



deegree User Rights, Roles and Resources (U3R) v2.5

lat/lon GmbH

Aennchenstr. 19
53177 Bonn
Germany
Tel ++49 - 228 - 184 96-0
Fax ++49 - 228 - 184 96-29
info@lat-lon.de
www.lat-lon.de

Geographisches Institut
Universität Bonn
Meckenheimer Allee 166
53115 Bonn

Tel. ++49 228 732098

Change log

Datum	Beschreibung	Author
2006-10-17	Verallgemeinerung des Dokumentes aus projektspezifischer Dokumentation	Markus Müller
2007-01-08	Abbildungen aktualisiert; Definition von Transaktionsrechten beschrieben	Andreas Poth
2007-01-16	Zusammenfassung der drei Dokumentationen zu U3R in ein Dokument; Harmonisierung mit Standard-deegree Dokumentationsstruktur	Markus Müller
2007-03-02	Präzisierung unklarer Stellen zur Verwaltung von FeatureTypes, dem Ändern des SEC_ADMIN Passwortes, zur Benutzung des Kommandozeilentools und einer Reihe weiterer Details.	Markus Müller, Andreas Poth
2007-04-23	Bug Fixes	Andreas Poth
2007-06-01	Erweiterung um context chooser für iGeoPortal	Judit Mays
2007-06-22	Ergänzungen für context chooser	Judit Mays
2007-06-22	Kleine Verbesserungen	Markus Müller
2008-04-02	Translation into English	Markus Lupp
2008-04-11	Proof-reading and corrections	Andy Turner
2009-07-27	Version changed (v2.2)	Andreas Poth
	Update to version 2.5 (to be done)	

Table of Contents

1 Introduction.....	5
2 Download / Installation.....	7
2.1 Prerequisites.....	7
2.2 deegree U3R Release.....	7
2.3 Implementation of the database schema.....	7
2.4 Installation of the Web-Frontend.....	8
2.5 Changing the password of SEC_ADMIN.....	11
3 U3R fundamentals.....	12
3.1 The concept of U3R.....	12
3.2 Example: accumulation of access rights.....	12
3.3 Administrators.....	14
4 Web frontend.....	15
4.1 Layers/FeatureTypes.....	16
4.2 User Editor.....	17
4.2.1 Extension of the User Editor: assign WebMapContext.....	19
4.3 Group-Editor.....	23
4.4 Role Editor.....	25
4.4.1 Create a role.....	25
4.4.2 Delete a role.....	25
4.4.3 Editing role-group associations.....	26
4.4.4 Edit a role.....	26
4.5 Rights Editor.....	26
5 Configuration using the command line.....	28
5.1 Program call.....	28
5.2 Common program parameters.....	28
5.3 known actions/operations.....	28
Appendix A: U3R database schema.....	35

Index of Tables

Table 1: Definition of roles and datasets.....	13
Table 2: Definition of groups and roles.....	13

Illustration Index

Figure 1: Initial login page.....	15
Figure 2: Central navigation page.....	16
Figure 3: Form for input of new Layers and FeatureTypes as well as for deletion of existing Layers and FeatureTypes.....	17
Figure 4: Formular for administration of users.....	18
Figure 5: User Editor with Context Chooser.....	20
Figure 6: Form for administration of groups.....	24
Figure 7: Form for administration of role-group associations.....	25
Figure 8: Rights Editor.....	27
Figure 9 database schema used by U3R.....	35

1 Introduction

deegree is a Java Framework offering the main building blocks for Spatial Data Infrastructures (SDIs). Its entire architecture is developed using standards of the Open Geospatial Consortium (OGC) and ISO Technical Committee 211 – Geographic information / Geoinformatics (ISO/TC 211). deegree is comprised of OGC Web Services implementations as well as clients for these services. It is Open Source Software covered by the GNU Lesser General Public License (GNU LGPL) and is available via the following URL:

<http://www.deegree.org>

deegree2 is the new release of deegree supporting a number of features that deegree1 was not able to handle. This documentation concerns the setup and configuration of deegree User Rights and Resources (U3R), the user and rights administration component of deegree2. U3R is part of deegree iGeoSecurity and can be used in conjunction with deegree owsProxy and deegree WASS.

This is user documentation for administrators of deegree2 U3R, i.e. those persons who register users and data with U3R and assign corresponding access rights. This documentation also describes the functionality and features of the system, so may be of interest to others.

U3R provides flexibility to define users and their rights and helps control what users can do in the system. It allows for complex tree or network structures using corresponding inheritance of rule-based access rights to be represented. It is highly recommended that some planning is done to ascertain what roles and rights and groups are wanted for users of the system as a preparatory step. Such planning should result in easier administration and management of the system.

Besides U3R, deegree comprises a number of additional services and clients. A complete list of deegree components can be found using the following URL and links:

<http://www.lat-lon.de> → Products

Downloads of packaged deegree components can be found via:

<http://www.deegree.org> → Download

2 Download / Installation

2.1 Prerequisites

To run deegree2 U3R you need:

- Java (JRE or JDK) version 1.5.x
- Tomcat 5.5.x

For installation of these components refer to the corresponding documentation at java.sun.com and tomcat.apache.org respectively.

2.2 deegree U3R Release

Until an installation package (Demo Release) for U3R is available, all the corresponding files have to be obtained from the deegree SVN.

To install U3R, the following components are needed:

- database schema for persistent storage of users and rights
- Java Server Pages (JSPs), realising the graphical administration tool for U3R
- Java classes, including an Application Programming Interface (API) and helper classes for rights administration. They are part of deegree and included in deegree2.jar.

2.3 Implementation of the database schema

To put U3R to use, a database schema implementation is required. Currently the only supported databases for this are PostgreSQL and Oracle. The installation files for deegree U3R include two SQL scripts (one for each of the above mentioned databases) which can be used to create and initialize the necessary tables.

An administrator account is created in the initialization script using the name 'SEC_ADMIN' and the password 'JOSE67' by default. Administration is needed to create new users, assign roles and groups, and diagnose system messages. The administration is done by administrators. While the name for the administrator account should not be changed from SEC_ADMIN, the password should be changed and a valid email address for the administrators should be used. There are two ways to achieve this: The modifications can be done using database mechanisms once the initialization script has been run; or, the following line of the initialization script can be modified:

```
INSERT INTO SEC_USERS ("ID", "PASSWORD", "FIRSTNAME", "LASTNAME",  
"EMAIL") VALUES (1, 'JOSE67', 'SEC_ADMIN', 'SEC_ADMIN',  
'admin@myhost.de');
```

JOSE67 should be replaced with the administrators password and admin@myhost.de should be replaced with the administrators email address.

The administrators email address is sent system messages containing information that might be useful for administration purposes.

When the script is executed using a suitable database client or from the command line, the following table should have been created in the database:

SEC_JT_GROUPS_GROUPS

SEC_JT_GROUPS_ROLES

SEC_JT_ROLES_PRIVILEGES

SEC_JT_ROLES_SECOBJECTS

SEC_JT_USERS_GROUPS

SEC_JT_USERS_ROLES

SEC_PRIVILEGES

SEC_RIGHTS

SEC_ROLES

SEC_USERS

SEC_GROUPS

SEC_SECURED_OBJECT_TYPES

SEC_SECURED_OBJECTS

SEC_SECURABLE_OBJECTS

(see Fig. 9)

2.4 Installation of the Web-Frontend

After the database is successfully set up, the graphical user interface for administration can be installed. It is implemented as a number of JSP pages that are called by a central Java servlet. Therefore, a Web Context of a servlet container (e.g. Apache Tomcat) is needed.

[Extract from `$TOMCAT_HOME$/conf/server.xml`]

```
...
<Context path="/drm-admin" docBase="D:/java/webapps/drm-admin" debug="0"
  reloadable="true">
  <Logger className="org.apache.catalina.logger.FileLogger" directory="logs"
    prefix="log_drm-admin." suffix=".txt" timestamp="true"/>
</Context>
...
```


All classes specific to deegree are included in the archive deegree2.jar. Additionally the following libraries are needed¹:

- acme.jar
- bzw.jar
- commons-codec-1.3.jar
- commons-httpclient-2.0.2-deegreeversion.jar
- commons-discovery-0.2.jar
- commons-logging.jar
- j3dcore.jar
- j3dutils.jar
- jai_codec.jar
- jai_core.jar
- jaxen-1.1-beta-8.jar
- jts-1.8.jar
- log4j-1.2.9.jar
- mail.jar
- mlibwrapper_jai.jar
- ojdbc14_10g.jar
- postgresql-8.0-311.jdbc3.jar
- vecmath.jar
- xerces_2_5_0.jar
- xml-apis.jar

For registration with the servlet engine, a number of initialization parameters have to be defined in the deployment descriptor (web.xml). The following is an example; the deployment descriptor is given, then the different initialization parameters are explained.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
    "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">
<web-app>
  <servlet>
    <servlet-name>SecurityRequestDispatcher</servlet-name>
    <servlet-class>
      org.deegree.portal.standard.security.control.SecurityRequestDispatcher
    </servlet-class>
```

¹If using Apache Tomcat, it is sufficient to copy the jar-archives in the lib directory of the corresponding web context.

```

<init-param>
  <param-name>Handler.configFile</param-name>
  <param-value>
    WEB-INF/conf/security_controller.xml
  </param-value>
</init-param>
<init-param>
  <param-name>Security.configFile</param-name>
  <param-value>
    WEB-INF/conf/security.xml
  </param-value>
</init-param>
</servlet>
<servlet-mapping>
  <servlet-name>SecurityRequestDispatcher</servlet-name>
  <url-pattern>/SecurityRequestDispatcher</url-pattern>
</servlet-mapping>
</web-app>

```

The parameter `Handler.configFile` is used to define the location of the configuration file for distribution of requests to internal handler classes and JSP pages. Modifications of this file may result in system malfunction.

The second parameter, `Security.configFile`, defines a reference to an XML file that includes the access parameters for the database (of the schema implementation).

```

<security>
  <registryClass>org.deegree.security.drm.SQLRegistry</registryClass>
  <readWriteTimeout>300</readWriteTimeout>
  <registryConfig>
    <jdbc:JDBCConnection xmlns:jdbc="http://www.deegree.org/jdbc">
      <jdbc:Driver>oracle.jdbc.OracleDriver</jdbc:Driver>
      <jdbc:Url>jdbc:oracle:thin:@localhost:1521:latlon</jdbc:Url>
      <jdbc:User>security</jdbc:User>
      <jdbc:Password>security</jdbc:Password>
      <jdbc:SecurityConstraints/>
      <jdbc:Encoding>iso-8859-1</jdbc:Encoding>
    </jdbc:JDBCConnection>
  </registryConfig>
</security>

```

- driver: Java driver class for JDBC access (dependent on the used database management system; in the example an ODBC-database is configured).
- url: address and name of the database can be accessed using the JDBC driver
- user: user name (optional)
- password: password (optional)
- SecurityConstraints and Encoding have to be supplied but are currently only used as dummies and are not evaluated.

2.5 Changing the password of SEC_ADMIN

The default initialization of U3R sets a default password for the SEC_ADMIN administrators account. This should be changed as detailed in Section 2.3.

To use the command line tool DRMAccess (see chapter 5) `org/deegree/tools/security/sec.properties` has to be adjusted accordingly.

3 U3R fundamentals

3.1 The concept of U3R

deegree's rights management allows control over which geospatial objects a user is allowed to access/modify. For implementing this the following concepts are used:

User:

- The person accessing a resource.
- For identification user name and password are used².
- Editing of user data is only possible using the administration GUI.

Groups:

- Are used to collect users.
- Groups can be combined to super groups, i.e. they can be members of a super group.
- Arbitrary networks of relationships between groups can be defined.

Roles:

- Are collections of rights relative to the datasets (i.e. a role either has the access right for a dataset or not).
- Access rights can be parametrized; it is therefore possible to define an access right for a layer that is only valid on a specific spatial extent.
- The assigned access rights can be edited at all times.
- Roles are associated with an arbitrary number groups. Additionally a group also can be assigned several roles. The members of this group then have accumulated rights of all roles that are associated with the group.

3.2 Example: accumulation of access rights

What follows is an example to illustrate role/rights definition:

²Alternative mechanisms like usage of the network account of a user are also implementable

	Dataset DS1	Dataset DS2	Dataset DS3	Explanation
Role R1	x	x	-	R1 can access DS1 and DS2
Role R2	-	x	-	R2 can access DS2
Role R3	x	-	-	R3 can access DS1
Role R4	-	-	x	R4 can access DS3

Table 1: Definition of roles and datasets

	Role R1	Role R2	Role R3	Role R4	Explanation
Group G1 (Abandoned_hazardous_sites)	x	x	-	-	G1 is associated with R1 and R2
Group G2 (Chemicals)	-	x	x	-	G2 is associated with R2 and R3
Group G3 (Test)	-	-	-	x	G3 is associated with R4

Table 2: Definition of groups and roles

User **Smith** is a member of the groups **Abandoned_hazardous_sites** and **Chemicals**. How does the system determine which datasets he is allowed to access?

1. It determines which groups **Smith** belongs to, in this example **Abandoned_hazardous_sites** and **Chemicals**.
2. The group **Abandoned_hazardous_sites** is associated with the roles **R1** and **R2**, the group **Chemicals** with **R2** and **R3**.
3. Consequently this user is associated with the roles **R1**, **R2** and **R3** (the doubled association with R2 is not relevant).
4. The role **R1** includes access rights for **DS1** and **DS2**. The role **R2** includes access rights for **DS2**, The role **R3** includes access rights for **DS1**.
5. This user therefore can access the datasets **DS1** and **DS2** (multiple access rights on the same dataset are again not relevant).

3.3 Administrators

With regard to the administration of access rights, additional concepts are used besides users, groups and roles:

- All users connected to the role 'SEC_ADMIN' are administrators. 'SEC_ADMIN' is a particular role that is used internally by the system and cannot be deleted.
- Only administrators may access the administration pages, they are allowed to:
 - create, delete and edit users
 - create and delete groups
 - edit membership of groups
 - create, edit and delete roles
 - edit associations between roles and groups
 - associate rights with roles (and define constraints on rights)

U3R can either be administered using a graphical web-interface or a command line tool. The following sections outline both options.

4 Web frontend

The administration user interface is implemented using dynamic web pages (JSP) and JavaScript. This provides an intuitive interface to the user, but not all applied changes are instantly stored on the server-side, the changes have to be communicated to the server before they take effect. Therefore for all screens the following has to be kept in mind:

- **"übernehmen"** ("apply") send the applied changes to the server and stores them. Successful application of the function is confirmed in the browser.
- **"abbrechen"** ("cancel") cancels all applied changes and creates the original state of the system that existed before the corresponding screen was started.

At some places it is possible to select multiple entries from a list and process them in one step (e.g. associate several groups with one role). Multiple selection is invoked by keeping the SHIFT key pressed and simultaneously selecting the desired entries. Instantly after the installation of the application only the system administrator (SEC_ADMIN) can be used as user. This user can then create new users.



Figure 1: Initial login page

After successful login, the central navigation page is displayed (Fig. 2). This can be used to navigate to different pages that allow the user to register and delete resources (*Layers/FeatureTypes*), to create, delete and edit users (*Benutzer*), to create, delete and edit groups (*Gruppen*) and to create, delete and edit roles (*roles*). The current user can also log out (*logout*).

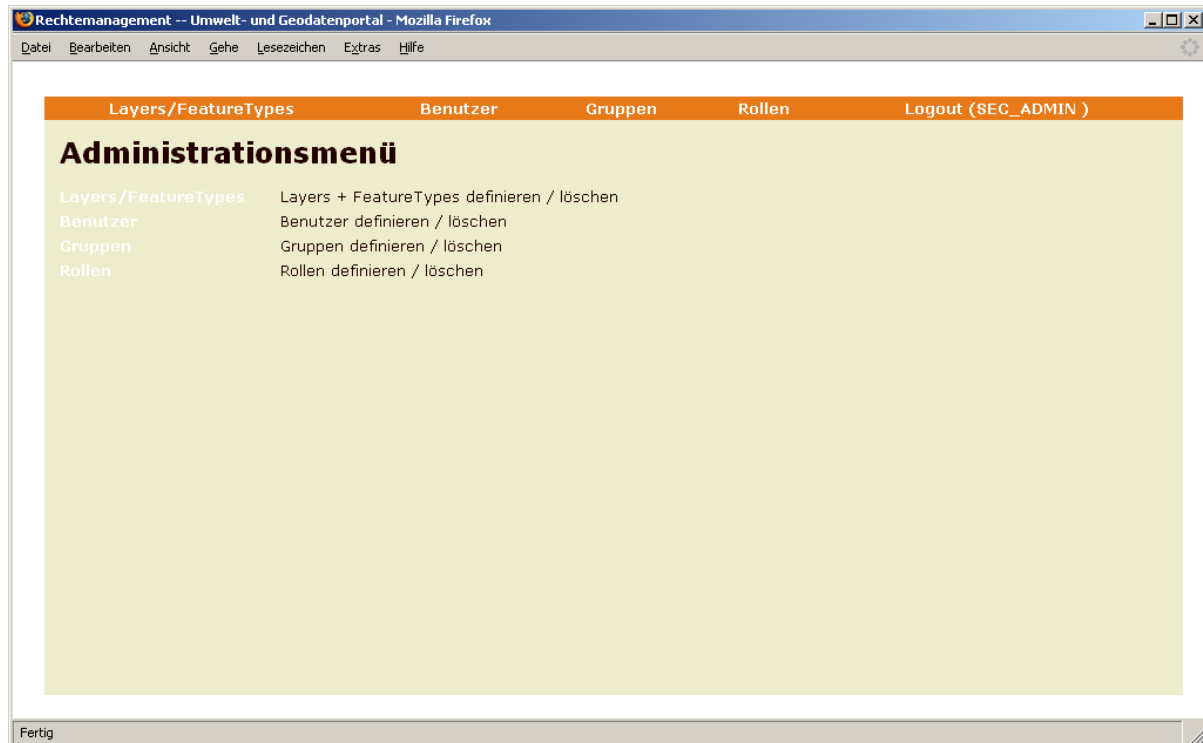


Figure 2: Central navigation page

The menu bar on top is visible on all pages so that these menu items can be accessed at all times.

4.1 Layers/FeatureTypes

Each object that has to be protected using the access rights administration first has to be registered. The type of these objects is unrestricted. To make the input of protected objects as easy as possible, the administration pages are restricted to support (WMS)Layer and (WFS/WFS-G) FeatureTypes (Fig. 3).

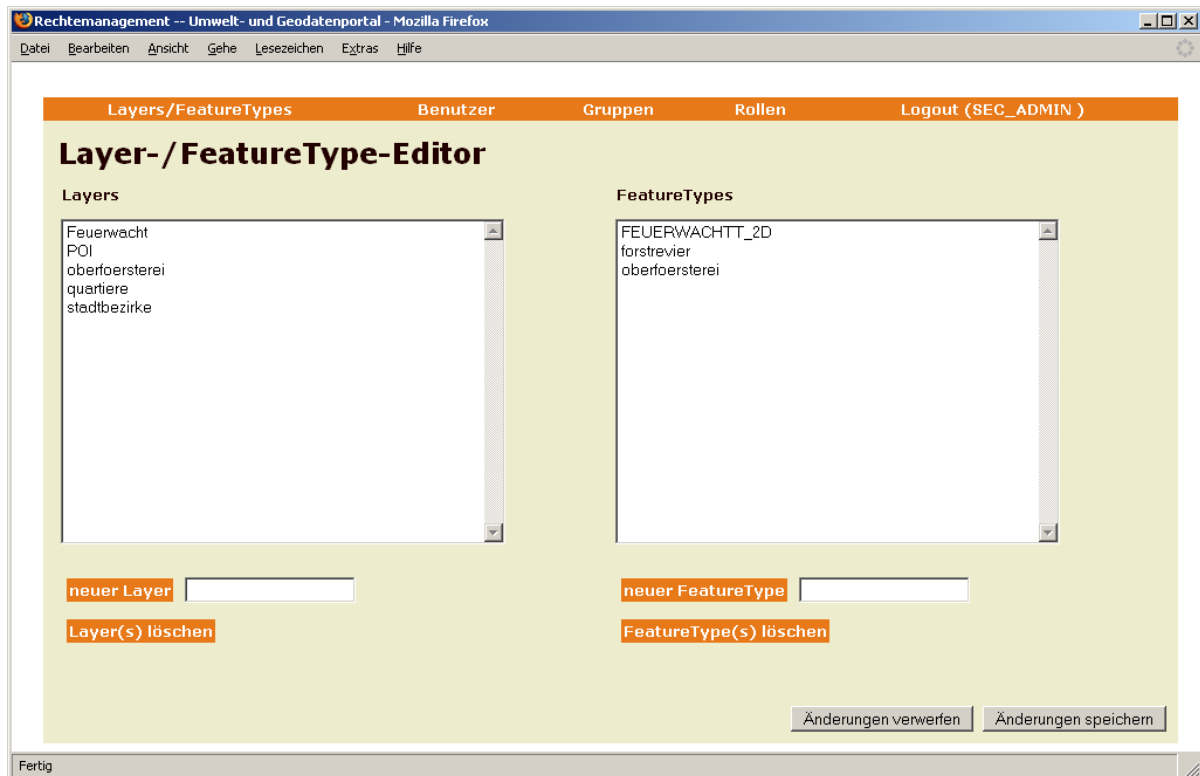


Figure 3: Form for input of new Layers and FeatureTypes as well as for deletion of existing Layers and FeatureTypes

Usage of this form is hopefully self-explanatory. A new Layer or FeatureType can be input in the fields below the lists of already known Layers/FeatureTypes and clicking the buttons on the left of the fields add them to the lists.

Hint: In case of using WFS in version 1.1.0 the FeatureTypes have to be specified including their namespaces. The syntax for this is as follows:

```
{http://www.deegree.org/app}:FT1
```

The namespace has to precede the FeatureType name and has to be enclosed by curly brackets.

By selecting entries in the lists and then using the button 'selektierte Layer löschen' respectively 'selektierte FeatureTypes löschen' the corresponding entries are deleted. All changes are only applied permanently to U3R after the button 'übernehmen' is pressed and the following dialogue is confirmed.

4.2 User Editor

The user editor allows for the creation of new users as well as deletion of existing users or editing of their corresponding data (Fig. 4). In regard to the data used to describe a user, the system is largely open. By adjusting the database schema, arbitrary attributes can be attached to a user. The following attributes are mandatory:

- user name (“Benutzername”; name used for logging into the system)
- password (“Passwort”)
- email address (“Email-Adresse”)

Additional but optional attributes might include first name (“vorname”) and last name (“Nachname”). These can be used to address a user, e.g. when sending an email message.

The layout of the user administration window is displayed in fig. 4. On its left side, a list of already registered users is displayed. Below the list there is a button and an input field for creation of new users and also a button allowing to delete selected users. To the right of the user list are six input fields corresponding to data about a particular user.

To modify the data of an existing user, the entries in the six input fields can simply be edited. Recall that the fields 'Benutzername', 'eMail' and 'Passwort' are mandatory. By clicking the button 'Benutzerdetails ändern' the changes are taken over in the list at the left.

To create a new user, a name has to be entered into the text field below the users list. Subsequently the button 'neuer Benutzer' has to be pressed. The empty data fields can then be filled in.

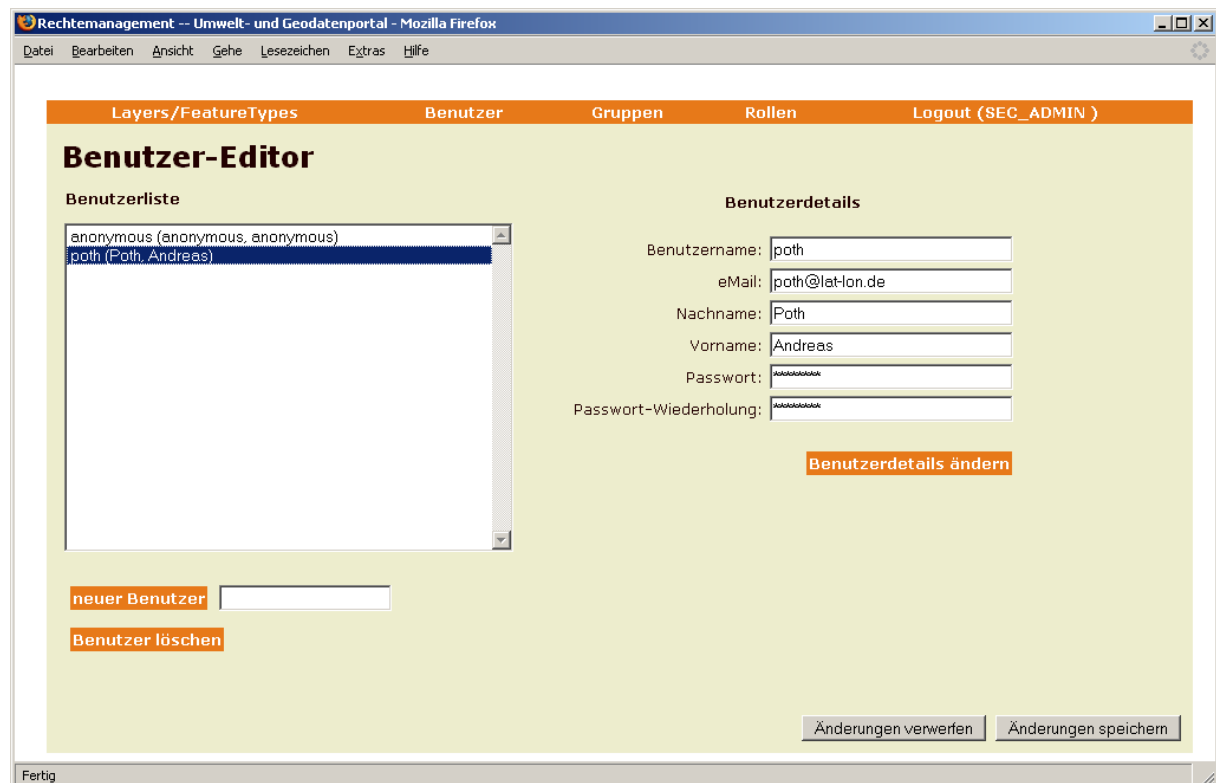


Figure 4: Formular for administration of users

As usual, all changes are only applied to the database after the 'Änderungen speichern' button is pressed.

4.2.1 Extension of the User Editor: assign WebMapContext

User administration can be extended from what has been outlined above by assigning a WebMapContext (WMC) as starting context. This extension is only available using U3R in conjunction with deegree iGeoPortal.

To activate this extension the following changes are necessary:

1. Definition of the parameter "configFile"
2. Configuration of "configFile"

4.2.1.1 Definition of the parameter configFile

In the file `security-controller.xml` (in directory `WEB-INF/conf/security/` or `WEB-INF/conf/drm-admin/`) a parameter is passed to the event "initUserEditor":

```
<event name="initUserEditor"
class="org.deegree.portal.standard.security.control.InitUserEditorListener"
next="usereditor.jsp">
  <!-- you might want to comment in this parameter in order to use
        the context chooser extension for iGeoPortal -->
  <!--
  <parameter>
    <name>configFile</name>
    <value>WEB-INF/conf/security/config_startcontext.xml</value>
  </parameter>
  -->
</event>
```

This parameter defines where the configuration file for the WMCs are to be found. This file is interpreted and processed during initialisation of the User Editor.

4.2.1.2 Adjustment of configFile

The configuration file defines which WMC-files can be assigned to a user via the User Editor (`availableWMC`), which context is a default context (`isDefault="1"`) and where the user's directory is to be found (`UserDirectory`).

```
<?xml version="1.0" encoding="UTF-8"?>
<deegree:Drm xmlns:deegree="http://www.deegree.org/security">
  <deegree:availableWMC>
    <!--
    must contain an entry for each start context that shall be used by
    the context chooser. Relative paths from this file to
    $iGeoPortal_home$ probably need to be adjusted on your system
    -->
    <deegree:WMC isDefault="1">
      <deegree:Name>startcontext</deegree:Name>
      <deegree:URL>
        ../../../../deegree2_igeo_std/WEB-INF/conf/igeoportal/wmc_start_utah.xml
      </deegree:URL>
    </deegree:WMC>
    <deegree:WMC>
      <deegree:Name>saltLakeCity</deegree:Name>
```

```

        <degree:URL>
        ../../../../degree2_igeo_std/WEB-INF/conf/igeoportal/wmc_saltlake.xml
        </degree:URL>
    </degree:WMC>
    <!-- add more context files here -->
</degree:availableWMC>
<degree:UserDirectory>
    ../../../../degree2_igeo_std/WEB-INF/conf/igeoportal/users
</degree:UserDirectory>
</degree:Drm>

```

For each WMC there are two entries; `<degree:Name>` and `<degree:URL>`. `<degree:Name>` is evaluated for display in the User Editor. Therefore, short, descriptive and unequivocal names are advise. `<degree:URL>` defines where the WMC-file can be found in the directory structure. Finally, `<degree:UserDirectory>` declares where the users directory of the corresponding iGeoPortal instance can be found.

N.B. All path definitions are relative, starting from the current directory.

4.2.1.3 Explanation regarding administration

The changes described in chapter 4.2.1.1 and 4.2.1.2 result in the following changes to the user interface:

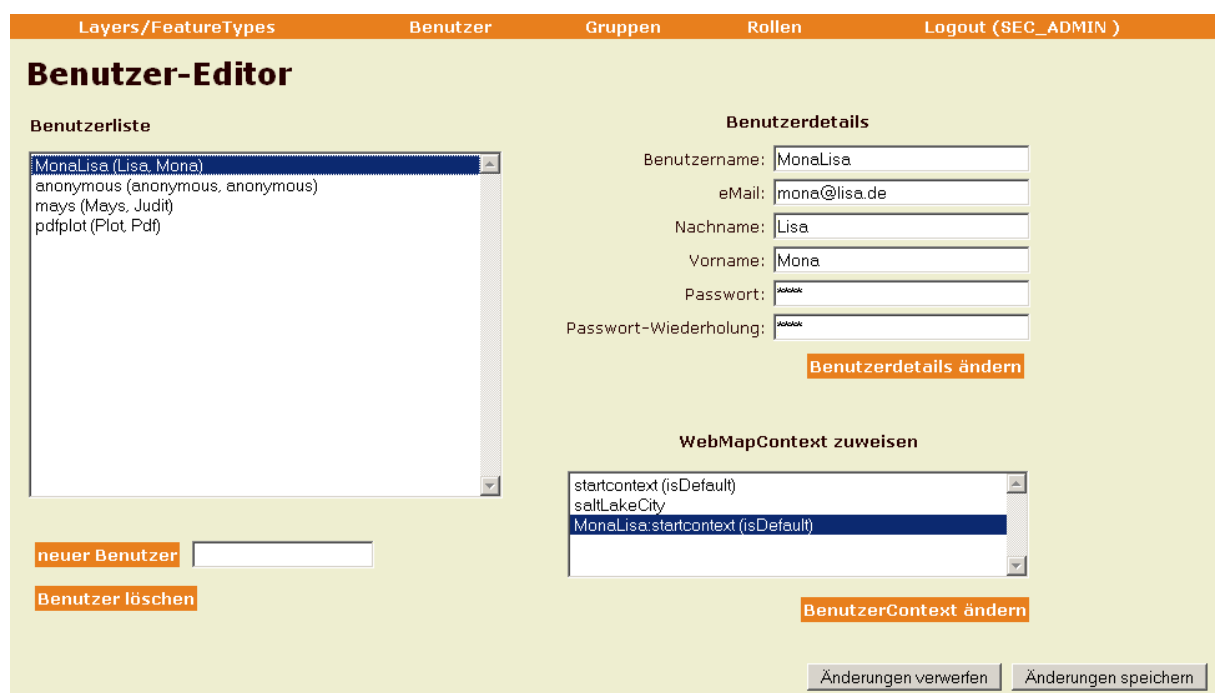


Figure 5: User Editor with Context Chooser

The standard User Editor is now extended to provide a way to assign specific WMC-files to particular users. The assigned WMC-files are uses as starting context, i.e. they define the map composition of a particular user at start-up.

The starting context of a user in regard to iGeoPortal is determined by the entry `STARTCONTEXT=wmc_datei.xml` in the file `context.properties`. First, it is checked if a user has such a file in their user directory (`WEB-INF/conf/igeoportal/users/nutzername/`). If not, the file with the same name in the `users`-directory is evaluated. This mechanism is used in the extension of the User Editor to assign a starting context to a user.

Administration screen:

In the list choice “WebMapContext zuweisen” (assign WebMapContext), all starting contexts are listed that are defined in `<degree:availableWMC>` configuration file. The context associated with the attribute `isDefault="1"` is marked accordingly in the list choice.

Additionally the list choice includes another entry that is created by combining the name of the current user and the WebMapContext assigned to him (entry `MonaLisa:startkontext(isDefault)` in figure 5). The suffix (`isDefault`) indicates, that the user `MonaLisa` does not have a `context.properties` file in their user directory, and that a default is used instead.

If a user should be assigned another starting context, it must be chosen from the list of available contexts and the button “Benutzerkontext ändern” (change user context) is pressed. The list then changes accordingly and the entry for the current user is changed to `username:newcontext`. This change is only applied to the database after the “Änderungen speichern” (apply changes) button is actioned.

If the attribute `isDefault="1"` is assigned to a new WebMapContext and Tomcat is started anew (see below), a corresponding message is shown when the administration screen is next started.

The configuration file:

Using the attribute `isDefault="1"` it is determined which Web Map Context in `WEB-INF/conf/igeoportal/users/context.properties` is saved as starting context. It is therefore advisable to assign this attribute to a context that is used as starting context by most users. If the administrator wants to change the default starting context, the attribute “`isDefault`” can be assigned to a another WebMapContext in the configuration file. After restarting Tomcat the value `STARTCONTEXT` will be changed accordingly in the above mentioned file.

If iGeoPortal was already configured without this extension, the start context defined in `context.properties` should under all circumstances be included in the list of available WebMapContexts and be assigned the attribute `isDefault="1"`. Once this is done it is unlikely that the starting context of the user will be changed accidentally.

As a consequence of the mechanisms as described, there can be a situation where a user is assigned the default starting context as WebMapContext without the appendix being defined behind the entry `Nutzername:KontextName`. This means that the user has a `personal.context.properties` file which links to the context which is also the `DefaultStartContext`, but the `DefaultStartContext` can be changed to a different context without changing this users context. The following details how this situation can occur:

Consider the case where there are 2 context files (context1 and context 2), and 3 users (user1, user2 and user3). The assignment of users to context files is as follows:

```
users/context.properties: STARTCONTEXT=./context1
users/user1/context.properties STARTCONTEXT=../context1
users/user2/context.properties: STARTCONTEXT=../context2
users/user3/      (no context.properties file of its own available)
```

Now the configuration file is created and context1 is assigned the suffix `isDefault="1"`. When starting the `drm-admin` the system recognizes that user1 and user2 already have `context.properties` files and their starting contexts are read and displayed. User3 without a `context.properties` file is assigned the default context:

```
user1:context1
user2:context2
user3:context1(isDefault)
```

N.B. Although user1 and user3 are assigned an identical context, the representation in the user interface differs. The suffix `(isDefault)` only is displayed in conjunction with the user that does not yet have a `personal.context.properties` file.

Now consider what happens when the entry `isDefault="1"` is moved from context1 to context2 and Tomcat and `drm-admin` are restarted. While opening the User Editor, user3 is informed that the default context has changed. Simultaneously the file `users/context.properties` is overwritten: `STARTCONTEXT=./kontext2`. This results in the following entries in the User Editor:

```
user1:context1
user2:context2
user3:context2(isDefault).
```

N.B. The context for user1 remains the same even though the default context was switched from context1 to context2. The context for user2 has also remained the same and although this is now the same as the DefaultMapContext, it does not get assigned the suffix (isDefault). The context of user3 keeps the suffix (isDefault), their starting context has changed from context1 to context2 and the user still does not have a personal context.properties file.

Thus care is needed by reference to the attributes isDefault="1" when changing the DefaultMapContext to ensure that the intended context changes occur.

4.3 Group-Editor

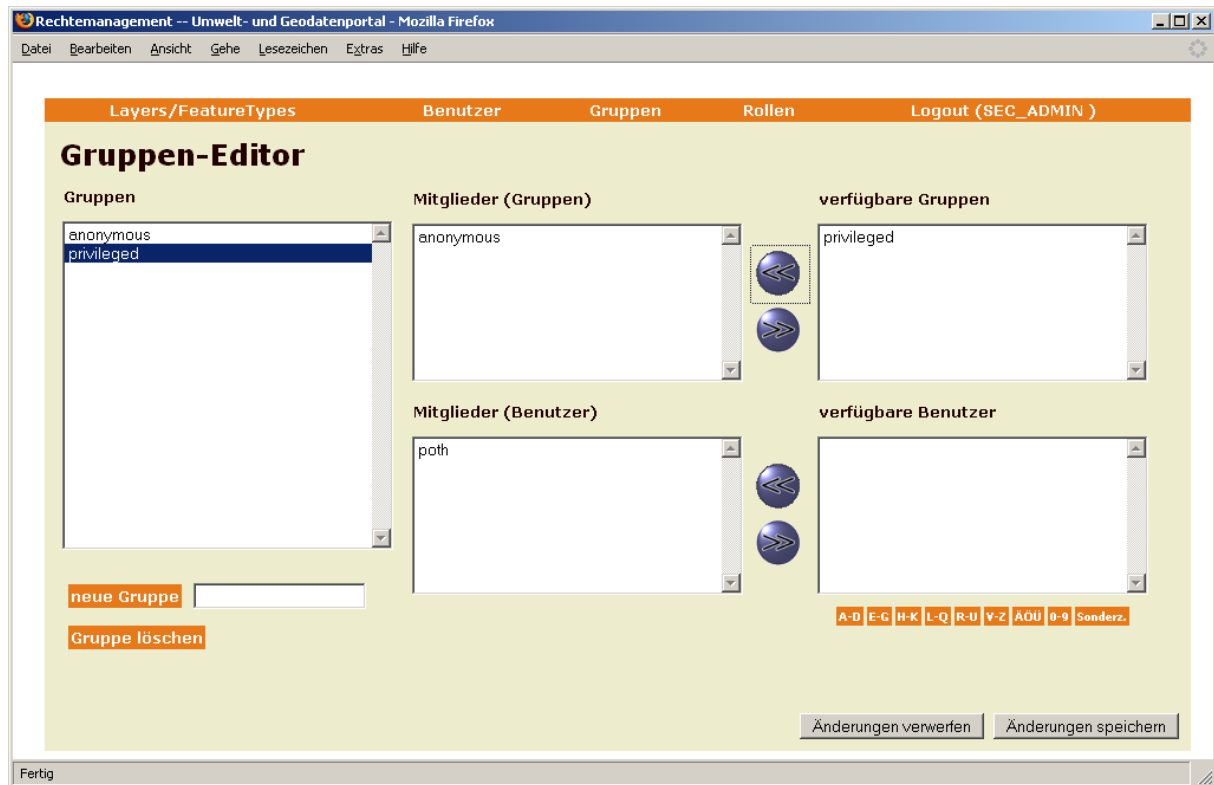
With U3R users can be grouped into a group and a number of groups can also be grouped. The GUI used to organise groups is displayed in figure 6.

On the left, there is a list of all registered groups ("Gruppen"). Below this is a text field and a button for creation of new groups and a button for deletion of existing groups. To the right of Gruppen there are additional GUI elements that can be used to edit the composition of the selected groups.

It is possible to assign an existing group to another group. In figure 6 the group 'anonymous' was assigned the group 'privileged'. Therefore all member of 'privileged' "inherit" all rights assigned to 'anonymous'. Using this mechanism it is possible to define tree or network-like relationships between groups. Network-structures come about with one or more cyclic relationships between groups, e.g 'privileged' might be a member of 'anonymous' and 'anonymous' a member of 'privileged'. This would result in a cycle in which each group is associated with itself.

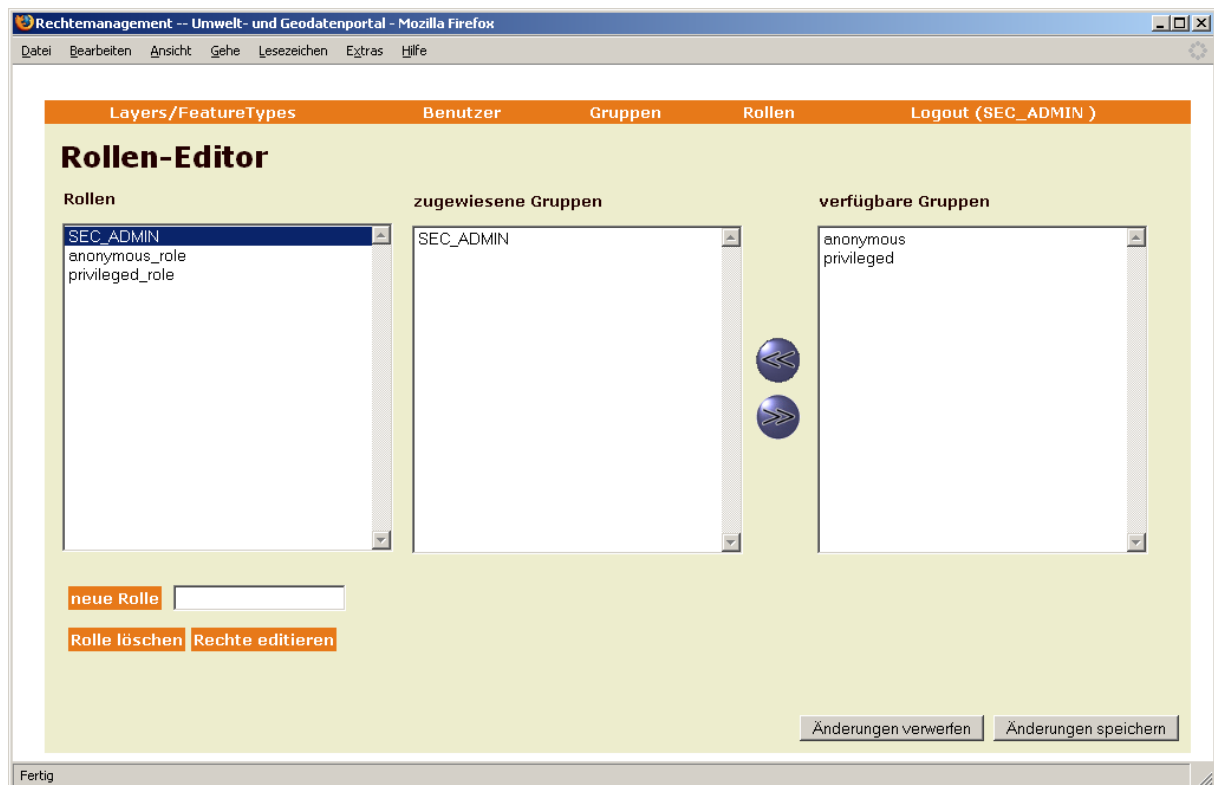
Direct or indirect cycles are permitted, but the user is informed in the GUI as this occurs and subsequently. deegree U3R has an internal mechanism for checking cycles that is able to detect complex situations. In this way, infinite regress results can be controlled.

The two lists in the lower half of figure 6 can be used to assign separate users to a group. The list of available users is organised and subdivided alphabetically to support finding/selecting unique users and to make sure that large numbers of users can also be handled by the system.



The changes are actuated in the system in the usual way.

4.4 Role Editor



The Role Editor (Fig. 7) is to associate selected groups with rights to datasets. This means that all direct and indirect members of a group associated with a role have all rights that are summarized under this role (see below).

4.4.1 Create a role

Input the role name in the field next to the button 'neue Rolle' (new role) and click the button to create a new role which appears in the list 'Rollen' (roles).

4.4.2 Delete a role

The role that is intended to be deleted has to be selected in the left of the three lists and the button 'Rolle löschen' (delete role) has to be activated. The confirmation is that the role is deleted from the list (as usual the change is only applied to the database after pressing 'übernehmen').

4.4.3 Editing role-group associations

Select the role to be edited in the left list. In the list 'zugewiesene Gruppen' (associated groups) the associated groups will then appear. At 'verfügbare Gruppen' (available groups) all groups known to the system are listed except those which are already associated with this role. To associate further groups with the role they have to be selected in the list to the right and the double arrow pointing to the left has to be pressed. The selected groups are then moved into the list in the middle. To delete associations between groups and roles, the roles to be deleted must be selected in the list in the middle and the double arrow pointing to the right has to be pressed.

4.4.4 Edit a role

To edit a right that is included in a role, the role has to be selected in the left list and then 'Rolle editieren' (edit role) pressed. This is only possible if no changes were done that are not yet stored. Otherwise first 'übernehme' (apply) and the 'abbrechen' (cancel) has to be used.

4.5 Rights Editor

The Rights Editor allows the association of individual rights that belong to a single role (Fig. 8) in the upper left part of the screen the FeatureTypes are listed that can be accessed by this role (respectively the groups/users associated with the role). On the upper right are the FeatureTypes for which access is denied. Using the buttons with the double arrows between the lists the rights-associations can be edited (analogous to the association between roles and groups).

For each FeatureType the transactions allowed for the current role can be defined using the three checkboxes below the list of the selected FeatureTypes. First a FeatureType has to be selected. Afterwards, rights for insert, delete and update of the associated FeatureType can be allocated or deallocated.

In the lower part the rights are edited that the role owns in regard to individual layers. The role does have access rights to the layers in the left list, but not the right list.

By pressing the button 'Änderungen speichern' (apply/store changes) all rights changes are made persistent. By using 'Änderungen verwerfen' all changes are cancelled.

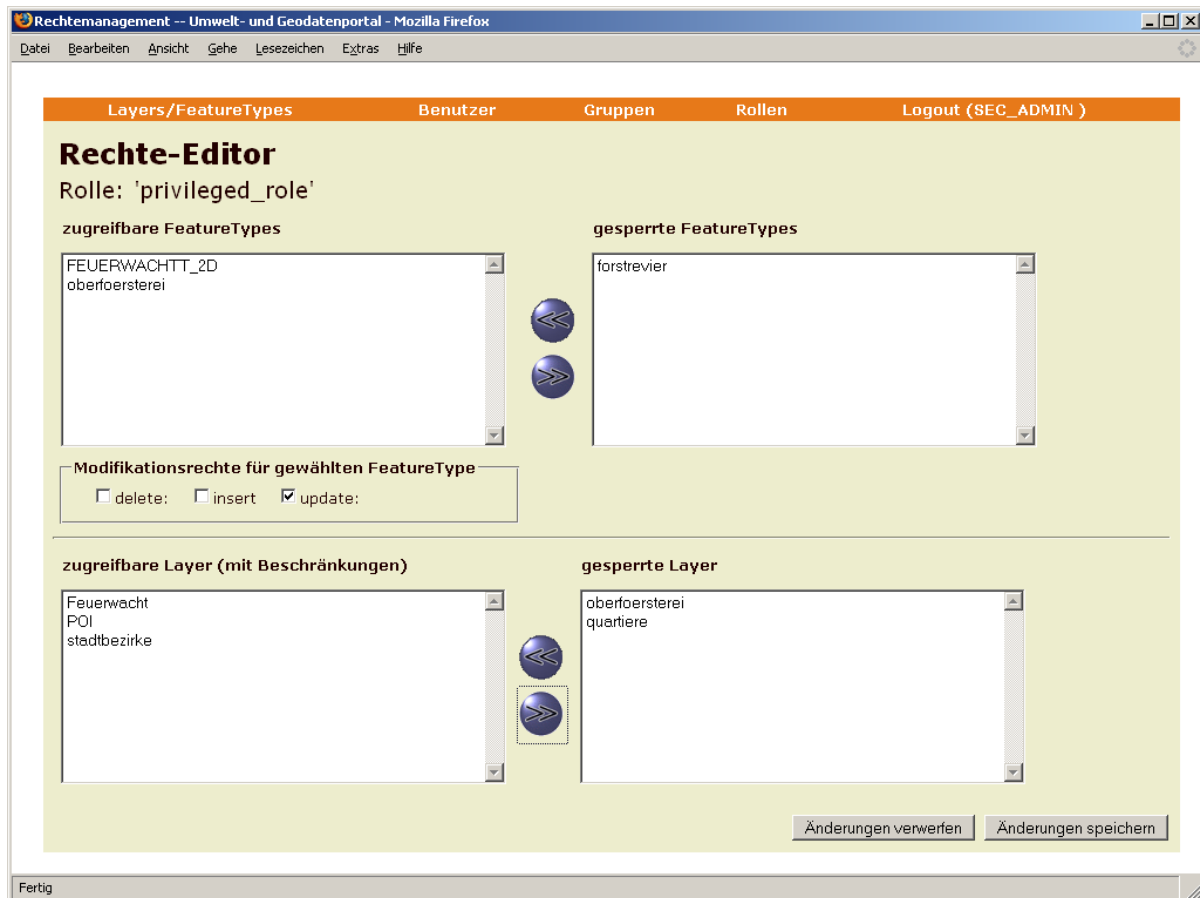


Figure 8: Rights Editor

5 Configuration using the command line

As an alternative to defining rights, roles, users and groups with the help of the web-interface of U3R, it is also possible to use a command line tool. This tool can be used programmatically to administer U3R, e.g. to automatically create great number of users or layers.

5.1 Program call

The tool can be started using the following program call:

```
$JAVA_HOME$/bin/java -classpath .;deegree2.jar;$database JDBC
driver$;$additional libraries$ org.deegree.tools.security.DRMAccess
```

The parameters that have to be appended to these program call are described in the following sections.

5.2 Common program parameters

Each operation / each call requires parameters for definition of the database connection.

```
-driver [JDBC driver] (e.g. sun.jdbc.odbc.JdbcOdbcDriver for an ODBC
database)
-logon jdbc:odbc:security [logon to database] (e.g. . ODBC name)
-user [user name] (optional)
-pw [users password] (optional)
```

5.3 known actions/operations

```
action (addUser, removeUser, addGroup, removeGroup, addRole,
removeRole, addUserToGroup, assignRoleWithGroup, addSecuredObject,
removeSecuredObject, assignRights, removeRights, clean)
```

Defines the action to be performed. Possible actions are listed in brackets.

addUser -> adds a user to the right management

```
-name [users login name]
-password [users password]
-firstName [first name of the user]
-lastName [last name of the user]
-email [email address of the user]
```

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action addUser -name lkee -password lkee01 -firstName Andreas  
-lastName Poth -email info@lat-lon.de
```

removeUser -> removes a user from the right management

-name [users login name]

Example:

```
java -classpath .;deegree.jar;$database JDBC driver$  
org.deegree.tools.security.DRMAccess -driver  
sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security -action  
removeUser -name latlon
```

addGroup -> adds a group to the right management system

-name [name of the group]

-title [title of the group]

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action addGroup -name Group1 -title TGroup1
```

removeGroup -> removes a group from the right management

-name [name of the group to be removed]

addRole -> adds a role to the right management system

-name [name of the role]

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action addRole -name Role1
```

removeRole -> removes a role from the right management

-name [name of the role to be removed]

addUserToGroup -> adds a user to a named group

-userName [name of the user]

-groupName [name of the group]

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action addUserToGroup -userName lkee -groupName Group1
```

assignRoleWithGroup -> assigns a group with a role

-groupName [name of the group]

-roleName [name of the role]

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action assignRoleWithGroup -groupName Group1 -roleName Role1
```

addSecuredObject -> adds a new secured object to the right management system

-soType [type of the secured object] (e.g. Layer, FeatureType, Coverage ...)

-soName [name of the secured object]

-soTitle [title of the secured object]

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action addSecuredObject -soType Layer -soName oberfoersterei  
-soTitle Oberfoersterei
```

removeSecuredObject -> removes a secured object from the right management system

-soType [type of the secured object] (z. B. Layer, FeatureType, Coverage ...)

-soName [name of the secured object]

assignRights -> assigns rights on a named secured object to a role

-constraints [comma separated list of absolute paths to filter encoding files]

-rights [comma separated list of rights to assign] (he number of rights must be equal to the number constraints)

-soName [name of the secured object]

-soType [type of the secured object]

-role [name of the role the rights shall be given to]

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security
-action assignRights -constraints -;-;- -soName oberfoersteriei
-soType Layer -role Role1 -rights
GetLegendGraphic;GetMap;GetFeatureInfo
```

```
.;deegree.jar org.deegree.tools.security.DRMAccess -driver
sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security -action
assignRights -constraints -;e:/temp/ComplexFilter.xml -soName
{http://www.deegree.org/app}:WPVS -soType Featuretype -role
anonymous_role -rights GetFeature,GetFeature_Response
```

ComplexFilter.xml:

```
<ogc:Filter xmlns:ogc="http://www.opengis.net/ogc">
  <ogc:And>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>horizontalAccuracy</ogc:PropertyName>
      <ogc:Literal>10</ogc:Literal>
    </ogc:PropertyIsEqualTo>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>verticalAccuracy</ogc:PropertyName>
      <ogc:Literal>1</ogc:Literal>
    </ogc:PropertyIsEqualTo>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>xslt</ogc:PropertyName>
      <ogc:Literal>file:/d:/temp/test.xsl</ogc:Literal>
    </ogc:PropertyIsEqualTo>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>instanceFilter</ogc:PropertyName>
```

```

<ogc:Literal>
  <![CDATA[<ogc:Filter xmlns:ogc="http://www.opengis.net/ogc"
    xmlns:app="http://www.deegree.org/app">
    <ogc:And>
      <ogc:PropertyIsGreaterThan>
        <ogc:PropertyName>app:dateOfBirth</ogc:PropertyName>
        <ogc:Literal>1820</ogc:Literal>
      </ogc:PropertyIsGreaterThan>
      <ogc:PropertyIsEqualTo>
        <ogc:PropertyName>
          app:placeOfBirth/app:Place/app:country/app:Country/app:name
        </ogc:PropertyName>
        <ogc:Literal>Germany</ogc:Literal>
      </ogc:PropertyIsEqualTo>
    </ogc:And>
  </ogc:Filter>]]></ogc:Literal>
</ogc:PropertyIsEqualTo>
</ogc:And>
</ogc:Filter>

```

The following types of secured objects are currently supported:

- Layer
- Featuretype
- MetadataSchema

The following rights are currently supported:

- access
- query
- delete
- delete_Response
- insert
- insert_Response
- execute
- update
- update_Response
- view
- grant
- GetMap
- GetMap_Response
- GetFeatureInfo
- GetFeatureInfo_Response

- GetLegendGraphic
- GetLegendGraphic_Response
- GetFeature
- GetFeature_Response
- DescribeFeatureType
- DescribeFeatureType_Response
- GetCoverage
- GetCoverage_Response
- DescribeCoverage
- DescribeCoverage_Response
- GetRecords
- GetRecords_Response
- GetRecordById
- GetRecordById_Response
- DescribeRecordType
- DescribeRecordType_Response

removeRights -> removes rights on a named secured object from a role

```
-rights [comma separated list of rights to remove.]
-soName [name of the secured object]
-soType [type of the secured object]
-role [name of the role the rights shall be given to]
```

Example:

```
java -classpath .;deegree.jar;$database JDBC driver$
org.deegree.tools.security.DRMAccess -driver
sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security -action
removeRights -soName stadtbezirke -soType Layer -role testAccess
-rights GetLegendGraphic
```

hasRight -> checks if a named user has a specific right

```
-userName [name of the user]
```

-password [the users password]
-soName [name of the secured object that shall be accessed]
-soType [type of the secured object](z. B. Layer, FeatureType, ...)
-right [right that shall be accessed] (z. B. GetMap, GetFeature, ...)

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action hasRight -user lkee -password lkee01 -soName oberfoersterei  
-soType Layer -right GetMap
```

clean -> cleans the complete right management system database by deleting all entries!

Example:

```
java -classpath .;deegree.jar org.deegree.tools.security.DRMAccess  
-driver sun.jdbc.odbc.JdbcOdbcDriver -logon jdbc:odbc:security  
-action clean
```

Appendix A: U3R database schema

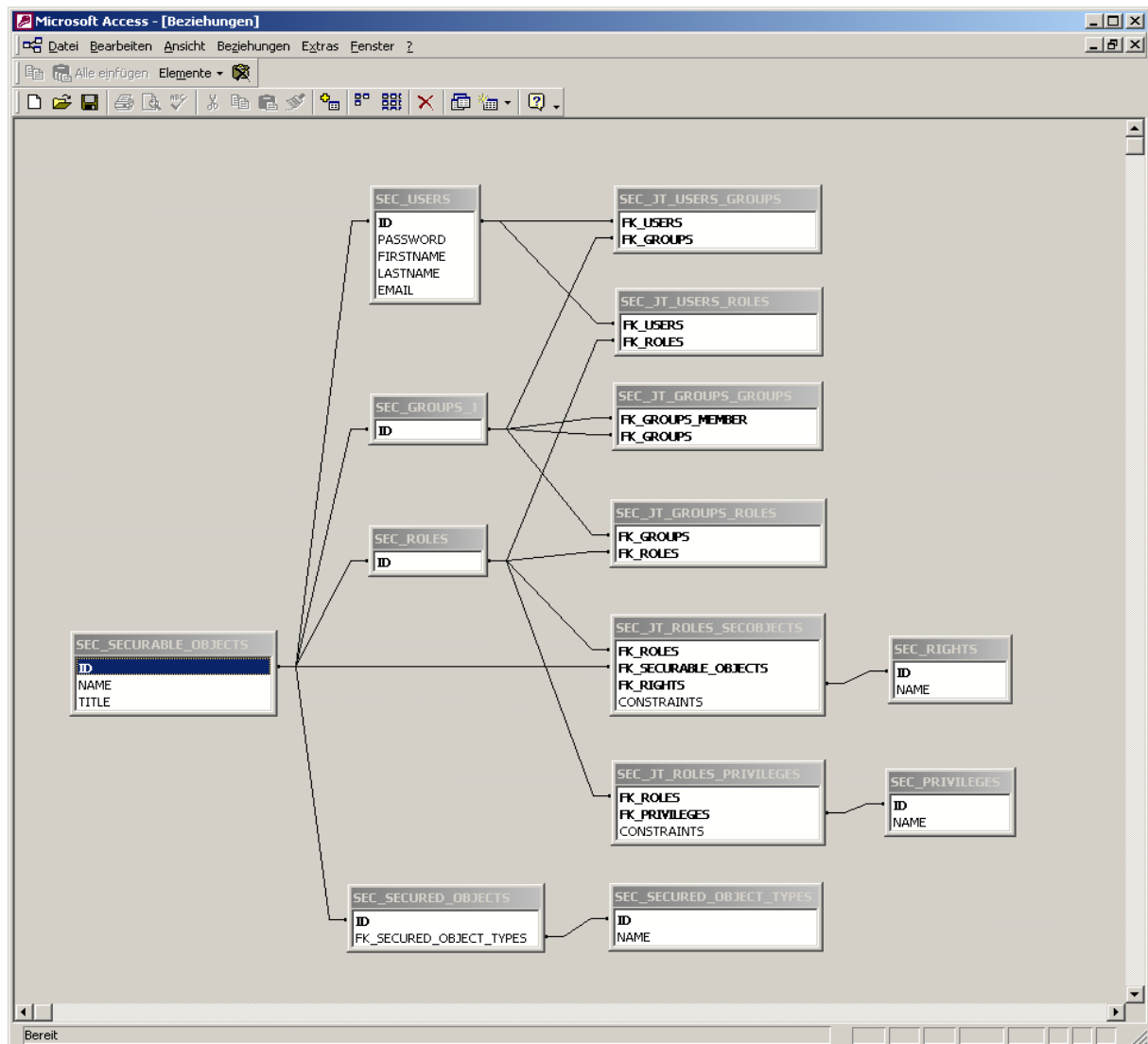


Figure 9 database schema used by U3R