Data Portal Version 3

Session Manager Web Service

Author: **Lisa Blanshard**

Doc version: **0.1**

Date: **February 2003**

CONTENTS

Data Portal – Session Manager Web Service

1. DESCRIPTION

The Session Manager Module monitors user's sessions. It uses a postgres database to keep track of user sessions. On user login a new session is created and set to the current time. This time is updated each time any service checks if a session is current. The session is removed when a user logs out.

A session is considered valid if it exists in the database (user has not logged out) and the timestamp is less than the timeout period (currently configured to be 15 minutes).

2. WEB SERVICES OFFERED

`Integer` **`startSession`**`(String user, org.w3c.dom.Element permissionList)`

Adds a new session with userName and the user's permissions to the database. The permissions list the facilities that the user has access to. For each facility, it lists the read access to metadata and data (true or false). An example follows:

```
<UserAccessPrivilege>
       <User>lisa_blanshard</User>
       <Facility name="BADC access="t">
              <MetaData>t</MetaData>
              <Data>f</Data>
       </Facility>
       <Facility name="MPIM" access="t">
              <MetaData>f</MetaData>
              <Data>f</Data>
       </Facility>
</UserAccessPrivilege>
```

Finally it returns a new session identifier currently generated by a postgres database sequence.

Note that it only stores details in the database about facilities that the user has access to. Any facilities in the permissions where access="f" are ignored

`Boolean` **`isValid`**`(Integer sid)`

Checks if the sessionID is valid i.e the user has logged in and the session has not timed out. If the session is valid it updates the timestamp in the database. If the session has timed out it throws a SessionTimedoutException.

`org.w3c.dom.Element` **`getPermissions`**`(Integer SID)`

Checks if the sessionID is valid i.e the user has logged in and the session has not timed out. If the session is valid it updates the timestamp in the database. It then

returns the permissions (as described above). If the session has timed out it throws a SessionTimedoutException.

`String` **`getDName`**`(Integer sid)`

Checks if the sessionID is valid i.e the user has logged in and the session has not timed out. If the session is valid it updates the timestamp in the database. It then returns the user name or distinguished name (whichever was passed in startSession). If the session has timed out it throws a SessionTimedoutException.

`void` **`endSession`**`(Integer sid)`

Removes the session and permissions from the database.

2.1 Services that use the Session Manager services

The following modules call the Query and Reply's services:

- Authentication & Authorisation uses startSession()
- Shopping Cart uses getDName()
- Query & Reply uses getPermissions()
- Web Interface uses getPermissions()

2.2 Services that the Query and Reply module calls

The Session Manager service calls no other services.

3. USE CASES

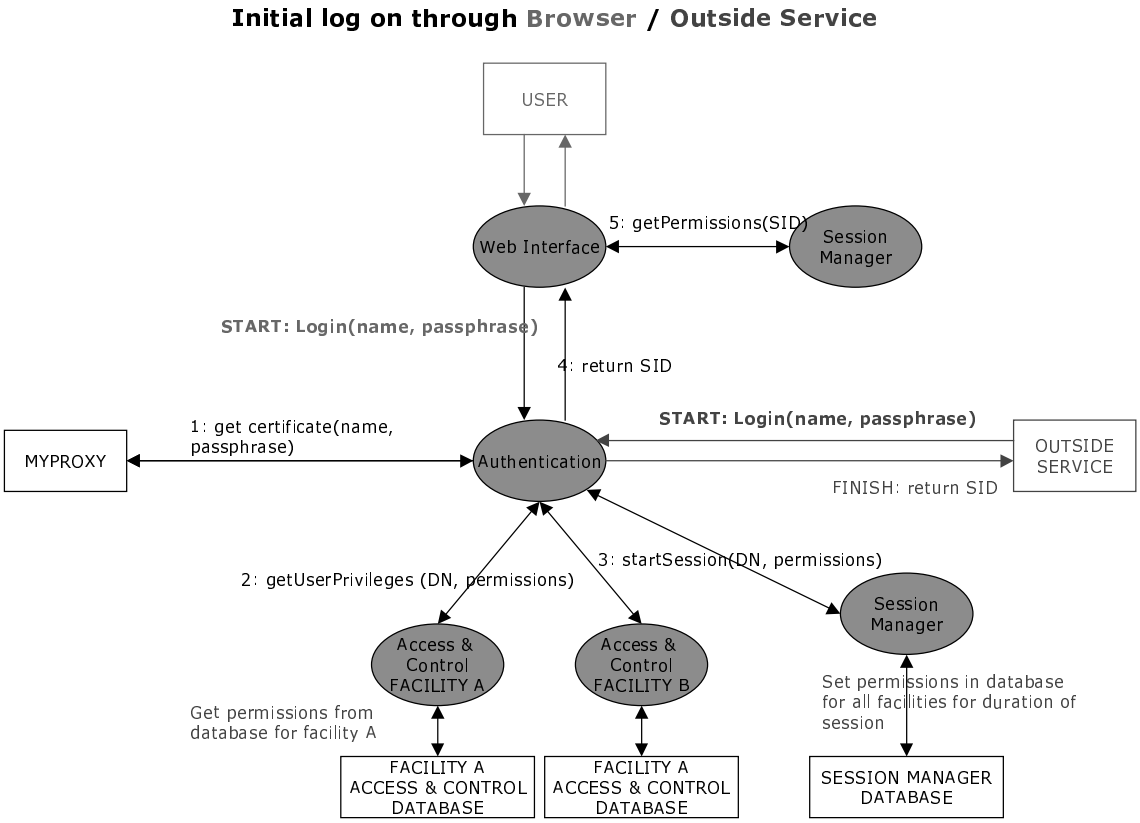

Figure 1 Authentication & Authorisation calls SessionManager:startSession(), Web Interface calls SessionManager:getPermissions()

3.1 Login through Browser

User logs in using a standard web browser. The Web Interface calls the Authentication & Authorisation service that gets a certificate from the MY PROXY server. The AA service then requests the user's permissions from each Access & Control module (one per facility) and these are collated. Finally it calls startSession() on the Session Manager and receives a new session identifier.

1. Request a session identifier from the database

The Session Manager requests the next value from a database sequence.

2. Create a new session

Using the new session identifier, the Session Manager inserts a new row into the SESSION table with the current timestamp. It then creates new rows in the table

SESSION_ACCESS_RIGHTS representing the access rights for the user (see 4 Database Tables).

3. Return the new session identifier

The new session identifier is returned to the Authentication & Authorisation service which then returns it to the Web Interface.

3.2 Login through Outside Service

If an Outside Service has requested the login on behalf of a user or application, the Authentication service behaves in the same way, apart from checking the facilities that the user has access to.

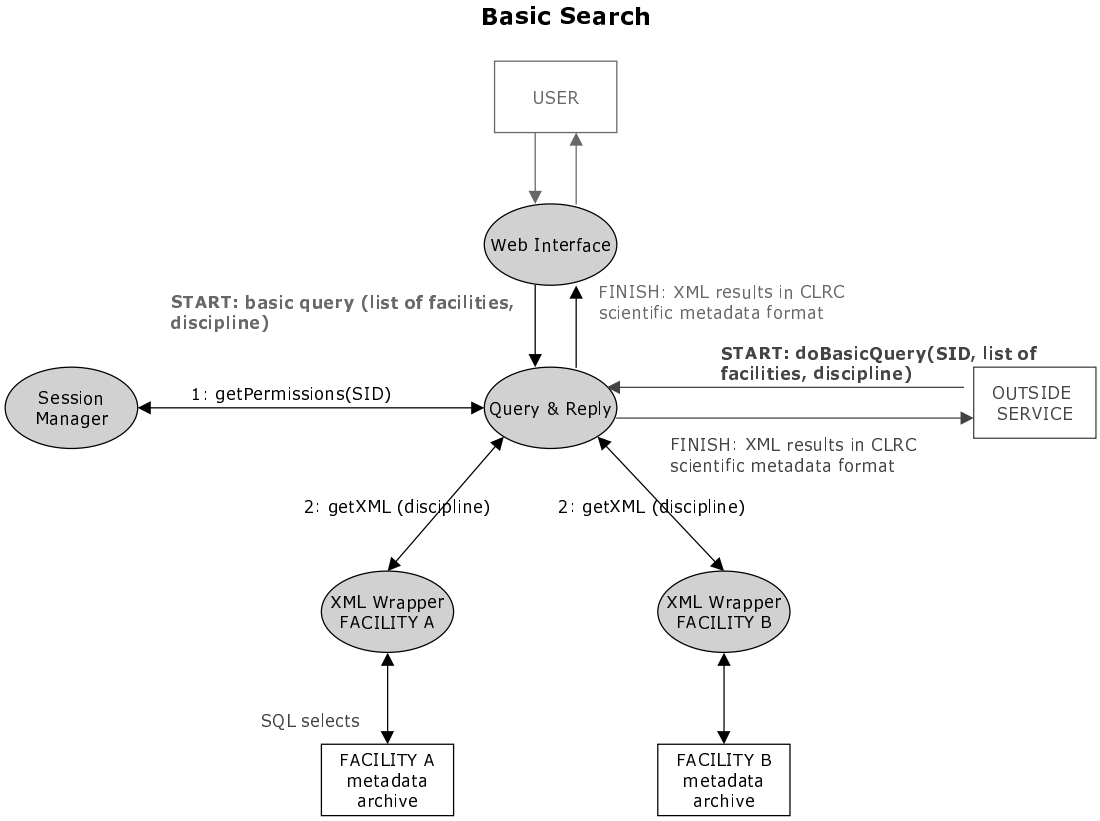

Figure 2 Query & Reply calls SessionManager:getPermissions()

3.3 Basic Search through a browser or outside service

User has already logged on and is authenticated, using a standard web browser. Then user has requested a basic search and chosen one or more facilities and a discipline and pressed submit. The Web Interface handles the request and and passes it to the

Query & Reply Module. Note that the user can only select facilities that he has access to since the Web Interface has already determined these following login.

The Query & Reply service then asks the Session Manager for the facilities that the user has access to using the getPermissions() method.

1. The Session Manager uses the SID to select the data from the SESSION and SESSION_ACCESS_RIGHTS tables.

2. The permissions are put into a new XML document as descibed above.

The Query & Reply sends the discipline to the XML Wrapper for each facility that the user has access to simultaneously. It waits for a configurable number of seconds for the reply in XML that represents the results from the remote metadata archive relating to the discipline chosen. Once all the replies are received (or timed out), they are collated into a single document. The format of the replies from the wrappers and the final document is the CLRC Scientific Metadata Format.

The final xml document is returned to the calling service.

If an Outside Service has requested a basic search it must first have called Authentication and Authorisation to login and receive a SID (Session Identifier). The Query & Reply service may then be called in the normal way.

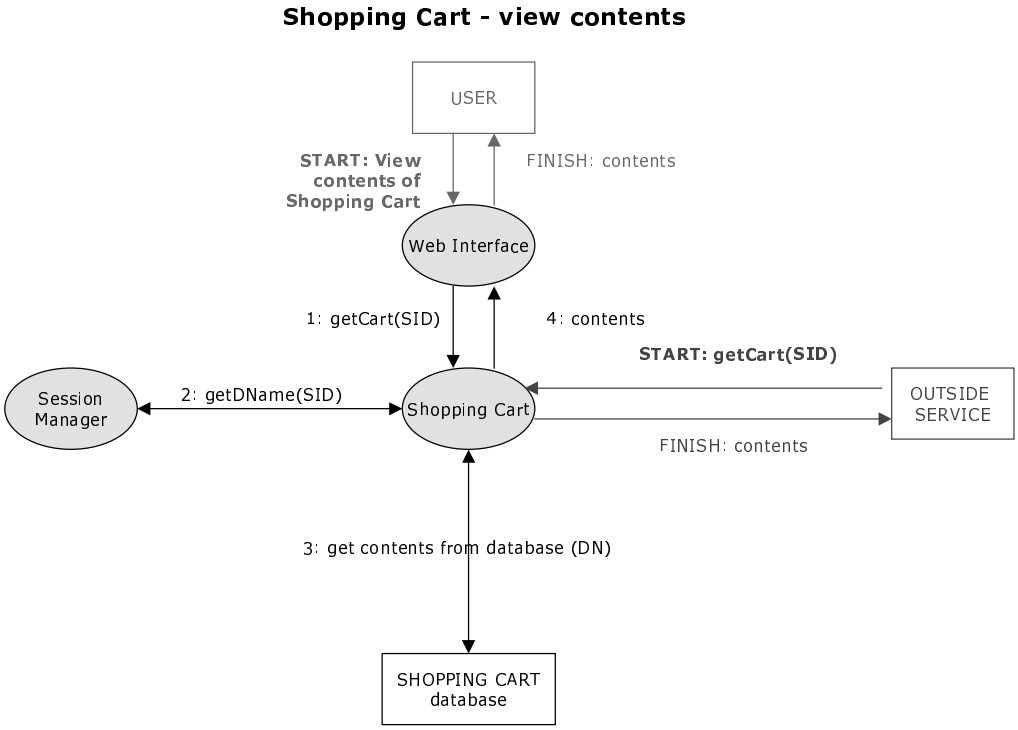

Figure 3 Shopping Cart calls Session Manager:getDName()

A user can add or remove links to the data to a personal shopping cart for later download. He may then view the contents of the shopping cart. In each case the user must have already authenticated and received a valid session identifier. The shopping cart then calls the Session Manager:getDName() to check if the session is valid and to get the Distinguished Name. This is so that the shopping cart record can be identified in the Shopping Cart database.

4. DATABASE TABLES

4.1 Table SESSION

Column	Key	Type	Description	Example
SID	Y	NUMBER	Unique session id – valid for current session	1234
USER_ID		STRING(200)	Distinguished name from the certificate (unique user id) used to initiate this session	CN=Lisa Blanshard,L=DL,OU=CLRC,O=eScience,C=UK
LAST_ACCESSED		DATE	Date and time a data portal service was last requested. Used to see if the session has timed out.	

4.2 Table SESSION_ACCESS_RIGHTS

Column	Key	Type	Description	Example
SID	Y	NUMBER	Unique for current session	1234
FACILITY_CODE		STRING(10)	Facility user has access to	BADC
METADATA_ACCESS		BOOLEAN	Has the user access to this facility's metadata?	
DATA_ACCESS		BOOLEAN	Has the user access to this facility's data?	

4.3 Relationship

One SESSION to many SESSION_ACCESS_RIGHTS