:ubern:18
urn:wisebed:node:ubern:msb1
urn:wisebed:node:ubern:msb2
urn:wisebed:node:ubern:msb3
urn:wisebed:node:ubern:msb4
urn:wisebed:node:ubern:msb5
urn:wisebed:node:ubern:msb6
urn:wisebed:node:ubern:
block for maintenance
B

Figure 54: Testbed management: block for maintenance.

(B) The TARWIS Admin has the same reservation schedule sheet as the TARWIS user, but the administrator can block some nodes (or the whole testbed) for a certain time period due to maintenance reasons.

Figure 55: Testbed management: undo blocking.

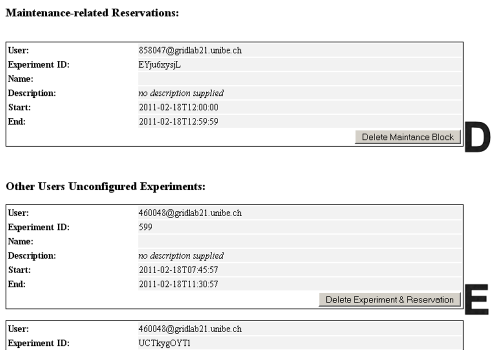
(C) After blocking the administrator can also undo it.

(D) The administrator has a list of all maintenance blocks and can delete them.

(E) And, it has a list of all users' reservations and experiment configurations and can delete them as well.

(A) By clicking on the *Testbed Management → Reservations and Experiments* tab, the user can add new sensor nodes to the testbed (cf. Fig. 58), or update existing ones (cf. Fig. 59).

(B) By selecting *Add New Sensor Node(...)* from the drop-down menu, the user can enter the properties of the new sensor node into the form.



Maintenance-related Reservations:

User:	858047@gridlab21.unibe.ch
Experiment ID:	EYju6xysjL
Name:	
Description:	*no description supplied*
Start:	2011-02-18T12:00:00
End:	2011-02-18T12:59:59

Delete Maintance Block

Other Users Unconfigured Experiments:

User:	460048@gridlab21.unibe.ch
Experiment ID:	599
Name:	
Description:	*no description supplied*
Start:	2011-02-18T07:45:57
End:	2011-02-18T11:30:57

Delete Experiment & Reservation

User:	460048@gridlab21.unibe.ch
Experiment ID:	UCTkygOYT1

Figure 56: Testbed management: list of maintenance blocks and users' reservations.

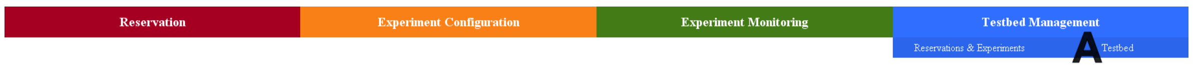
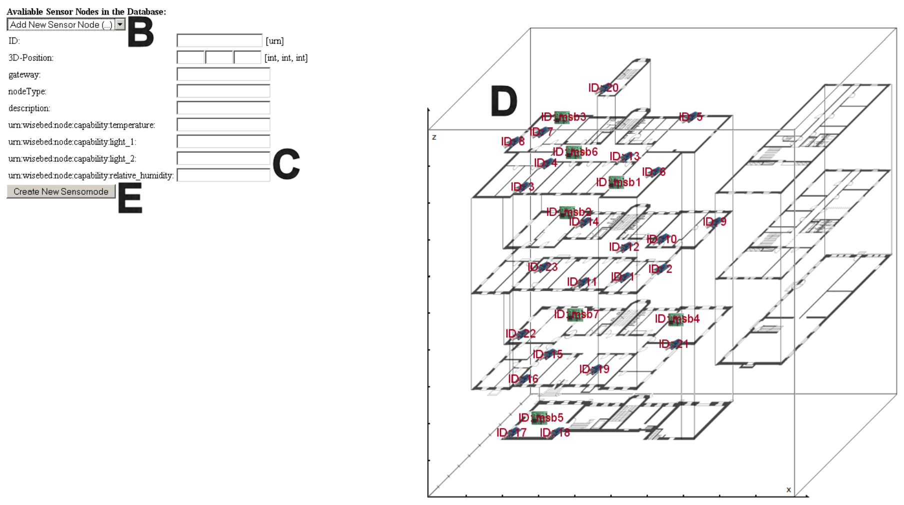


Figure 57: Testbed management: menu.



Avaliable Sensor Nodes in the Database:

Add New Sensor Node (...)

ID:	[urn]
3D-Position:	[int, int, int]
gateway:	
nodeType:	
description:	
urn:wisebed:node:capability:temperature:	
urn:wisebed:node:capability:light_1:	
urn:wisebed:node:capability:light_2:	
urn:wisebed:node:capability:relative_humidity:	

Create New Sensornode

Figure 58: Testbed management: add new sensor node.

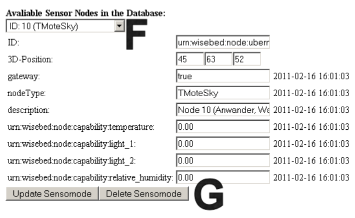
(C) This includes the ID (urn), position, gateway, type, description, and capabilities.

(D) The x-, y-, and z-value of the sensor nodes' position are between 0 and 100 according to the coordinates in the map.

(E) By clicking the *Create New Sensor Node* button, the node is added to the TARWIS database and depicted in the map.

(F) By selecting an existing sensor node from the drop-down menu, the user can update the properties of the new sensor node.

(G) The user can either update the properties for the selected sensor node or delete the sensor node from the database.



Avaliable Sensor Nodes in the Database:

ID: 10 (TMoteSky)

ID:	urn:wisebed:node:uberr	
3D-Position:	45 63 52	
gateway:	true	2011-02-16 16:01:03
nodeType:	TMoteSky	2011-02-16 16:01:03
description:	Node 10 (Anwander, Wa	2011-02-16 16:01:03
urn:wisebed:node:capability:temperature:	0.00	2011-02-16 16:01:03
urn:wisebed:node:capability:light_1:	0.00	2011-02-16 16:01:03
urn:wisebed:node:capability:light_2:	0.00	2011-02-16 16:01:03
urn:wisebed:node:capability:relative_humidity:	0.00	2011-02-16 16:01:03

Figure 59: Testbed management: update sensor node.

References

[1] B. Blywis, F. Juraschek, M. Günes, and J. Schiller. Design concepts of a persistent wireless sensor testbed. In *7. GI/ITG KuVS Fachgespräch Sensornetze*, 2008.

[2] I. Haratcherev, G. Halkes, T. Parker, O. Visser, and K. Langendoen. PowerBench: A scalable testbed infrastructure for benchmarking power consumption. In *Int. Workshop on Sensor Network Engineering (IWSNE)*, pages 37–44, Santorini Island, Greece, June 2008.

[3] Ertin, E. et. al. Kansei: a testbed for sensing at scale. In *Intl. Conference On Information Processing In Sensor Networks (IPSN)*, 2006.

[4] Seventh Framework Programme FP7 - Information and Communication Technologies. Wireless sensor networks testbed project (wisebed), ongoing project since june 2008. http://www.wisebed.eu.

[5] P. Hurni and G.Wagenknecht and M. Anwander and T. Braun. A Testbed Management System for Wireless Sensor Network Testbeds (TARWIS). European Conference on Wireless Sensor Networks (EWSN), February 17-19, Coimbra, Portugal, 2010.

[6] G. Werner-Allen, P. Swieskowski, and M. Welsh. Motelab: A wireless sensor network testbed. In *IPSN-SPOTS '05*, pages 483–488, California, USA, April 2005.