Web Services for the CLARIN Component Registry

Introduction

The GUI of the CLARIN metadata editor and search services needs access to component and profile registered in the Component Registry. This REST specification should provide such access.

The Component Registry

The registry contains all CLARIN metadata components and metadata profiles used to describe all metadata. It is expected to contain around 1k components and around 1k profiles. Reuse of components and profiles is stimulated as much as possible.

Identification

Components and Profiles have an Id which is of type URI (this should be seen as a relaxed URI also allowing "almost" URIs like "hdl:1569/88128812".

Web Services

- Components
  - list all components or add one (GET, POST)
  - individual component (GET, DELETE)
- Profiles
  - list all profiles (GET, POST)
  - individual profile (GET, DELETE)
- Component usage (GET)
- Comments
  - list all comments for a component or profile or post one (GET, POST)
  - individual comment on profile or component (GET, DELETE)

Components:

Pattern	http://cmdregistry/rest/registry/components	
Examples	List all publicly registered components http://cmdregistry/rest/registry/components List all privately registered components http://cmdregistry/rest/registry/components?userspace=true	
Methods	GET	Get the component descriptions in the specified format. The GET-parameter **userspace** determines whether the components from the public registry or the private workspace are returned.

		Authentication required when set to true.
	POST	Create a component in the collection of components. Returning a response containing the description of the component or the possible reasons for failing registration. Requires authentication.
	HEAD, PUT, DELETE	Not allowed
Response	XML	Default.
	JSON	Can return JSON when request-header has Accept field set to application/json. In a curl command that would look like: `curl -i -H "Accept:application/json" -X GET http://localhost:8080/ComponentRegistry/rest/registry/components`

Return Values GET.

The returned values are:
ComponentDescriptions with the elements: id, name, description, registration date, creatorName, xlink:href to actual component and a groupName.
Return value examples in case of two registered components:

When xml is returned

```
<componentDescriptions>
  <componentDescription>
      <id>c_1259853703335</id>
      <description>Test file</description>
      <name>component-access.xml</name>
      <registrationDate>12/03/2009 16:21:43 CET</registrationDate>
      <creatorName>J,Smith</creatorName>
      <xlink:href>rest/registry/component/c_1259853703335</xlink:href>
      <groupName>imdi</groupName>
      </componentDescription>
      <componentDescription>
      <id>c_1259853703336</id>
      <description>Test file 2</description>
      <name>component-actor.xml</name>
      <registrationDate>12/03/2009 16:21:43 CET</registrationDate>
      <creatorName>J,Smith</creatorName>
      <xlink:href>
http://localhost:8080/ComponentRegistry/rest/registry/component/c_12598
53703336</xlink:href>
      <groupName>imdi</groupName>
  </componentDescription>
</componentDescriptions>
```

When json is returned

```
{
"componentDescription":[
{
```

```
"id":"c_1259853703335",
"description":"Test file",
"name":" component-access.xml ",
"registrationDate":"12/03/2009 16:21:43 CET ",
"creatorName":"J,Smith”,
"xlink:href":"
http://localhost:8080/ComponentRegistry/rest/registry/component/c_12598
53703335"},
"groupName":"imdi"
{
"id":"c_1259853703336",
"description":"Test file",
"name":" component-actor.xml ",
"registrationDate":"12/03/2009 16:21:43 CET ",
"creatorName":"J,Smith”,
"xlink:href":"
http://localhost:8080/ComponentRegistry/rest/registry/component/c_12598
53703336"},
"groupName":"imdi"
}
```

Note the “id”/”xlink:href” which is generated by the application and needs to be used to do a successful POST of the component data.

POST method.

A post method can be used to add Components to the registry. A post request is a Multipart Form Data consisting of the fields: “name”, “description”, “creatorName”, “group” and stream “data” part which is the uploaded component file which will be registered.
The request will be validated and the result will be wrapped in a RegisterResponse.

A successful POST contains the description created:

```
 <registerResponse registered="true" isProfile="false">
    <errors/>
    <description xsi:type="componentDescription"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <id>c_1259853703337</id>
        <description>myD</description>
        <name>Name</name>
        <registrationDate>myDate</registrationDate>
        <creatorName>myC</creatorName>
        <xlink:href>
http://localhost:8080/ComponentRegistry/rest/registry/component/c_12598
53703337</xlink:href>
        <groupName>imdi</groupName>
    </description>
</registerResponse>
```

An unsuccessful POST contains the errors explaining why it is not accepted:

```
<registerResponse registered="false" isProfile="false">
    <errors>
        <error>Error 1</error>
        <error>Error 2</error>
    </errors>
</registerResponse>
```

Profiles:

Pattern	http://cmdregistry/rest/registry/profiles	
Examples	List all publicly registered profiles http://cmdregistry/rest/registry/profiles List all registered profiles that are marked to be shown in metadata editors http://cmdregistry/rest/registry/profiles?mdEditor=true	
Methods	GET	Get the profiles in the specified format. The GET-parameter **userspace** determines whether the profiles from the public registry or the private workspace are returned. Authentication required when set to true. When GET-parameter **mdEditor** (default value: *false*) is set to *true*, only profiles with showInEditor = true are shown.
	POST	Create a profile in the collection of profiles. Returning a response containing the description of the profiles or the possible reasons for failing registration. Requires authentication.
	HEAD, PUT, DELETE	Not allowed
Response	XML	Default.
	JSON	Can return JSON when request-header has Accept field set to application/json.

Return Values GET.

ProfileDescriptions with the elements: id, name, description, registration date, creatorName, xlink:href to actual profile, commentsCount and a boolean showInEditor indication.

Return value examples are similar as previously described only an xml example is shown:

```
</profileDescriptions>
  <profileDescription>
      <id>p_1257850388373</id>
      <description>Test</description>
      <name>TestProfile</name>
      <registrationDate>Tue Nov 10 11:53:08 CET 2009</registrationDate>
      <creatorName>J. Smith</creatorName>
      <xlink:href>
      http://cmdregistry/rest/registry/profile/p_1257850388373</xlink:h
      ref>
      <commentsCount>1</commentsCount>
      <showInEditor>true</showInEditor>
```

```
  </profileDescription>
</profileDescriptions>
```

POST method.

Similar to post of components.

Individual component:

Pattern	http://cmdregistry/rest/registry/components/<component-id> or http://cmdregistry/rest/registry/components/<component-id>/xml http://cmdregistry/rest/registry/components/<component-id>/xsd	
Examples	Get registered component http://cmdregistry/rest/registry/components/clarin.eu:cr1:p0001	
Methods	GET	Get the component in its xml format. The GET-parameter **userspace** determines whether the component should be retrieved from the public registry or the private workspace. Authentication required when set to true.
	DELETE	Deletes the component from the registry. Only allowed if owner or admin.
	HEAD, PUT, POST	Not allowed
Response	XML	Default.
	JSON	Can return JSON when request-header has Accept field set to application/json. (Not supported with /xml and /xsd types.)

Return Values GET.

The returned value is:
The xml representation of the specified component. Specification of the components structure can be found in [http://trac.clarin.eu/browser/metadata/trunk/toolkit/general-component-schema.xsd].

Return values when specifying /xml or /xsd.
A pretty printed xml representation of the component (useful for presenting the xml to a user)or the xsd schema of the component.

Individual profile:

Pattern	http://cmdregistry/rest/registry/profiles/<profile-id> or http://cmdregistry/rest/registry/profiles/< profile -id>/xml http://cmdregistry/rest/registry/profiles/< profile -id>/xsd	
Examples	Get registered profile http://cmdregistry/rest/registry/profiles/clarin.eu:cr1:p0001	
Methods	GET	Get the profile in its xml format. The GET-parameter **userspace** determines whether the profile should be retrieved from the public registry or the private workspace. Authentication required when set to true.
	DELETE	Deletes the profile from the registry. Only allowed if owner or admin.
	HEAD, PUT, POST	Not allowed
Response	XML	Default.
	JSON	Can return JSON when request-header has Accept field set to application/json. (Not supported with /xml and /xsd types.)

Return Values GET.

See get Component.

Component usage:

Pattern	http://cmdregistry/rest/registry/components/usage/< component –id>	
Examples	Get profiles and components that reference component c0001 http://cmdregistry/rest/registry/components/usage/clarin.eu:cr1:c0001	
Methods	GET	Get descriptions for the referencing profiles and components in the specified format.
	HEAD, PUT, POST, DELETE	Not allowed
Response	XML	Default.
	JSON	Can return JSON when request-header has Accept field set to application/json

Return Values GET.

See list all components/profiles. Notice that the list type is abstractDescription (which profileDescription and componentDescription extend)

Comments for component or profile:

Pattern	http://cmdregistry/rest/registry/components/< component –id>/comments http://cmdregistry/rest/registry/profile/< profile–id>/comments	
Examples	Get comments of public component c0001 http://cmdregistry/rest/registry/components/clarin.eu:cr1:c0001/comments Get comments of private profile c0001 http://cmdregistry/rest/registry/profiles/clarin.eu:cr1:c0001/comments?userSpace=true	
Methods	GET	Get comments for the referenced profile or component. The GET-parameter **userspace** determines whether the component/profile should be retrieved from the public registry or the private workspace. Authentication required when set to true.
	POST	Create a comment on the specified profile or component. Body should be a valid comment.
Response	XML	Default.
	JSON	Can return JSON when request-header has Accept field set to application/json

Return Values GET.

The return values are:
Comment elements with child elements 'id', 'comments' (containg the text content), 'commentDate', 'userName' (the display name, userId not exposed) and depending on whether the comment concerns a profile or component a 'profileDescriptionId' or 'componentDescriptionId'.

```
<comments>
      <comment>
            <comments>Content of the comment</comments>
            <commentDate>2012-01-03T13:13:03+00:00</commentDate>
            <profileDescriptionId>clarin.eu:cr1:p_1297242111880</profil
      eDescriptionId>
            <id>1</id>
            <userName>Joe C. Poster</userName>
      </comment>
</comments>
```

POST method.

A post method can be used to add comments to the component/profile. A post request is a Multipart Form Data (for consistency with profile/component posts) consisting of just a stream "data" part that contains the comment serialized as XML.

The request will be validated and the result will be wrapped in a RegisterResponse like components and profiles (see above). An unsuccessful POST contains the errors explaining why it is not accepted, similar to comments and profiles.

Individual comment:

Pattern	http://cmdregistry/rest/registry/profiles/<profile-id>/comments/<comment-id> or http://cmdregistry/rest/registry/components/<component-id>/comments/<comment-id>	
Examples	Get comment with id 42 on public component c0001 http://cmdregistry/rest/registry/components/clarin.eu:cr1:c0001/comments/42 Get comment with id 66 on private profile p0001 http://cmdregistry/rest/registry/profiles/clarin.eu:cr1:p0001/comments/66?userspace=true	
Methods	GET	Get the comment in its xml format.
	DELETE	Deletes the comment from the registry. Only allowed if poster or admin.
	HEAD, PUT, POST	Not allowed
Response	XML	Default.
	JSON	Can return JSON when request-header has Accept field set to application/json. (Not supported with /xml and /xsd types.)

Up to date WADL:

The REST service can generate a WADL of the current implemented services by accessing: http://cmdregistry/rest/application.wadl.