Automating Production of Cross Media Content for Multi-channel Distribution

www.AXMEDIS.org

DE3.1.2.3.9

Specification of AXMEDIS database and query support, update of DE3.1.2.2.9

Version: 1.5
Date: 16 July 2007
Responsible: EXITECH (revised and approved by coordinator)

Project Number: IST-2-511299 Project Title: AXMEDIS Deliverable Type: report Visible to User Groups: yes Visible to Affiliated: yes Visible to the Public: yes
Deliverable Number: DE3.1.2.3.9 Contractual Date of Delivery: M34 Actual Date of Delivery: 30/06/2007 Title of Deliverable: Specification of AXMEDIS database and query support, update of DE3.1.2.2.9 Work-Package contributing to the Deliverable: WP3.1 Task contributing to the Deliverable: WP3, WP2 Nature of the Deliverable: report Author(s): EXITECH, UPC (FUPF)

Abstract: this part includes the specification of components, formats, databases and protocol related to the AXMEDIS Framework area related to database and query support. It deals with the problems related to the Database Area and Data Gathering and therefore the specification of the database and the modalities to access data in the AXDB are reported together with the related tools such as crawling system for importing in AXMEDIS existing contents in user CMS; Query support for allowing user and tools to query the system; support for storing and querying licenses; support for automatic generation of licenses and contracts; production on demand and its relationships with gathering and queries.

Keyword List: Database, query, crawler, production on demand, licenses, PAR

1. All Works shall bear a clear notice asserting the Licensor's Copyright. The notice shall use the wording employed by the Licensor in its own copyright notice unless the Licensor otherwise instructs licensees.

6. **WARRANTY**

1. The Licensor warrants the licensee that the present license is issued on the basis of full Copyright ownership or re-licensing agreements granting the Licensor full licensing and enforcement power.
2. For the avoidance of doubt the licensee should be aware that although the Copyright in the documents is given under warranty this warranty does not extend to the content of any document which may contain references or specifications or technologies that are covered by patents (also of third parties) or that refer to other standards. AXMEDIS is not responsible and does not guarantee that the information contained in the document is fully proprietary of AXMEDIS consortium and/or partners.
3. Licensee hereby undertakes to the Licensor that he will, without prejudice to any other right of action which the Licensor may have, at all times keep the Licensor fully and effectively indemnified against all and any liability (which liability shall include, without limitation, all losses, costs, claims, expenses, demands, actions, damages, legal and other professional fees and expenses on a full indemnity basis) which the Licensor may suffer or incur as a result of, or by reason of, any breach or non-fulfillment of any of his obligations in respect of this License.

7. **INFRINGEMENT**

1. Licensee undertakes to notify promptly the Licensor of any threatened or actual infringement of the Copyright which comes to licensee notice and shall, at the Licensor's request and expense, do all such things as are reasonably necessary to defend and enforce the Licensor's rights in the Copyright.

8. **GOVERNING LAW AND JURISDICTION**

1. This Licence shall be subject to, and construed and interpreted in accordance with Italian law.
2. The parties irrevocably submit to the exclusive jurisdiction of the Italian Courts.

Please note that:

- You can become affiliated with AXMEDIS. This will give you the access to a huge amount of knowledge, information and source code related to the AXMEDIS Framework. If you are interested please contact P. Nesi at nesi@dsi.unifi.it. Once affiliated with AXMEDIS you will have the possibility of using the AXMEDIS specification and technology for your business.
- You can contribute to the improvement of AXMEDIS documents and specification by sending the contribution to P. Nesi at nesi@dsi.unifi.it
- You can attend AXMEDIS meetings that are open to public, for additional information see WWW.axmedis.org or contact P. Nesi at nesi@dsi.unifi.it

Table of Content

1 Executive Summary and Report Scope

The full AXMEDIS specification document has been decomposed in the following parts:

DE number	Deliverable title	responsible
DE3.1.2.3.1	Specification of General Aspects of AXMEDIS framework AXMEDIS-DE3-1-2-3-1-Spec-of-AX-Gen-Asp-of-AXMEDIS-framework	DSI
DE3.1.2.3.2	Specification of AXMEDIS Command Manager AXMEDIS- DE3-1-2-3-2-Spec-of-AX-Cmd-Man	DSI
DE3.1.2.3.3	Specification of AXMEDIS Object Manager and Protection Processor AXMEDIS-DE3-1-2-3-3-Spec-of-AXOM-and-ProtProc	DSI
DE3.1.2.3.4	Specification of AXMEDIS Editors and Viewers AXMEDIS-DE3-1-2-3-4-Spec-of-AX-Editors-and-Viewers	DSI
DE3.1.2.3.5	Specification of External AXMEDIS Editors/Viewers and Players AXMEDIS-DE3-1-2-3-5-Spec-of-External-Editors-Viewers-Players	DSI
DE3.1.2.3.6	Specification of AXMEDIS Content Processing AXMEDIS-DE3-1-2-3-6-Spec-of-AX-Content-Processing	DSI
DE3.1.2.3.7	Specification of AXMEDIS External Processing Algorithms AXMEDIS-DE3-1-2-3-7-Spec-of-AX-External-Processing-Algorithms	FHGIGD
DE3.1.2.3.8	Specification of AXMEDIS CMS Crawling Capabilities AXMEDIS-DE3-1-2-3-8-Spec-of-AX-CMS-Crawling-Capab	DSI
DE3.1.2.3.9	Specification of AXMEDIS database and query support AXMEDIS-DE3-1-2-3-9-Spec-of-AX-database-and-query-support	EXITECH
DE3.1.2.3.10	Specification of AXMEDIS P2P tools, AXEPTool and AXMEDIS AXMEDIS-DE3-1-2-3-10-Spec-of-AXEPTool-and-AXMEDIA-tools	DSI
DE3.1.2.3.11	Specification of AXMEDIS Programme and Publication tools AXMEDIS-DE3-1-2-3-11-Spec-of-AX-Progr-and-Pub-tool	UNIVLEEDS
DE3.1.2.3.12	Specification of AXMEDIS Workflow Tools AXMEDIS-DE3-1-2-3-12-Spec-of-AX-Workflow-Tools	UR
DE3.1.2.3.13	Specification of AXMEDIS Certifier and Supervisor and networks of AXCS AXMEDIS-DE3-1-2-3-13-Spec-of-AXCS-and-networks	DSI
DE3.1.2.3.14	Specification of AXMEDIS Protection Support AXMEDIS-DE3-1-2-3-14-Spec-of-AX-Protection-Support	UPC
DE3.1.2.3.15	Specification of AXMEDIS accounting and reporting AXMEDIS-DE3-1-2-3-15-Spec-of-AX-Accounting-and-Reporting	EXITECH

1.1 This document concerns

Specification of AXMEDIS database and query support, first update of part E of DE3.1.2. The document has been reorganized and therefore some parts that were present in DE3.1.2E have been moved in other deliveralbles. The title have been maintained and under the title the reader can find the part of DE 3.1.2.2 that now contains the specification.

1.2 List of Modules or Executable Tools Specified in this document (EXITECH, UPC, FHGIGD, DSI)

A module is a component that can be or it is reused in other cases or points of the AXMEDIS framework or of other AXMEDIS based solutions.
The modules/tools have to include effective components and/or tools and also testing components and tools.

Module/tool Name	Module/Tool Description and purpose, state also in which other AXMEDIS area is used	Standards exploited if any
AXMEDIS Database Interface	This module is responsible for giving to the whole system a view of the database that is abstracted by the database that is really employed. This module can be decomposed in an abstraction layer with plugin that can have also different implementations on the basis of the database adopted.	MPEG21, Dublin Core
AXMEDIS Database Web Service Interface	This module is necessary to offer database service to modules that are not implemented in JAVA.	WSI
AXMEDIS Web Administrative Database Interface	This module will provide to the AXMEDIS administrator access to the main functionalities of AXMEDIS that are related to the database.	none
AXMEDIS Loader Saver	AXMEDIS object Loader/Saver is a Web service that is capable of getting an AXMEDIS object (in the MPEG21 compliant format plus the additional information stored inside) and putting it in the database, that is the saving function; and it is also capable, given an Object ID (AXOID) to return the AXMEDIS object in the MPEG21 compliant format, that is the loading function.	WSI, MPEG21, Dublin Core
Protection Model for AXMEDIS object Repository	This module describes the UML and relational models for representing AXMEDIS licenses, which are based on MPEG-21 REL format	UPC
History of AXMEDIS Objects	AXMEDIS has to track all version and has to be capable of having immediately available the last version of the object for indexing and for fast retrieval, and has to be capable of retrieving one of the previous version of the object, also if not indexed in the database. This module that is based on AXDB interface does this work	none
AXMEDIS Query Support	The query support is a combination of query mechanisms for retrieving content objects among the indexed information, in the AXMEDIS database (for content processing, composition, formatting, etc.), on the P2P AXEPTool, on the content collector referencing the content contained in the CMSs (Crawled Results Integrated Database).	none
User Selection Archive	This module takes care of storing the query and the selection of each user in order to have a personal archive for each user that want to store his/her queries and selection for future reuse.	EXITECH
Query User Interface Web	This module describes the user interface adopted for queries in a web environment such as for kiosks, portals, etc	EXITECH, DSI
Query User Interface C++	This module describes the query user interface developed in C++ for tool and application	EXITECH, DSI
Selection User Interface	This module describes the selection user interface developed in C++ for tool and application	DSI
Query for Production on demand	This module connects the distribution channels with the AXMEDIS query support. It retrieves client queries based on a simplified XML schema from the Distribution Server which include a client and a distribution profile, transfers them to an AXMEDIS XML query and passes them on to the AXDB Query Support. The query results are returned to the Distribution Channels.	MPEG21 DIA

Query Support for Client	This module is located on the distributor side. It is either implemented on the distribution server or it is integrated into the client application. The Query Support for the Client creates a query message in XML based on a simplified query schema. A client profile is added to such a XML query and then it is sent via the Distribution Server to the Query for Production on demand. Query results are retrieved and passed on to the Client GUI.	CC/PP, UAProf, MPEG21 DIA

1.3 List of Formats Specified in this document (EXITECH, UPC, DSI, FHGIGD)

A format can be (i) an XML content file for modeling some information, (ii) a file format for storing information, (iii) a format that is manipulated by the tools described in this document, etc...

Format Name	Format Description and purpose, state also in which other modules is used	Standards exploited if any
AXDB Mapping	Mapping between fields to be employed in a query and the database tables	XML
AXMEDIS Query	Format to express a query in the AXMEDIS system	XML
AXMEDIS Query Result	Format to express the result of a query on the AXMEDIS system	XML
AXMEDIS Selection	Format to express an AXMEDIS selction that is a set of AXOID and queries	XML
Simple Query	This format provides a simplified query schema that allows the distribution channels to place queries into the AXMEDIS database in a convenient and easy way .	XML
Distribution Profile	Format to express general information about the distribution channel, e.g. bandwidth.	XML

1.4 List of Databases Specified in this document (EXITECH, UPC, DSI, FHGIGD)

Database Name	database Description and purpose, state also in which other AXMEDIS area is using	Standards exploited if any
AXDB	Main AXMEIDS database for storing objects in the AXMEDIS format for querying purposes and future retrieval	none
PAR and License	Relational database for storing licenses associated to users and objects and Potential Available Rights associated to the AXMEDIS objects	UPC

1.5 List of Protocols Specified in this document

Protocol Name	protocol Description and purpose, state also in which other modules is used	Who is the master and who is the slave	Standards exploited if any
SAVER WEBSERVICE	Web Service for saving and indexing AXMEDIS files	Master is the web service, slave or clients are the application of other modules of the framework	MPEG21, DublinCore
COMMITLISTENER WEBSERVICE	Web Service to be implemented by whom has to use Saver Async methods	Master is the web service, slave or client is the Saver WS	none
LOADER WEBSERVICE	Web Service for retrieving AXMEDIS objects from AXDB	Master is the web service, slave or clients are the application of other modules of the framework	none
CHECKOUTLISTENER WEB SERVICE	Web Service to be implemented by whom has to use Loader Async methods	Master is the web service, slave or client is the Loader WS	none
PUBLICATION_SUPPORT WEB SERVICE	Web Services that returns the AXOID of the objects modified after a certain date	Master is the web service, slave or clients are the application of other modules of the framework	none
USER _SUPPORT WEB SERVICE	Web Service that allow the verification and authentication of factory users	Master is the web service, slave or clients are the application of other modules of the framework	none

QUERY_SUPPORT WEB SERVICE	Web Service that allows to issue queries and to have back resposnes	Master is the web service, slave or clients are the application of other modules of the framework	none
QUERY SUPPORT LISTENER WEB SERVICE	Listener that receives query responses in async manner	Master is the web service, slave is the Query Support	none
SELECTION ARCHIVE WEB SERVICE	Web Service that allows the management of the selection archive and the actualization of the selections	Master is the web service, slave or clients are the application of other modules of the framework	none
ACTUALIZE LISTENER	Listener that receives actualized selections in async manner	Master is the web service, slave is the selection archive web service	none
LOCKING WEB SERVICE	Web Service for locking unlocking objects in the database in order to avoid contemporary editing	Master is the web service, slave or clients are the application of other modules of the framework	none

2 General Use Cases and scenarios (EXITECH, UPC, DSI, FHGIGD)

- AXMEDIS Query Support Scenario
    - SCENARIO 6: Content accessibility, querying
    - SCENARIO 6.1: Making query on the several direction. Technical Querying, Collecting and integrating results.
    - SCENARIO 6.2: Producing a selection.
    - SCENARIO 6.5: Store a query/Selection and make it active later
    - SCENARIO 6.6: Make a query on a Kiosk (EXITECH, DSI, ILABS)

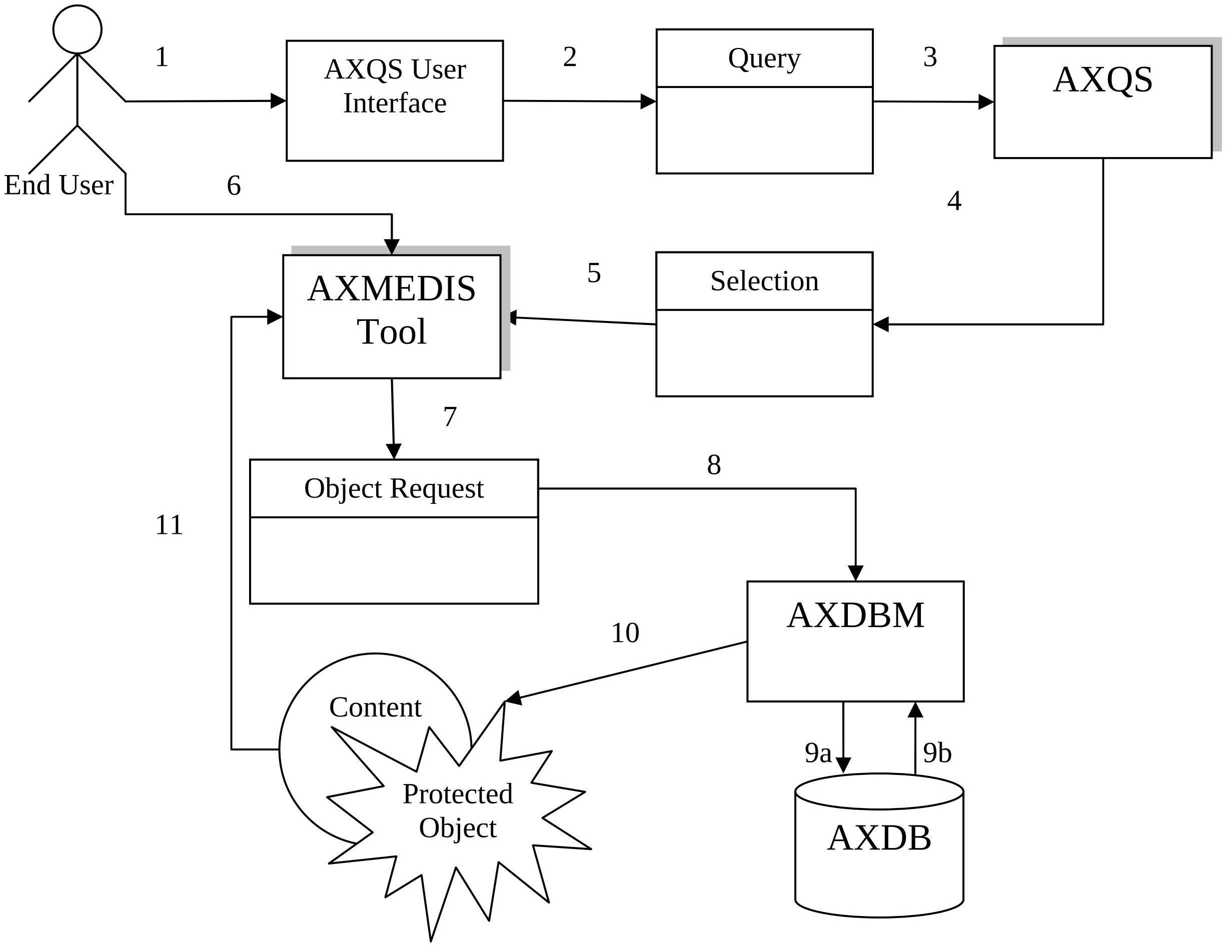
2.1 AXMEDIS Query Support Scenario

The main functionality that have been identified in the system that are also related to the query support are depicted in the scenario reported below.
The main scenario that have been identified are:

- SCENARIO 6: Content accessibility, querying
- SCENARIO 6.1: Making query on the several direction. Technical Querying, Collecting and integrating results.
- SCENARIO 6.2: Producing a selection.
- SCENARIO 6.5: Store a query/Selection and make it active later

For each scenario a brief description is reported.

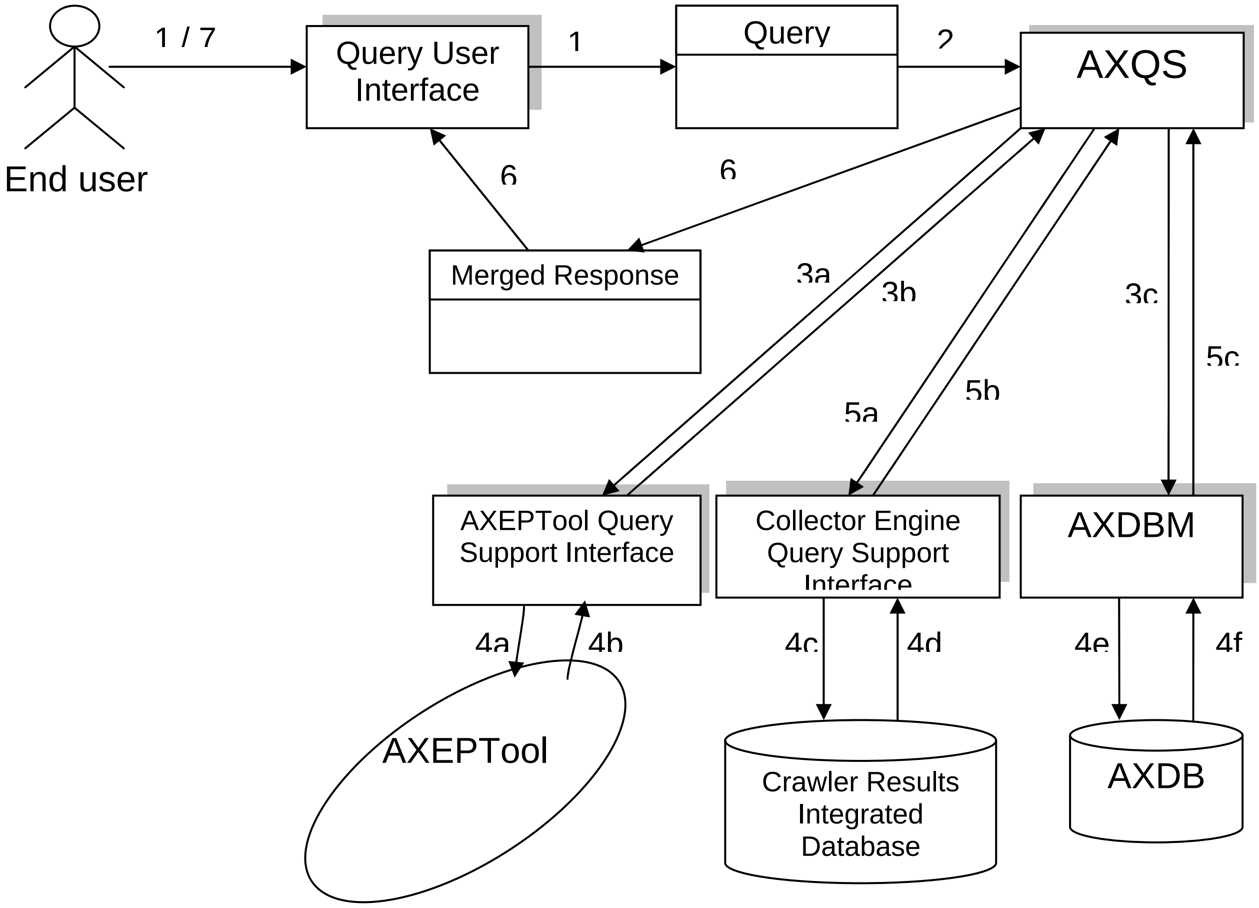
2.1.1 SCENARIO 6: Content accessibility, querying



1. The End-User creates a query on the user interface

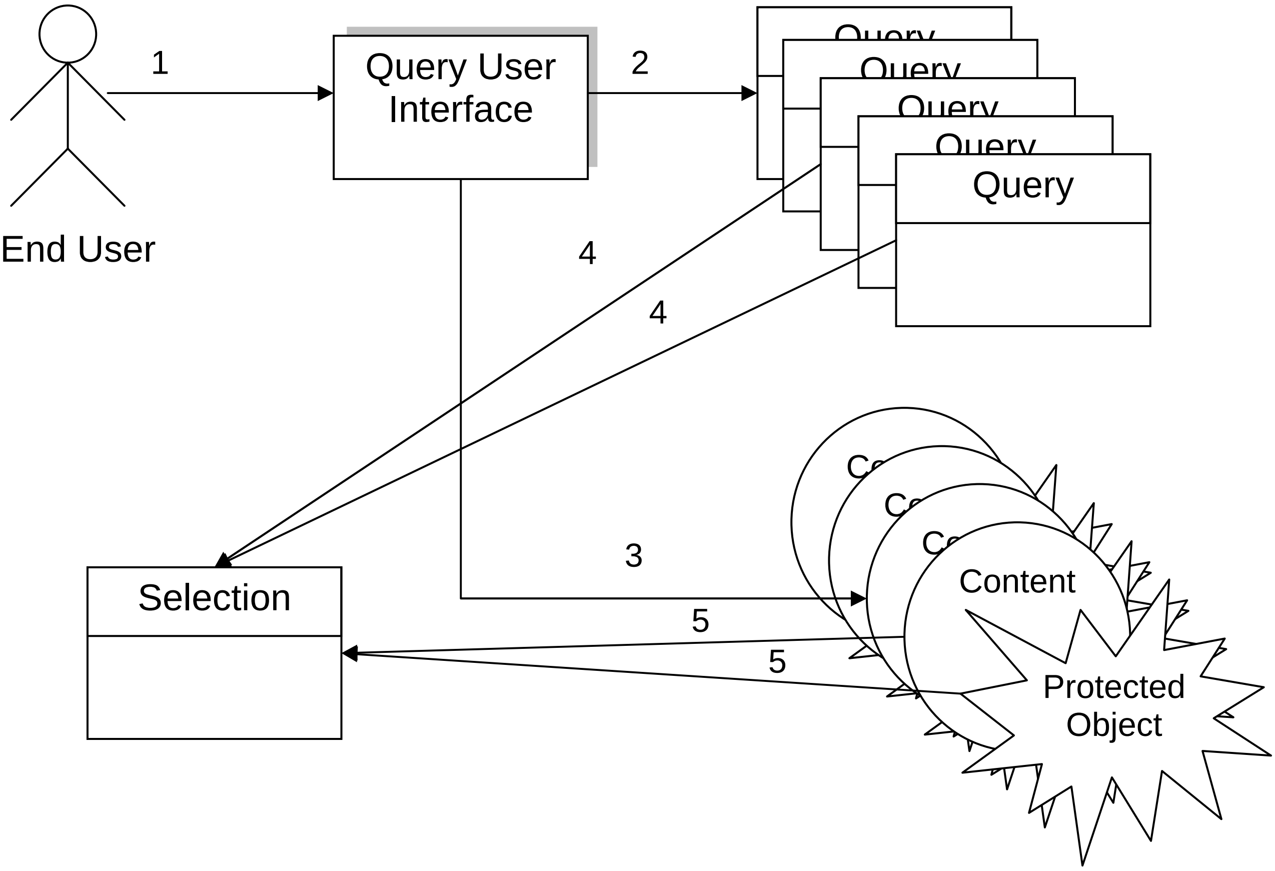
2., 3. The Query is send to the AXQS by the user interface
4. The AXQS returns a Selection (see Scenario 6.1)
5. The selection is processed by an AXMEDIS tool
6. The End-User selects one of more objects in the selection
7., 8. The AXMEDIS Tool issues a request for the set of objects to the AXDBM
9., 10. The AXDB returns the set of requested objects
11. The objects are ready to be used in the AXMEDIS Tool

2.1.2 SCENARIO 6.1: Making query on the several direction. Technical Querying, Collecting and integrating results.


End user
1 / 7
Query User Interface
1
Query
2
AXQS
6
6
Merged Response
3a
3b
3c
5a
5b
5c
AXEPTool Query Support Interface
Collector Engine Query Support Interface
AXDBM
4a
4b
4c
4d
4e
4f
AXEPTool
Crawler Results Integrated Database
AXDB

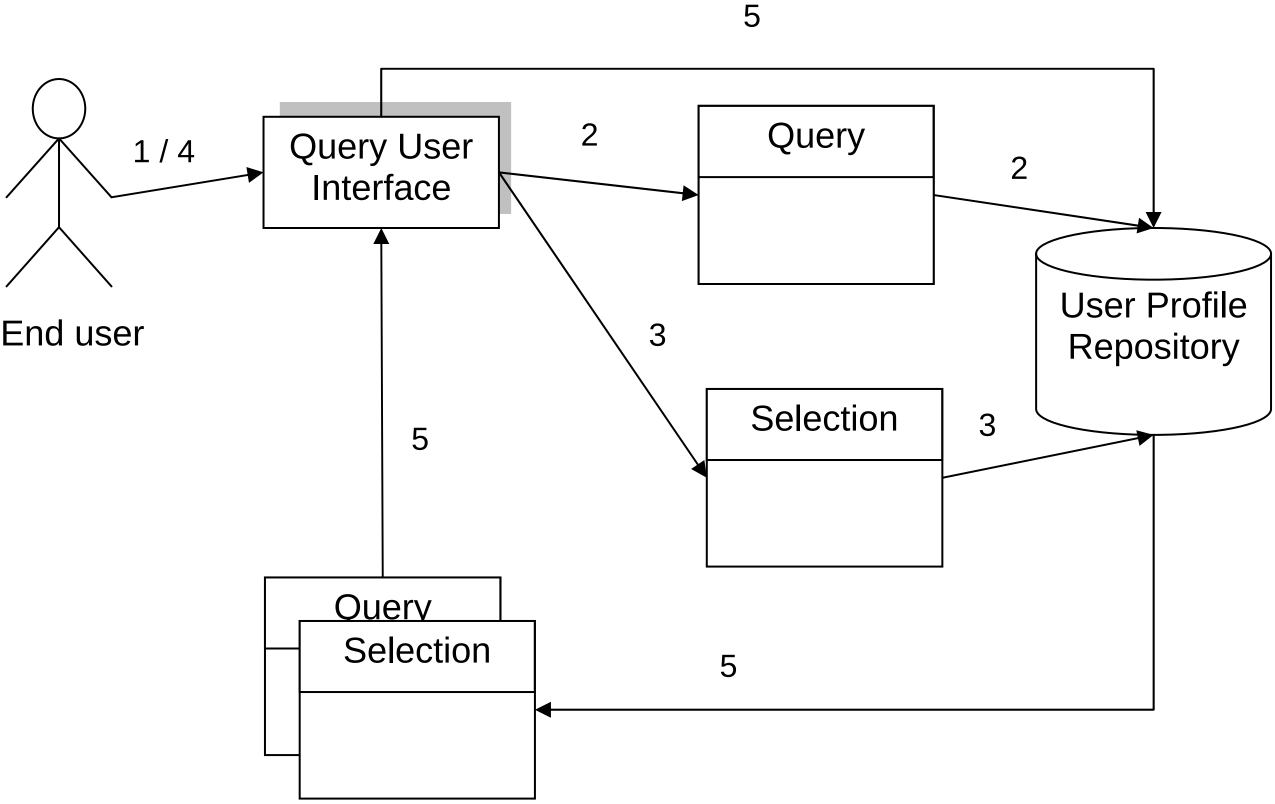
1. The End user, using the AXQS User Interface, composes a Query on aspects of interest (technical, DRM or feature related). Furthermore, the Actor chooses "where" to search for available AXMEDIS objects: within local AXMEDIS Database (and within AXMEDIS objects contained within the local AXDB), on AXEPTool network or among those contents which have to be collected, by the Collector Engine, and have not yet.
2. The End User submits the queries previously composed
3. AXQS submits the Actor's query to each of the chosen search "places" by using the corresponding specific interface: (i) Collector Engine Query Support Interface, (ii) AXEPTool Query Support Interface and (iii) AXMEDIS Database Manager
4. Each query interface (see step 3) looks for the required features in the corresponding domain
5. AXQS collects all the responses from the query interfaces
6. AXQS merges the results all together and return the complete list to the AXQS User Interface
7. AXQS User Interface shows the result to the Actor in an adequate manner, i.e. in such a way that the Actor can understand: (i) from which source an object come (ii) which are the restriction on the object (iii) etc…

2.1.3 SCENARIO 6.2: Producing a selection.



1., 2. End user can access through the AXQS User Interface to a collection of queries (previously issued)
3. End User can access also to a collection of Object (reference to object in the case of AXQS user interface)
4., 5. The set of an arbitrary number (0 or more) of queries and of an arbitrary number of reference to objects (0 or more) are put together to form a Selection

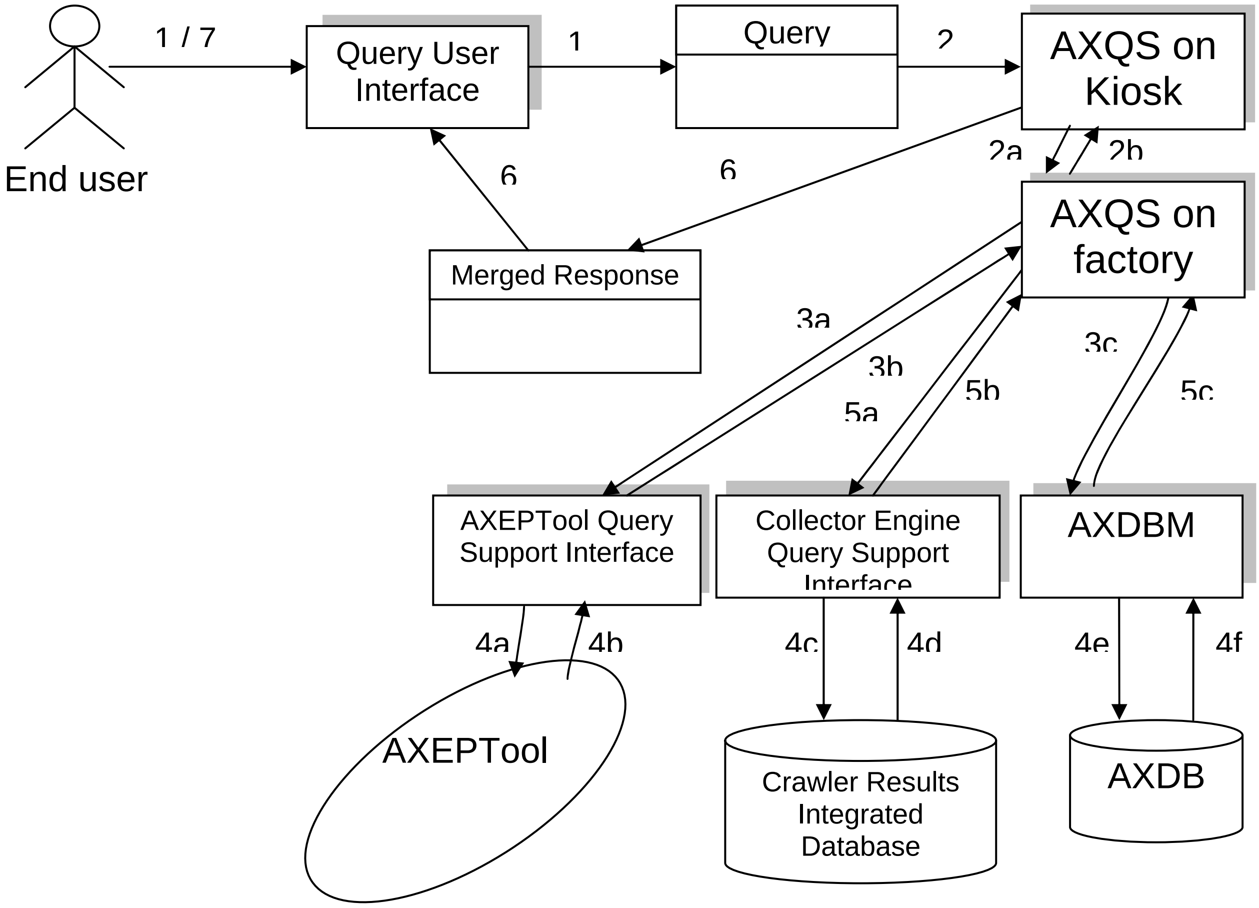
2.1.4 SCENARIO 6.5: Store a query/Selection and make it active later



1. End User uses AXQS user interface to interact with his/her personal profile
2. End User can save in his/her user profile some Queries
3. End User can save in his/her user profile some Selections
4. End User can recall a Selection/Query from user profile
5. Query or selection is returned to the User Interface for activation/modification

2.1.5 SCENARIO 6.6: Make a query on a Kiosk (EXITECH, DSI, ILABS)

This scenario allows to query the local Kiosk QS or to forward the query to the remote QS that is in the factory that in turn can forward the query to P2P, Factory AXDB and Factory crawling mechanism.



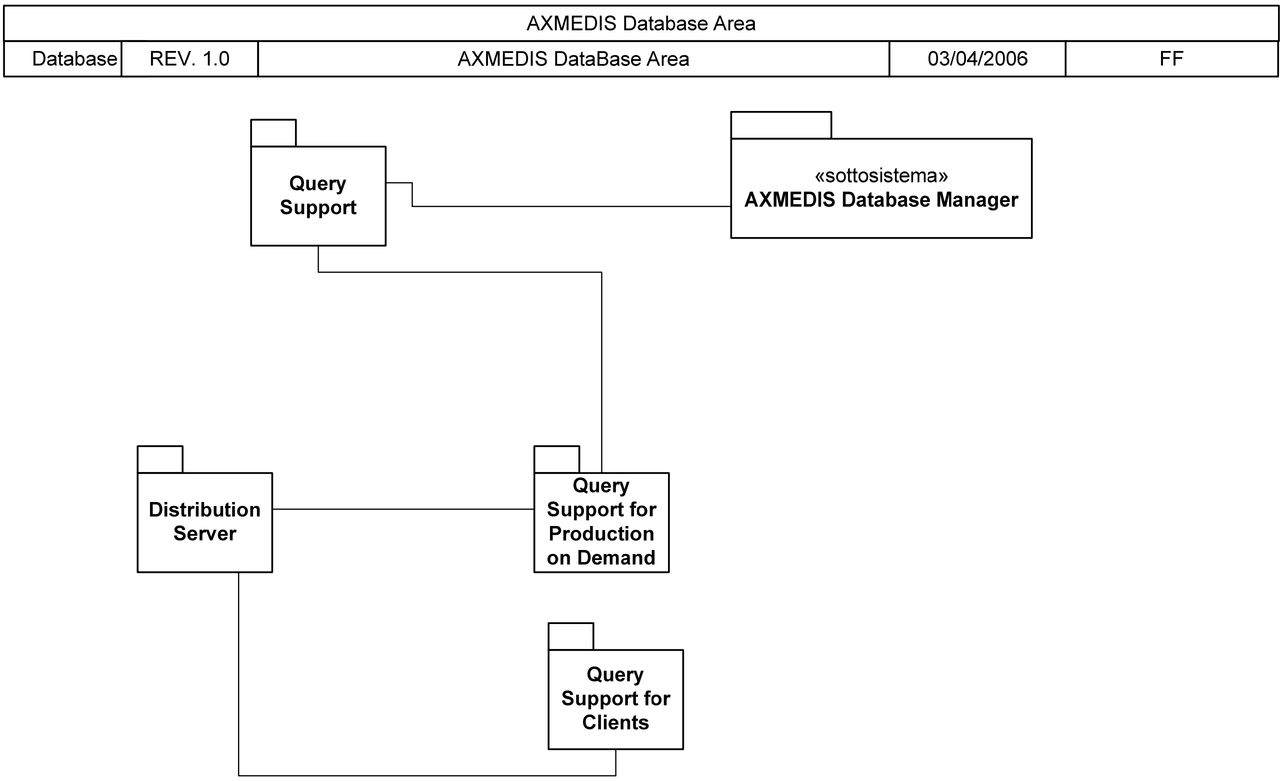
1. The End user, using the AXQS User Interface, composes a Query on aspects of interest (technical, DRM or feature related). Furthermore, the Actor chooses "where" to search for available AXMEDIS objects: within local AXMEDIS Database (and within AXMEDIS objects contained within the local AXDB), on AXEPTool network or among those contents which have to be collected, by the Collector Engine, and have not yet.
2. The End User submits the queries previously composed on the Kiosk
2a,b The query support of the Kiosk can forward the query to the Factory QS that in turn forwards it to the other information sources.
3. AXQS submits the Actor's query to each of the chosen search "places" by using the corresponding specific interface: (i) Collector Engine Query Support Interface, (ii) AXEPTool Query Support Interface and (iii) AXMEDIS Database Manager
4. Each query interface (see step 3) looks for the required features in the corresponding domain
5. AXQS collects all the responses from the query interfaces
6. AXQS merges the results all together and return the complete list to the AXQS User Interface
7. AXQS User Interface shows the result to the Actor in an adequate manner, i.e. in such a way that the Actor can understand: (i) from which source an object come (ii) which are the restriction on the object (iii) etc…

3 General architecture and relationships among the modules produced (EXITECH, UPC, DSI, FHGIGD)

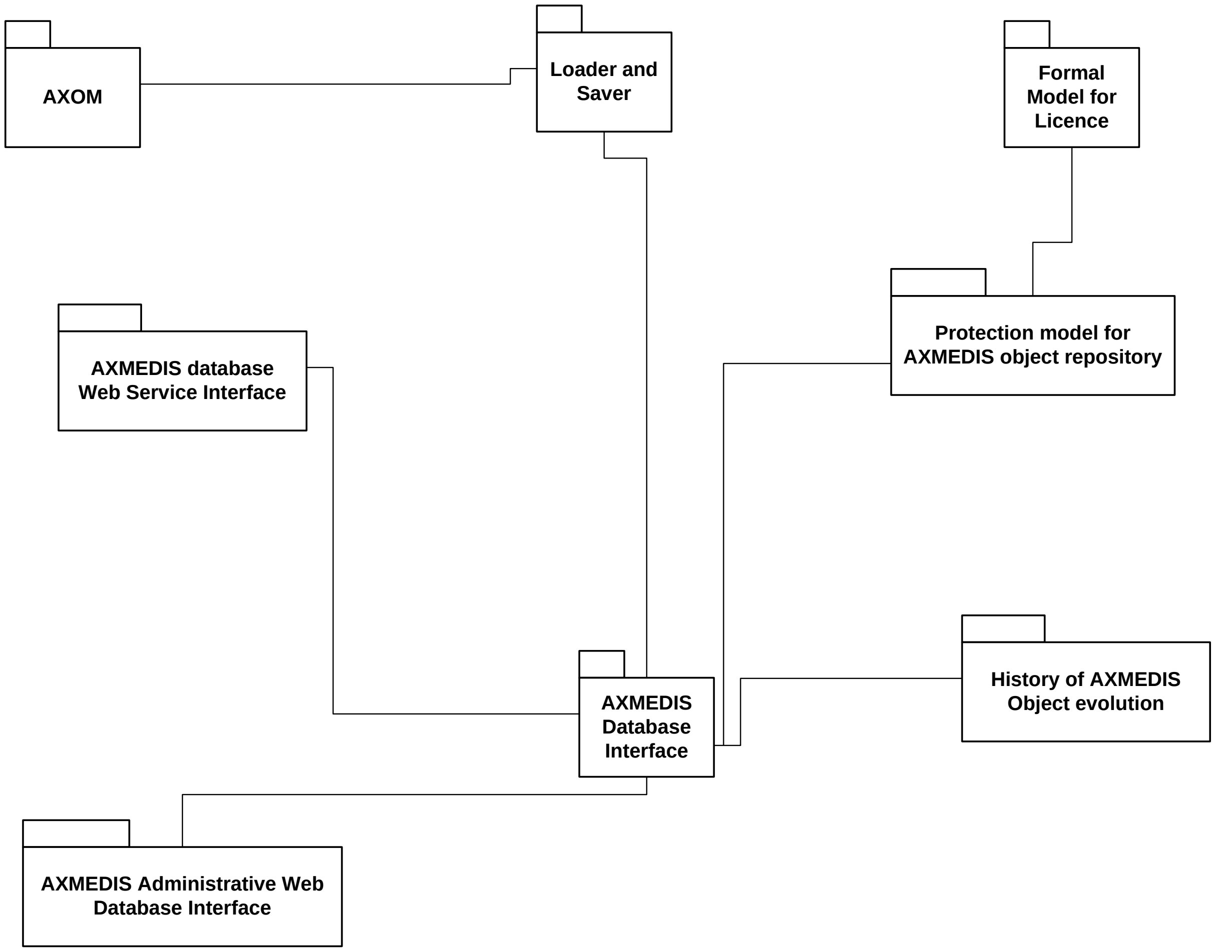
The AXMEDIS database area includes:

- **AXMEDIS Query Support**
- **AXMEDIS Query User Interface (both web and C++)**
- **AXMEDIS Selection Interface**
- **AXMEDIS Database Manager that in turn contains:**
    - **AXMEDIS loader Saver**
    - **AXMEDIS DataBase Interface**
    - **AXMEDIS Web Service DataBase Interface**
    - **AXMEDIS Web Administrative Interface**
    - **Protection Model for AXMEDIS object repository**
    - **History of AXMEDIS object evolution**
- **User selection Archive**
- **Query Support for production on demand**
- **Query Support for clients**

The relationships among the modules are expressed in the following diagram



AXMEDIS Database Manager				
Database	REV. 1.0	AXMEDIS DataBase Manager	03/04/2006	FF


AXOM
Loader and Saver
Formal Model for Licence
AXMEDIS database Web Service Interface
Protection model for AXMEDIS object repository
History of AXMEDIS Object evolution
AXMEDIS Database Interface
AXMEDIS Administrative Web Database Interface

4 AXMEDIS Database Interface (EXITECH)

Module/Tool Profile		
AXMEDIS Database Interface		
Responsible Name	Fioravanti	
Responsible Partner	Exitech	
Status (proposed/approved)	Approved	
Implemented/not implemented	Implemented	
Status of the implementation	Completed, under maintenance	
Executable or Library/module (Support)		
Single Thread or Multithread	Multi thread	
Language of Development		
Platforms supported		
Reference to the AXFW location of the source code demonstrator	No demonstrator provided for this back-office activity	
Reference to the AXFW location of the demonstrator executable tool for internal download	No demonstrator provided for this back-office activity	
Reference to the AXFW location of the demonstrator executable tool for public download	No demonstrator provided for this back-office activity	
Address for accessing to WebServices if any, add accession information (user and Passwd) if any	No web service provided	
Test cases (present/absent)	absent	
Test cases location	---	
Usage of the AXMEDIS configuration manager (yes/no)	no	
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved	-- --	
Major pending requirements	-- --	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section

Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

4.1 General Description of the Module

This module is responsible for giving to the whole system a view of the database that is abstracted by the database that is really employed. This module can be decomposed in an abstraction layer with plugin that can have also different implementations on the basis of the database adopted. One of the plugin that can change in the full text engine that can be different from system to system.

In this section it is important to define the possible interactions between other parts of the system and the database on the basis of the requested functionalities in the user requirements. In any we have two choices:

- Limit the AXMEDIS Database Interface to a very simple interface capable of executing SQL commands, translating and optimizing them for the specified engine demanding to the external tools the real functionalities toward the other parts of AXMEDIS;
- Create a very complex interface with all the functionalities inside this module having in external modules only the stubs for calling Database Interface;

Since it is very difficult to organize from the beginning all the functionalities offered by the database it is preferable to have a simple interface of the DB that allow to execute simple commands that can be used by the other modules that are part of the real High level interface to the DB.

The basic functionalities of this module are used by:

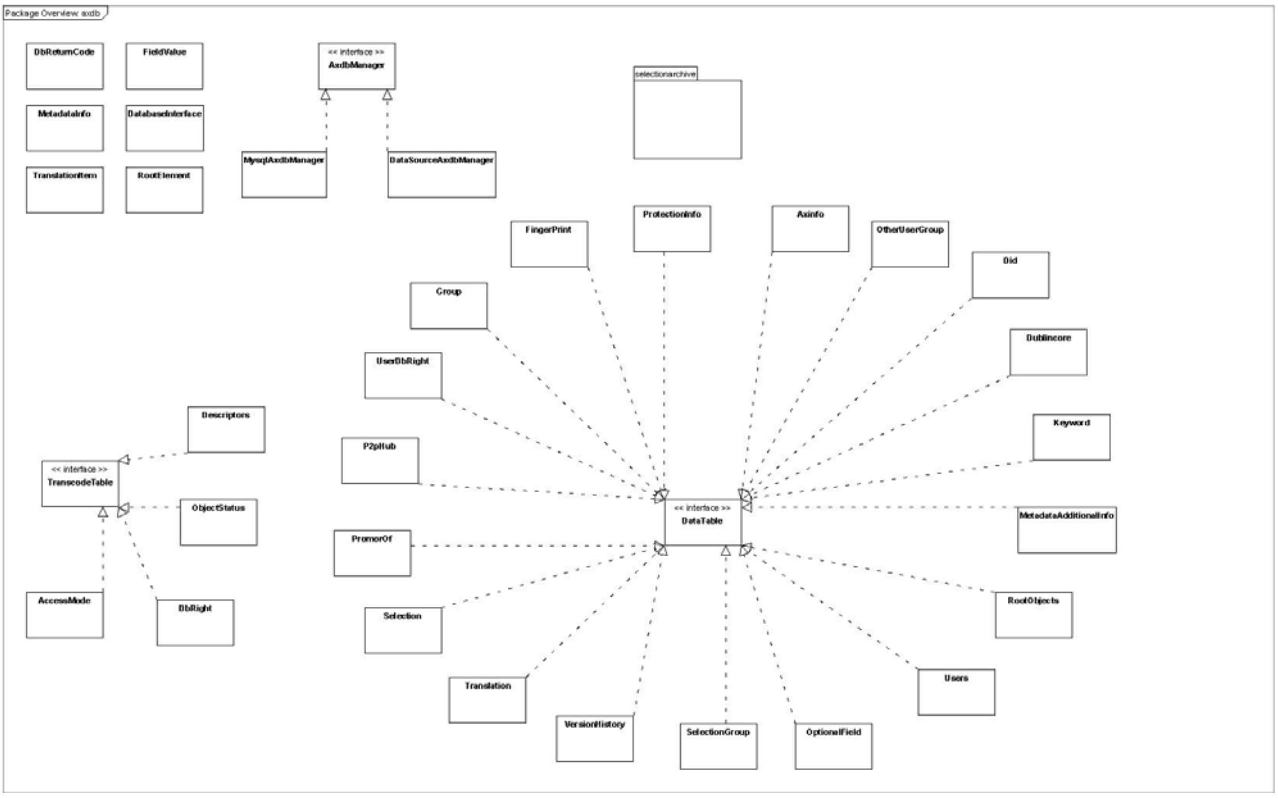
- AXMEDIS Object Loader/Saver: this module is responsible for inseriting, exporting and change versions of AXMEDIS objects;
- Core Accounting Manager and reporting tool: this module is responsible for the storing and retrieving of Action-Logs;
- Query Support Web Service Interface: this service is responsible for all kind of queries;

- Evolution in production Repository: this module is responsible for advanced management of versioning not covered by the Object Loader/Saver, such as analysis of changes in contents, different version of AXINFO and so on;
- Query and Selection Archive for Each User: for getting and saving Selections (and therefore also queries in the personal archive);

Starting from this point of view it can be understand easily that the Database Interface has to provide all back end services for interfacing optimally with the selected database engine and have to expose some simple methods for executing basic I/O and select operation of the DB.

4.2 Module Design in terms of Classes

Since this module is a Java package, it is necessary to draw the preliminary class diagram over which, it is possible to define the public methods exposed by the module at least in terms of interfaces.
A simplified class diagram for this module is reported below.



4.3 User interface description

This module has no user interface.

4.4 Technical and Installation information

This module is provided as a jar and therefore you need only to use it in your JAVA projects

4.5 Draft User Manual

No user manual apart from API specification provided in the repository and referred in section 17

4.6 Examples of usage

You can find examples of usages in all the modules that use this module or in the test source code provided with the module.

4.7 Integration and compilation issues

The JAR can be used in all JDK 1.5 environment and for compiling it you can use directly the project provided for Netbeans 5.5

4.8 Configuration Parameters

Config parameter	Possible values
axdbUrl	jdbc:mysql://localhost/axdb-test?jdbcCompliantTruncation=false
axdbUser	axdbuser
axdbPwd	mkzamk

4.9 Errors reported and that may occur

AXINFO_ERROR Error code related to Axinfo Tables
CORRUPTED_VERSION Error code related to DatabaseInterface class
DATABASE_ERROR Represents a database error (missed connection and so on
DCMI_ERROR Error code related to Dcmi Tables
DID_CORRUPTED Error code related to DatabaseInterface class
DID_ERROR Error code related to Did Tables
FP_ERROR Error code related to FingerPrint Tables
INVALID_ARG Error raised when an argument of a function is not valid
INVALID_ARG1 Error raised when an argument of a function is not valid
INVALID_ARG2 Error raised when an argument of a function is not valid
INVALID_ARG3 Error raised when an argument of a function is not valid
KEY_EXISTS Error raised when an insert with duplicated primary key is performed
KEY_NOT_EXISTS Error raised when an key is not found in a table where it is expected to be
LOCKED_BY_OTHER Error code related to an object that has been already unlocked by someone
METADATA_ERROR

Error code related to Medatata clean up
METADATAINFO_ERROR Error code related to MetadtaaAdditionalInfo Tables
MISSING_MANDATORY_FIELD1 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD10 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD11 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD12 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD13 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD14 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD15 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD2 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD3 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD4 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD5 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD6 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD7 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD8 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)
MISSING_MANDATORY_FIELD9 return code issued when a missing mandatory field is present (see the comment in the class raining the code for details on field)

NOT_ALL_DELETED Error code that means that not all the planned delete have been performed
NOT_ALL_INSERTED Error code that means that not all the planned insert have been performed
NOT_AUTHORIZED Error code related to user not authorized to operation
OBJECT_ALREADY_LOCKED Error code related to an object that has been already locked by someone
OBJECT_ALREADY_UNLOCKED Error code related to an object that has been already unlocked by someone
OBJECT_AVAILABLE Error code related to DatabaseInterface class
OBJECT_CORRUPTED Error code related to DatabaseInterface class
OBJECT_INVALID_URI Error code related to DatabaseInterface class
OBJECT_NOT_EXISTS Error code related to DatabaseInterface class
OBJECT_UNAVAILABLE Error code related to DatabaseInterface class
OF_ERROR Error code related to OptionalField Tables
OK return code for operations performed without problems
PASSWORD_INCORRECT Error code related to user password not correct
PROMOROF_ERROR Error code related to PromorOf Tables
TRANSCODE_TABLE_DB_ERROR Error code raised by the transcode tables that cannot find the requested Id
TRANSLATION_ERROR Error code related to PromorOf Tables
UNKNOWN_ERROR Represents an unknown db error
USER_NOT_PRESENT Error code related to user not present
VERSION_NOT_FOUND Error code related to loader not found object_version
VH_ERROR Error code related to VersionHistory Tables
WS_ERROR Error code related to AXDB WS not identified error

5 AXMEDIS Database Web Service Interface (EXITECH)

Module/Tool Profile		
AXMEDIS Database Web Service Interface		
Responsible Name	Fioravanti	
Responsible Partner	EXITECH	
Status (proposed/approved)	approved	
Implemented/not implemented	implemented	
Status of the implementation	under improvement	
Executable or Library/module (Support)		
Single Thread or Multithread	multithread	
Language of Development	JAVA	
Platforms supported	All supported by JDK 1.5	
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/WebServices/AXDBSupportWS/source/	
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/repos/WebServices/AXDBSupportWS/bin/Tomcat5.5/	
Reference to the AXFW location of the demonstrator executable tool for public download		
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	Endpoint: http://81.73.104.125:8080/AxdbWS/publication WSDL: http://81.73.104.125:8080/AxdbWS/publication?WSDL Endpoint: http://81.73.104.125:8080/AxdbWS/user WSDL: http://81.73.104.125:8080/AxdbWS/user?WSDL	
Test cases (present/absent)	absent	
Test cases location	none	
Usage of the AXMEDIS configuration manager (yes/no)	no	
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved	-- --	
Major pending requirements	-- --	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section

Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

5.1 General Description of the Module

The AXMEDIS Database Interface is comprised of two main modules that are the Database interface (already described in the previous section) and the Web Service database Interface. While the first module can be a JAVA package, the second module is necessary to offer database service to modules that are not implemented in JAVA. To this end part of the functionalities or more advanced functionalities can be put in the WebService part of the module to allow a better integration at the system level.

AXMEDIS Database WebService Interface is a critical module since allow via different methods of several web services to access to the functionalities that are provided by the database. The list of functionalities will be improved over the time as soon as other engine or tools will require new services.
For the moment the following Webservices are defined:

- Publication_support that will provide the services needed by the publication engine to recover the AXOID that have been inserted/update after a certain date/time;
- User_support will be provided to verify that if a user is authenticated for a specific operation to be performed;

In the following the definition of the web services with the related methods will be detailed, creating a subsection for each web service and inside the subsection all the methods will be detailed.

5.1.1 Publication_support

This web service has been requested by the authors of P&P in order to obtain the list of files that have been updated o modified after a certain date

Publication_support	
Method	***Description***
getModifiedObject	This method returns a list of AXOID modified or inserted after a certain timestamp

5.1.2 User Support

This web service is able to return an error code for a user authentication, when a triplet of user, password and operation is provided. The operation is optional and if not present only the password against the user name will be checked.

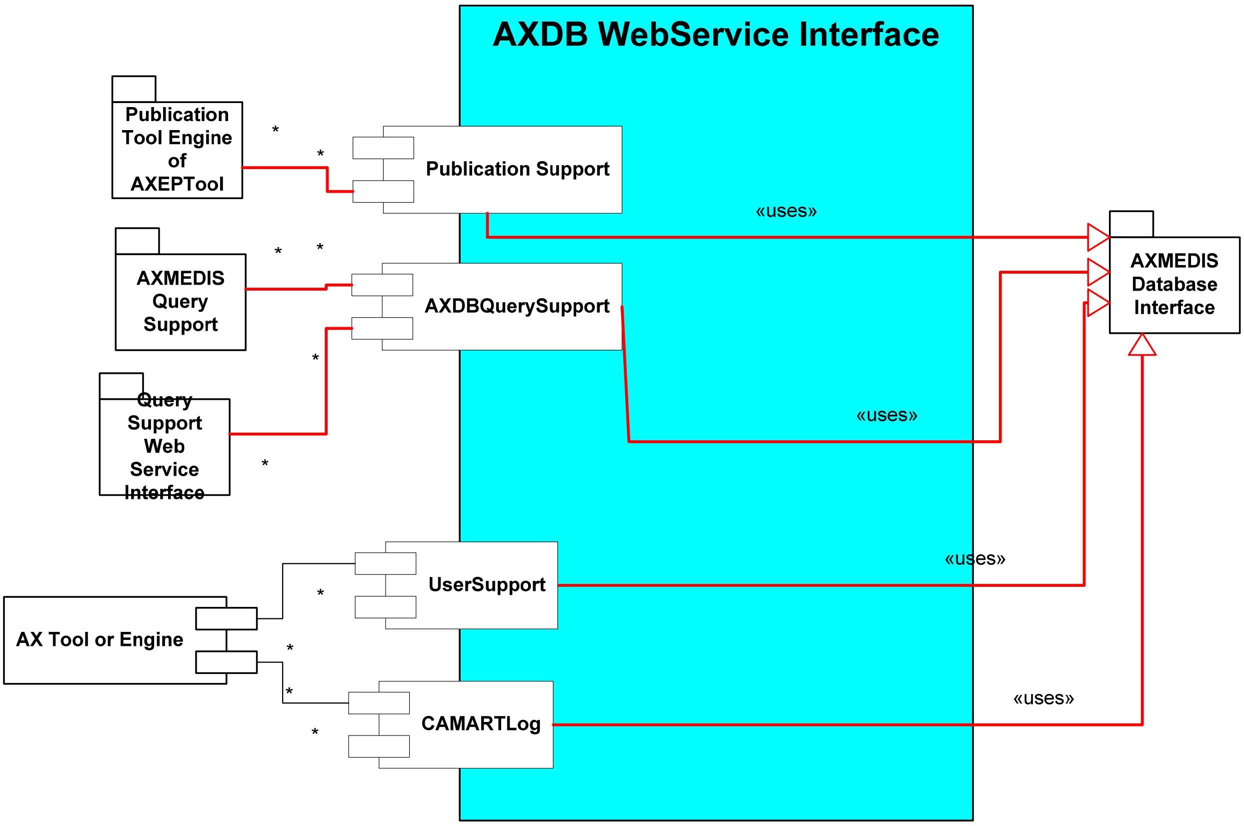
The error code returned are:

- 0: the username exists, the password is correct and the operation (if any) is allowed)
- 1: the username does not exist; no check on operation
- 2: the username exists, but the password do not match; no check on operation
- 3: the username exists, the password is correct but the user is not entitled for such operation

User_Support	
Method	***Description***
authenticateUser	This methods allows to verify if an user is authenticated or authorized to a certain operation.

5.2 Module Design in terms of Classes

For this module the following more detailed schema applies.


AXDB WebService Interface
Publication Tool Engine of AXEPTool
Publication Support
AXMEDIS Query Support
AXDBQuerySupport
Query Support Web Service Interface
UserSupport
AX Tool or Engine
CAMARTLog
AXMEDIS Database Interface
«uses»
«uses»
«uses»
«uses»
*
*

5.3 User interface description

No user interface is present for this module since it is comprised of back office web services.

5.4 Technical and Installation information

In this section it is reported a draft configuration manual for installing the service as it is provided by EXITECH in the SVN repository of the project.

5.4.1 Prerequisites

In this manual is it assumed that you install all in 1 WINDOWS server and that in this server you have:

- Mysql 4.1.10 or higher but not mysql 5.x installed and working
- Tomcat 5.5.x installed and working and respond on port 8080

It is assumed also that:

- your Tomcat is installed in $TOMCAT (say C:\Programmi\Apache Software Foundation\Tomcat 5.5, or something like that);
- your local copy of the AXMEDIS SVN repository is in $SVNROOT;
- you have got from the SVN repository the following file:
  - $SVNROOT\ Framework\doc\other\axdb\Mysql-4.1.10\axdbv4-clean.sql

If you made different assumption follow the guidelines and substitute the proposed path and names with the ones you chose.

5.4.2 Installation of AXMEDIS Database (AXDB)

We do not support more in AXDBv4 the possibility to have a prefilled database provided, since several object are available in the AXMEDIS network to test it.

5.4.2.1 Installation of an empty database

In order to install an empty axdb you have to use **axdbv4-clean.sql** script after creating a database named axdbv4 on that database.
After you have to create an user named axdbuser with all privileges on axdb-test with a password mkzamk.

5.4.2.2 Verification of the installation

The AXDB contains 35 tables and the command:
SHOW TABLES;
Will show something like:

```
Microsoft Windows XP [Versione 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\fabrizio>mysql -u root -p
Enter password: *******
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 4.1.15-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use axdbv4
Database changed
mysql> show tables;
+-----------------------+
| Tables_in_axdbv4      |
+-----------------------+
| accessmode            |
| accounting            |
| actionlog             |
| aiiconfiguration      |
| aiistyle              |
| axinfo                |
| credits               |
| dbrights              |
| dcmi                  |
| descriptors           |
| did                   |
| distributionusers     |
| download              |
| fingerprint           |
| groups                |
| groupsdbrights        |
| keywords              |
| licenselog            |
| listenertable         |
| locktable             |
| metadataadditionalinfo |
| objectstatus          |
| optionalfield         |
| otherusergroup        |
| p2phub                |
| promorof              |
| protectioninfo        |
| query                 |
| rootobjects           |
| selection             |
| selectiongroups        |
| translation            |
| users                  |
| usersdbrights          |
| versionhistory         |
+------------------------+
35 rows in set (0.09 sec)

mysql>
```

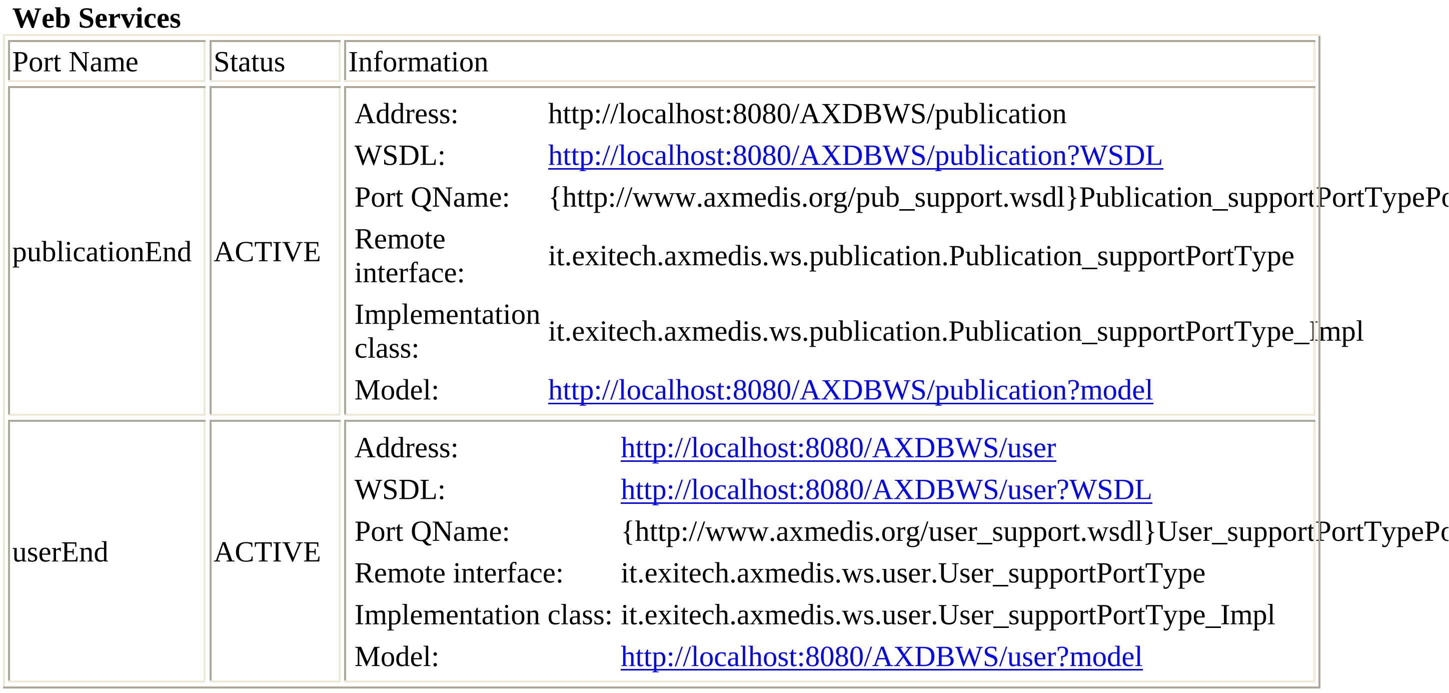
5.4.3 AXMEDIS Database Support Web Services

5.4.3.1 Installation of the WebServices

The installation of the web services identified in this section is very easy since the WAR file in $SVNROOT\WebServices/AXDBSupportWS/bin/Tomcat5.5/AXDBWS.war can be deployed as it is in Tomcat 5.5.x after installing in $TOMCAT/shared/libs the libraries in $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\Tomcat-shared-libs.zip.
If the parameters have been set as stated in section 5.4.2, nothing has to be changed, otherways it is necessary to modify the axdb.properties file to reflect the configuration of the system as stated in section 5.8.1.

5.4.3.2 Verification of the Installation

If the installation and therefore the deployment has been correctly done, in the Tomcat Manager a screen similar to the following will appear:



Web Services

Port Name	Status	Information
publicationEnd	ACTIVE	Address: http://localhost:8080/AXDBWS/publication WSDL: http://localhost:8080/AXDBWS/publication?WSDL Port QName: {http://www.axmedis.org/pub_support.wsdl}Publication_supportPortTypePo Remote interface: it.exitech.axmedis.ws.publication.Publication_supportPortType Implementation class: it.exitech.axmedis.ws.publication.Publication_supportPortType_Impl Model: http://localhost:8080/AXDBWS/publication?model
userEnd	ACTIVE	Address: http://localhost:8080/AXDBWS/user WSDL: http://localhost:8080/AXDBWS/user?WSDL Port QName: {http://www.axmedis.org/user_support.wsdl}User_supportPortTypePo Remote interface: it.exitech.axmedis.ws.user.User_supportPortType Implementation class: it.exitech.axmedis.ws.user.User_supportPortType_Impl Model: http://localhost:8080/AXDBWS/user?model

5.5 Draft User Manual

No user manual is present apart what stated in this section regarding installation and configuration

5.6 Examples of usage

Examples of usage are reported in the installation manual reported in section 5.4

5.7 Integration and compilation issues

The web services specified in this chapter are compatible with Tomcat 5.5 with JWSDP installed and with JDK 1.5 implementations.

5.8 Configuration Parameters

5.8.1 axdb.properties file

Config parameter	Possible values
axdbUrl	jdbc:mysql://mysqlhost/axdb-name?jdbcCompliantTruncation=false
axdbUser	User to be used for accessing the database (it depends on your installation of AXDB)
axdbPwd	Password of the User to be used for accessing the database (it depends on your installation of AXDB)
axdbMapping	NOT REQUESTED FOR THIS MODULE. The property file is shared with other modules that requires this parameter

6 AXMEDIS Web Administrative Database Interface (EXITECH)

Module/Tool Profile		
AXMEDIS Web Administrative Database Interface		
Responsible Name	Fioravanti	
Responsible Partner	EXITECH	
Status (proposed/approved)	Approved	
Implemented/not implemented	Implemented	
Status of the implementation	Completed, under maintenance	
Executable or Library/module (Support)		
Single Thread or Multithread	multithread	
Language of Development	JAVA	
Platforms supported	ALL supported by JAVA 1.5 and Tomcat 5.5	
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/Framework/source/axdb-administrative/	
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/repos/Framework/bin/axdb-administrative/	
Reference to the AXFW location of the demonstrator executable tool for public download		
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	Web Application http://81.73.104.125:8080/axdb-administrative/ User admin pwd Axmedis	
Test cases (present/absent)	absent	
Test cases location		
Usage of the AXMEDIS configuration manager (yes/no)	no	
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved	-- --	
Major pending requirements	-- --	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section

Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

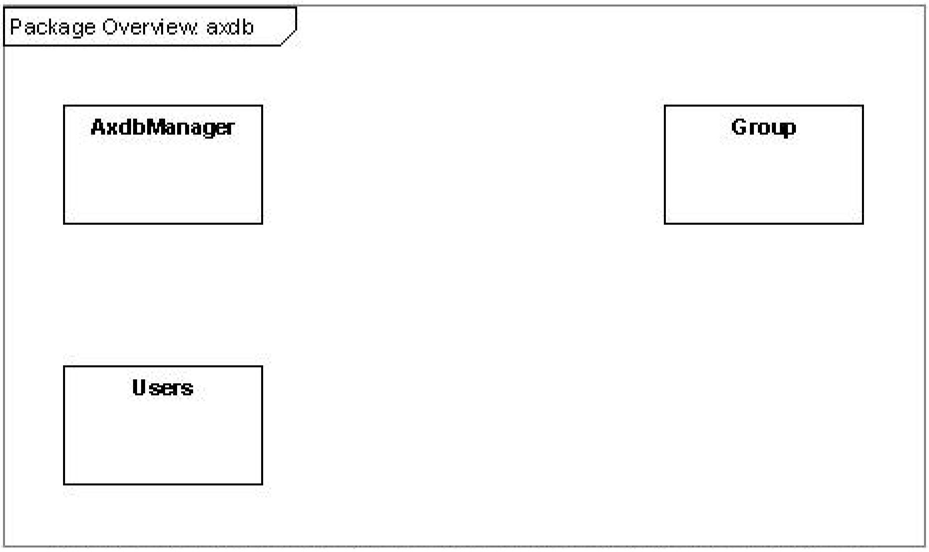
6.1 General Description of the Module

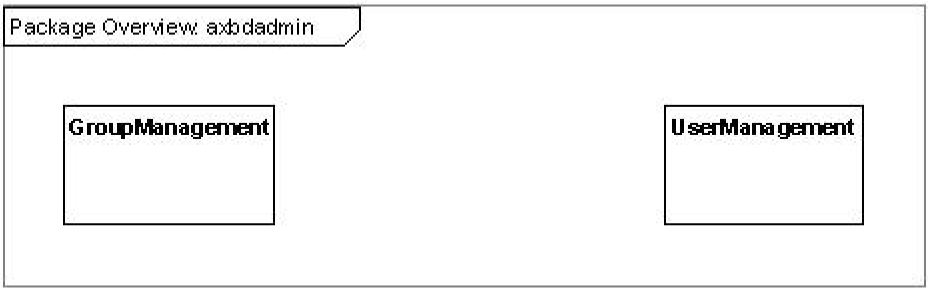
AXMEDIS Administrative Web Database Interface will provide to the AXMEDIS administrator access to the main functionalities of AXMEDIS that are related to the database. These functionalities are referred to the factory user and group and to the operations that are mainly related with the AXDB:
These functionalities will be mainly:

- Add user
- Delete User
- Update user data
- Update user rights
- Add Group
- Delete Group
- Update group data
- Assign users to group
- Lists user of the basis of the rights
- List users on the basis of the user data
- List group
- Download Object for opening
- Delete an object (this means that all the version of the object will be removed)
- Delete a specific version of an object
- Insert/Update objects
- Make query on the system (by the means of Query Support Web Service Interface), that means:
  - List object according to certain criteria

- List Action-Log on the basis of certain criteria
- Create selection/query
- Actualize selection/query
- Delete selection/query
- List Versions of the objects

6.2 Module Design in terms of Classes





6.3 User interface description

User interface is a typical web application that relay on Tomcat Application Server to be run. For more details on user interface refer to section 6.5 where in the draft manual all the details of the user interface are present.

6.4 Technical and Installation information

The WAR of the application is available on the CVS at the location https://cvs.axmedis.org/repos/Framework/bin/axdb-administrative/ and it is only needed to deploy the WAR after modifying a configuration file that is inside the WAR in the directory WEB-INF\classes and is named axdb.properties. Refer to Section 6.8 for more details of the configuration file.

References to other major components needed	AXDB is needed and therefore it is necessary to have installed somewhere in the network a MYSQL server
Problems not solved	Some minor issues on the interface has to be integrated with other tools but no problem exist at the moment
Configuration and execution context	This application requires Tomcat 5.5 to be deployed

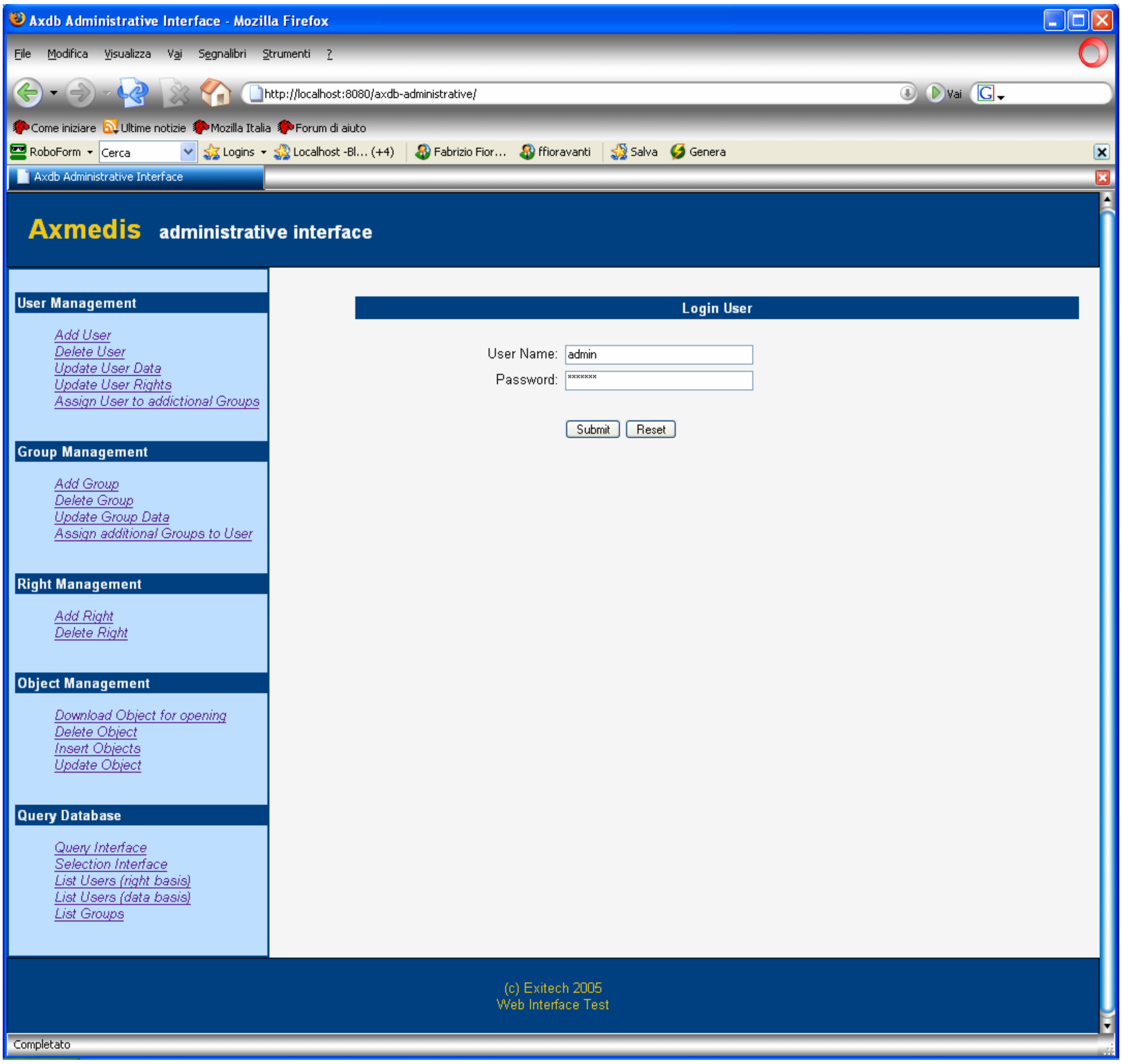
6.5 Draft User Manual

6.5.1 Login

Login interface will allow login for each user registered in the DB. Instead of the user text input it will appear the list of user and the menu on the right will also change of the basis of the user permission.

If you are not capable of login with the test or admin user provided by default with axdb, maybe it means that:

1. your database has not been started
2. your axdb is not correctly installed or jdbc connection not correctly mapped
3. your database does not contain the user adopted in the context.xml of the war



Login User

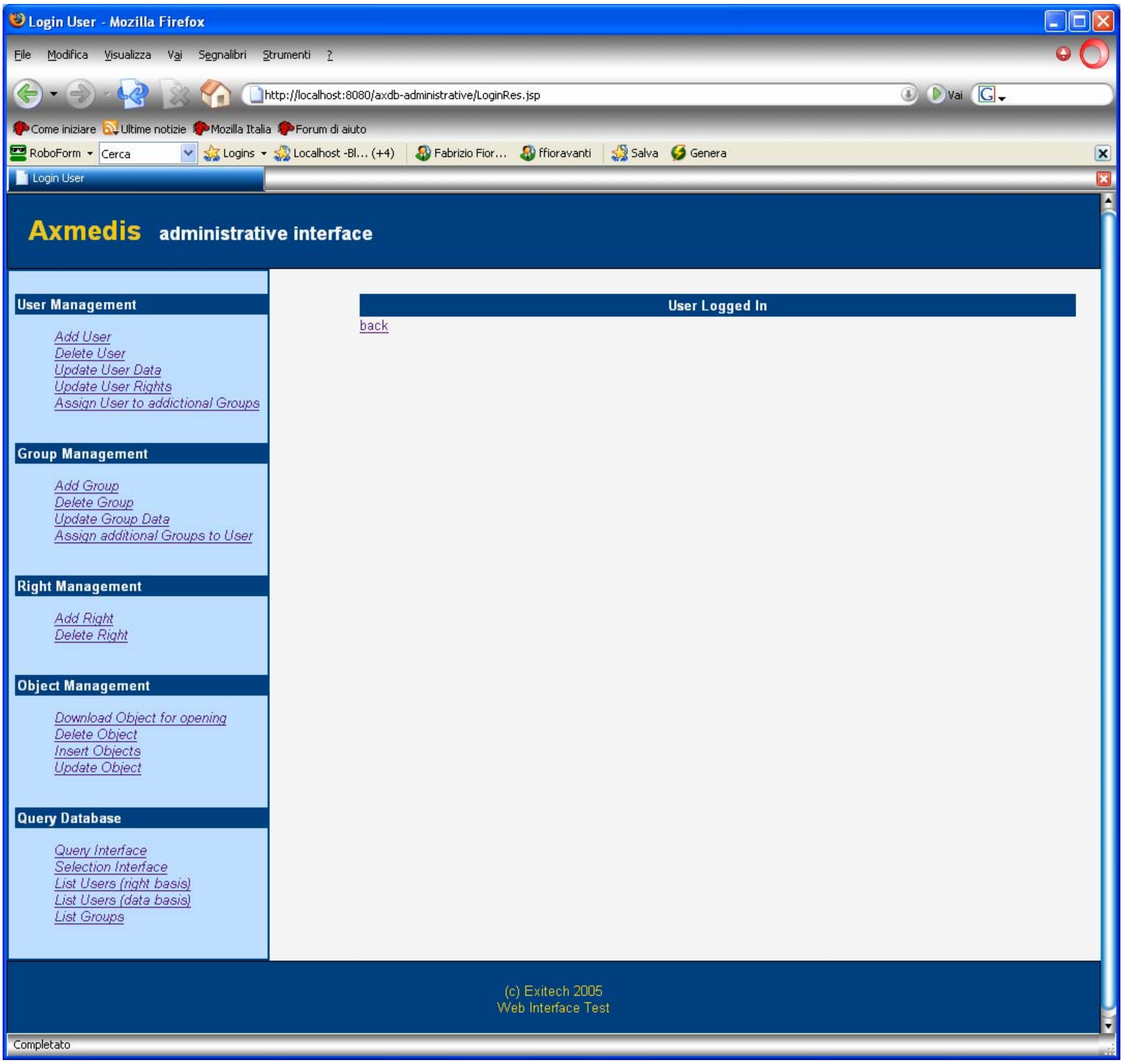
User Name: admin

Password: *******

(c) Exitech 2005
Web Interface Test

The web interface will be simple and effective having on the left side a menu bar for choosing operations and on the centre of the page the parameters for accessing to the requested functionality.

In order to increase accessibility no frame will be used.


Login User - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/LoginRes.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Login User
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
back
(c) Exitech 2005
Web Interface Test
Completato

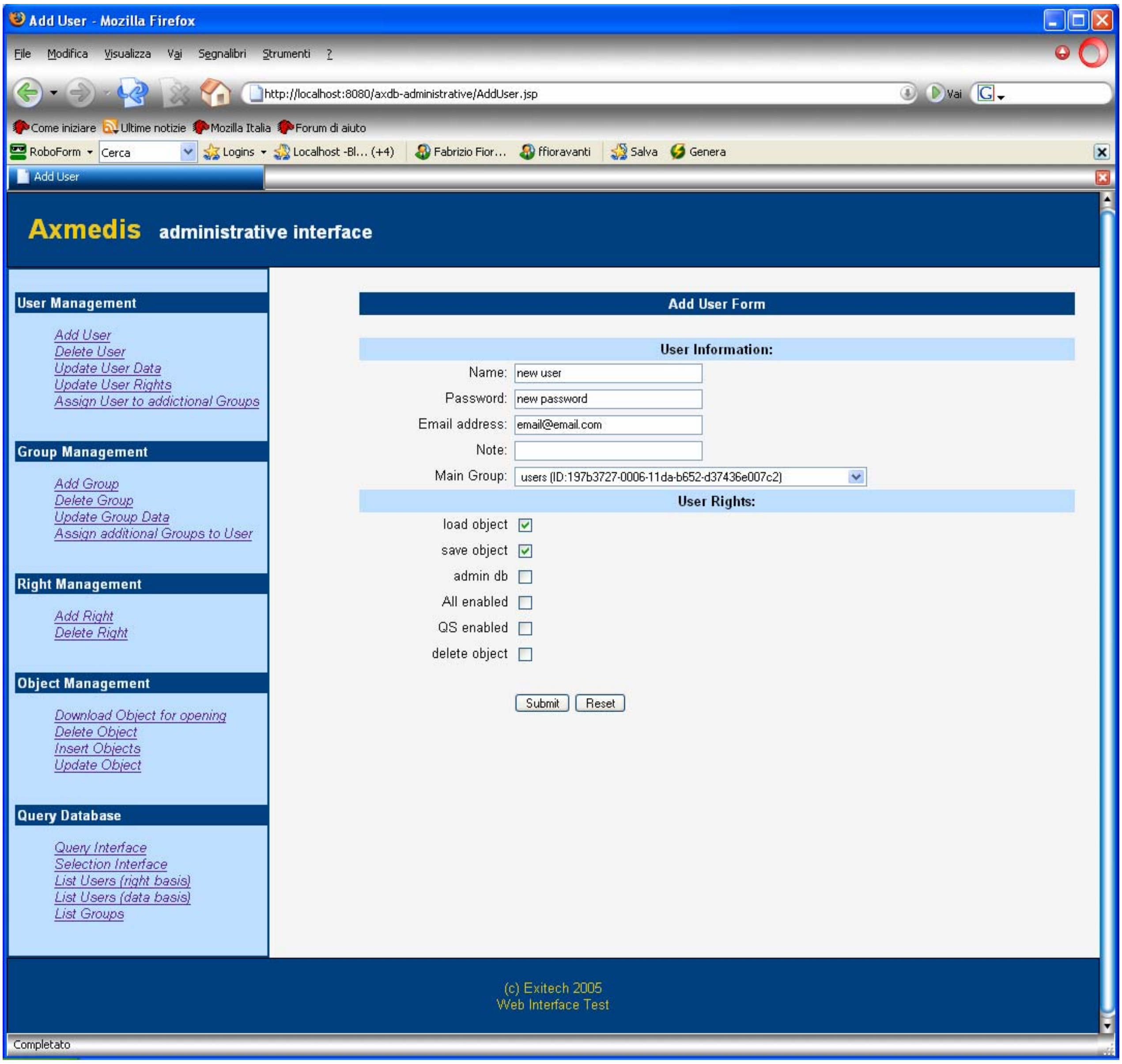
User Logged In

Each functionality will have a different mask in the centre of the page that allow to operate on the functionality.

In the following the functionality to be implemented will be listed together with some sample snapshots.

6.5.2 User Management

6.5.2.1 Add user


Add User - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/AddUser.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Add User
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
Submit
Reset
(c) Exitech 2005
Web Interface Test
Completato

Add User Form

User Information:

Name: new user

Password: new password

Email address: email@email.com

Note:

Main Group: users (ID:197b3727-0006-11da-b652-d37436e007c2)

User Rights:

load object ☑

save object ☑

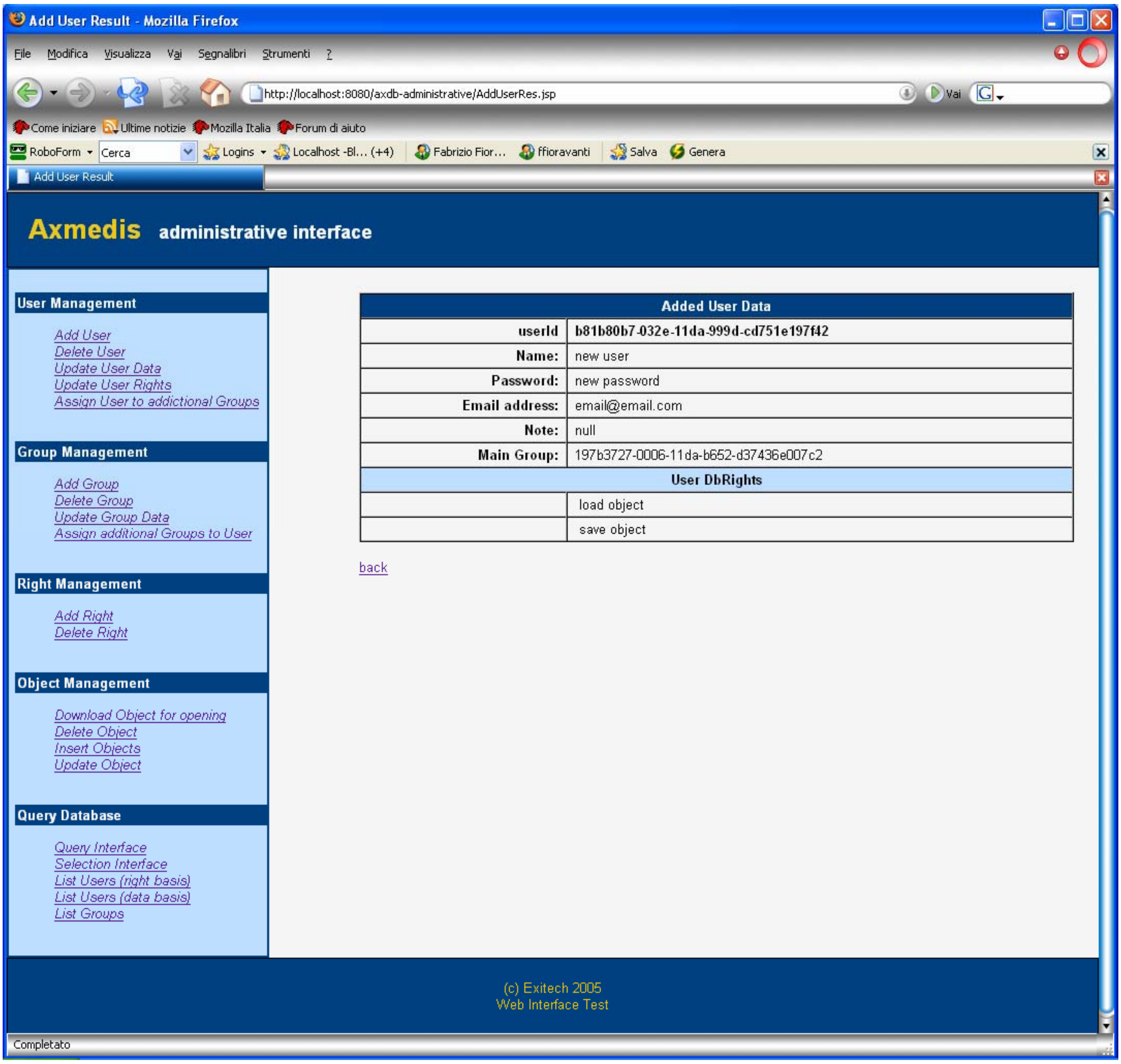
admin db □

All enabled □

QS enabled □

delete object □

The submission allows to show the inserted parameters and the generated User ID.



Add User Result - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/AddUserRes.jsp
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Add User Result
Completato

Axmedis administrative interface

User Management

- Add User
- Delete User
- Update User Data
- Update User Rights
- Assign User to addictional Groups

Group Management

- Add Group
- Delete Group
- Update Group Data
- Assign additional Groups to User

Right Management

- Add Right
- Delete Right

Object Management

- Download Object for opening
- Delete Object
- Insert Objects
- Update Object

Query Database

- Query Interface
- Selection Interface
- List Users (right basis)
- List Users (data basis)
- List Groups

Added User Data	
userId	**b81b80b7-032e-11da-999d-cd751e197f42**
Name:	new user
Password:	new password
Email address:	email@email.com
Note:	null
Main Group:	197b3727-0006-11da-b652-d37436e007c2
User DbRights	
	load object
	save object

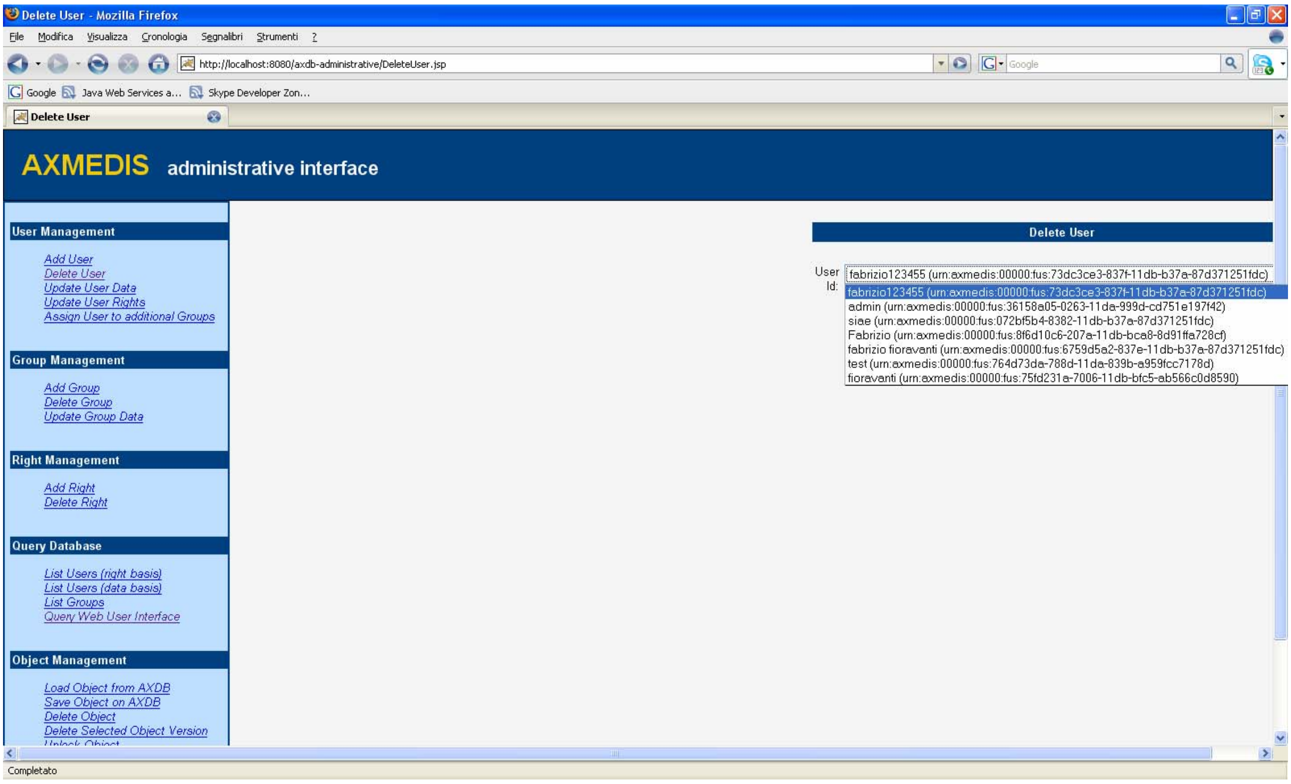
back

(c) Exitech 2005
Web Interface Test

Final version will have the ID prefixed by FUS- (that states for Factory User)

6.5.2.2 Delete user

This page allows to eliminate an user. It will be impossible to delete admin user.



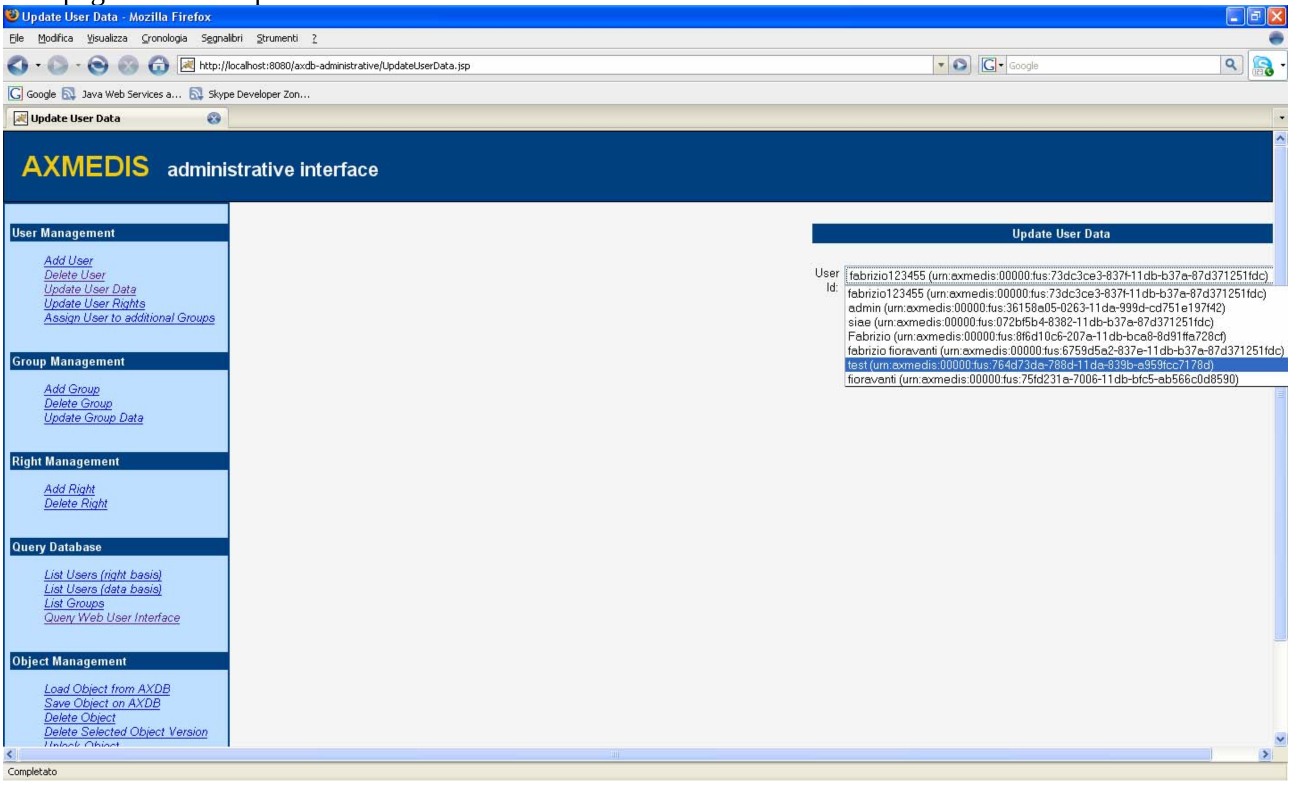
Delete User

User Id: fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)

fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)
admin (urn:axmedis:00000:fus:36158a05-0263-11da-999d-cd751e197f42)
siae (urn:axmedis:00000:fus:072bf5b4-8382-11db-b37a-87d371251fdc)
Fabrizio (urn:axmedis:00000:fus:8f6d10c6-207a-11db-bca8-8d91ffa728cf)
fabrizio fioravanti (urn:axmedis:00000:fus:6759d5a2-837e-11db-b37a-87d371251fdc)
test (urn:axmedis:00000:fus:764d73da-788d-11da-839b-a959fcc7178d)
fioravanti (urn:axmedis:00000:fus:75fd231a-7006-11db-bfc5-ab566c0d8590)

6.5.2.3 Update user data

This page allows to update the user data..

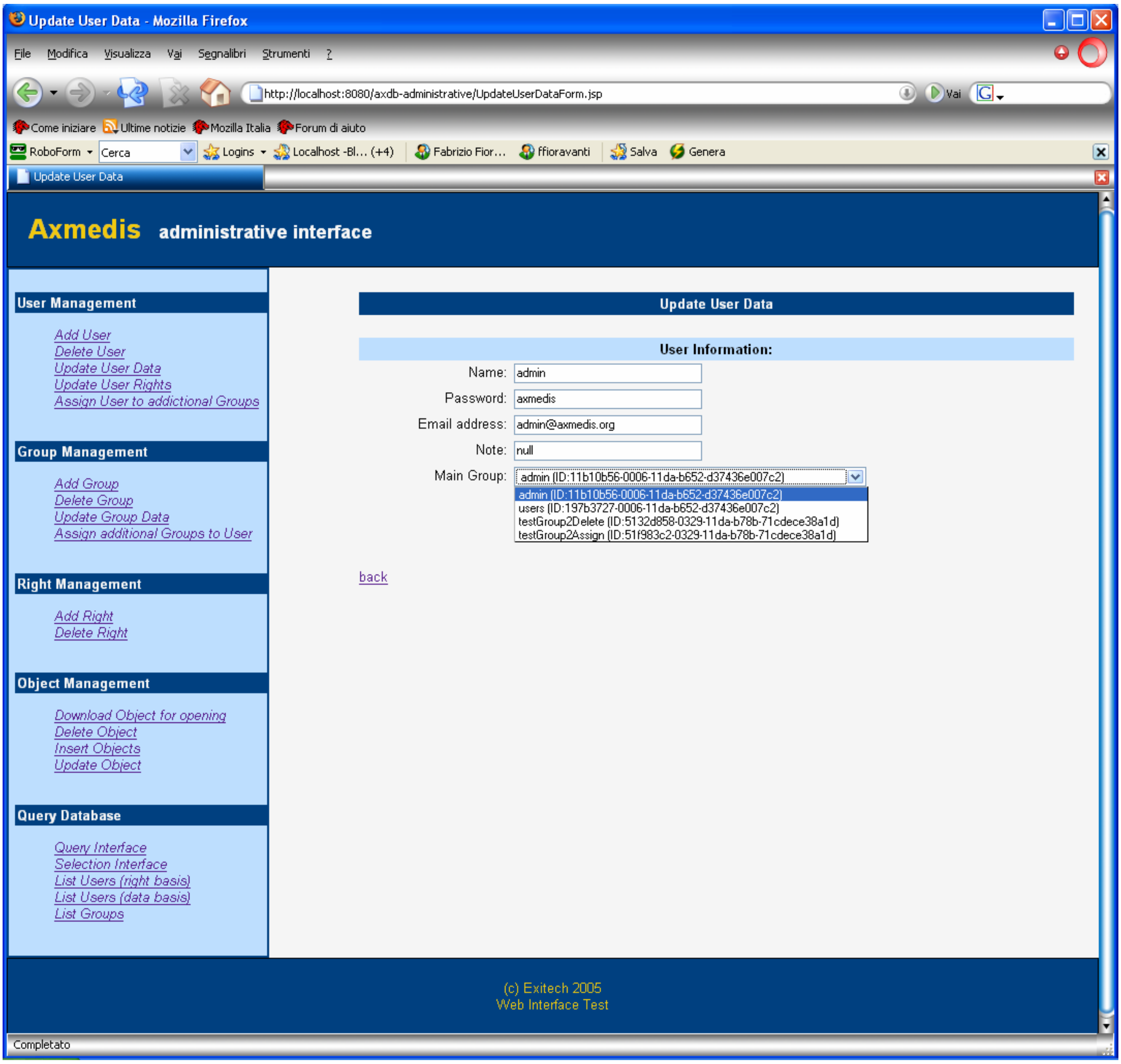


Update User Data

User Id: fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)

- fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)
- admin (urn:axmedis:00000:fus:36158a05-0263-11da-999d-cd751e197f42)
- siae (urn:axmedis:00000:fus:072bf5b4-8382-11db-b37a-87d371251fdc)
- Fabrizio (urn:axmedis:00000:fus:8f6d10c6-207a-11db-bca8-8d91ffa728cf)
- fabrizio fioravanti (urn:axmedis:00000:fus:6759d5a2-837e-11db-b37a-87d371251fdc)
- test (urn:axmedis:00000:fus:764d73da-788d-11da-839b-a959fcc7178d)
- fioravanti (urn:axmedis:00000:fus:75fd231a-7006-11db-bfc5-ab566c0d8590)

Once the user has been identified, it will be possible to change its name, password, email, note and main group.



Update User Data

User Information:

Name: admin

Password: axmedis

Email address: admin@axmedis.org

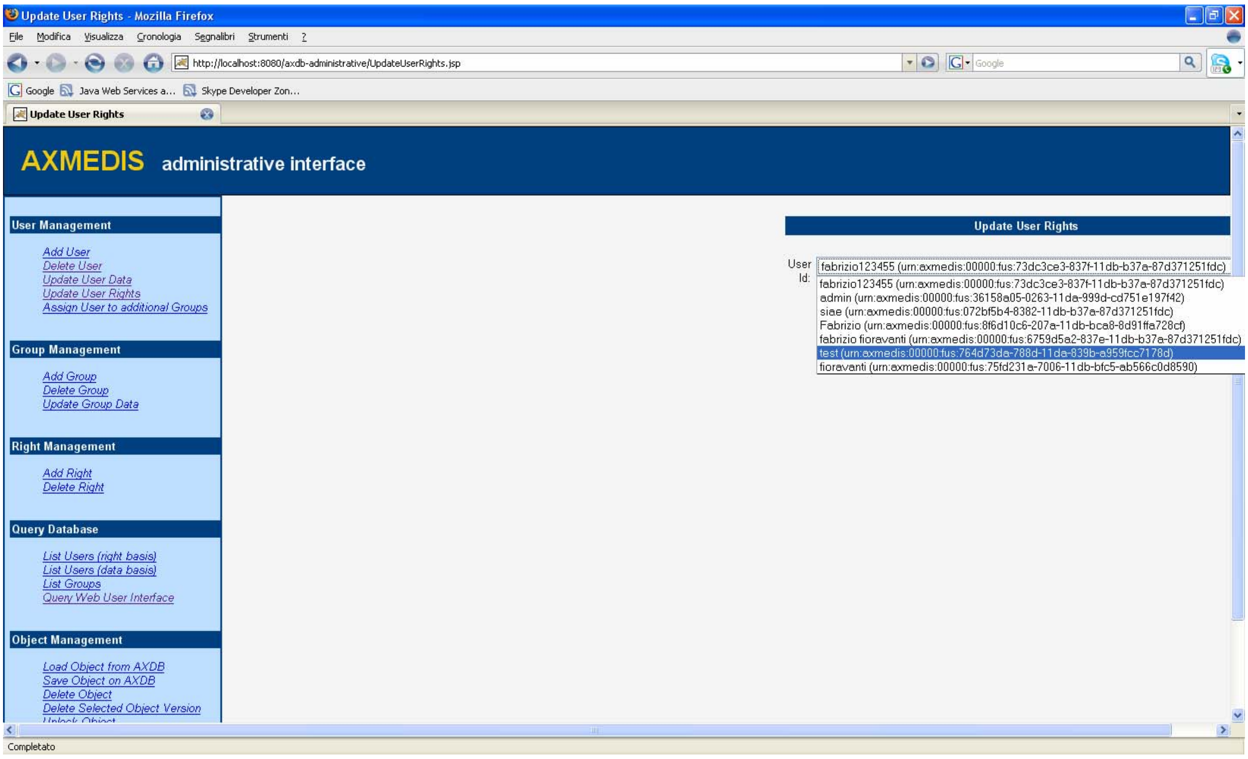
Note: null

Main Group: admin (ID:11b10b56-0006-11da-b652-d37436e007c2)

admin (ID:11b10b56-0006-11da-b652-d37436e007c2)
users (ID:197b3727-0006-11da-b652-d37436e007c2)
testGroup2Delete (ID:5132d858-0329-11da-b78b-71cdece38a1d)
testGroup2Assign (ID:51f983c2-0329-11da-b78b-71cdece38a1d)

6.5.2.4 Update user rights

This page allows to update the user rights..

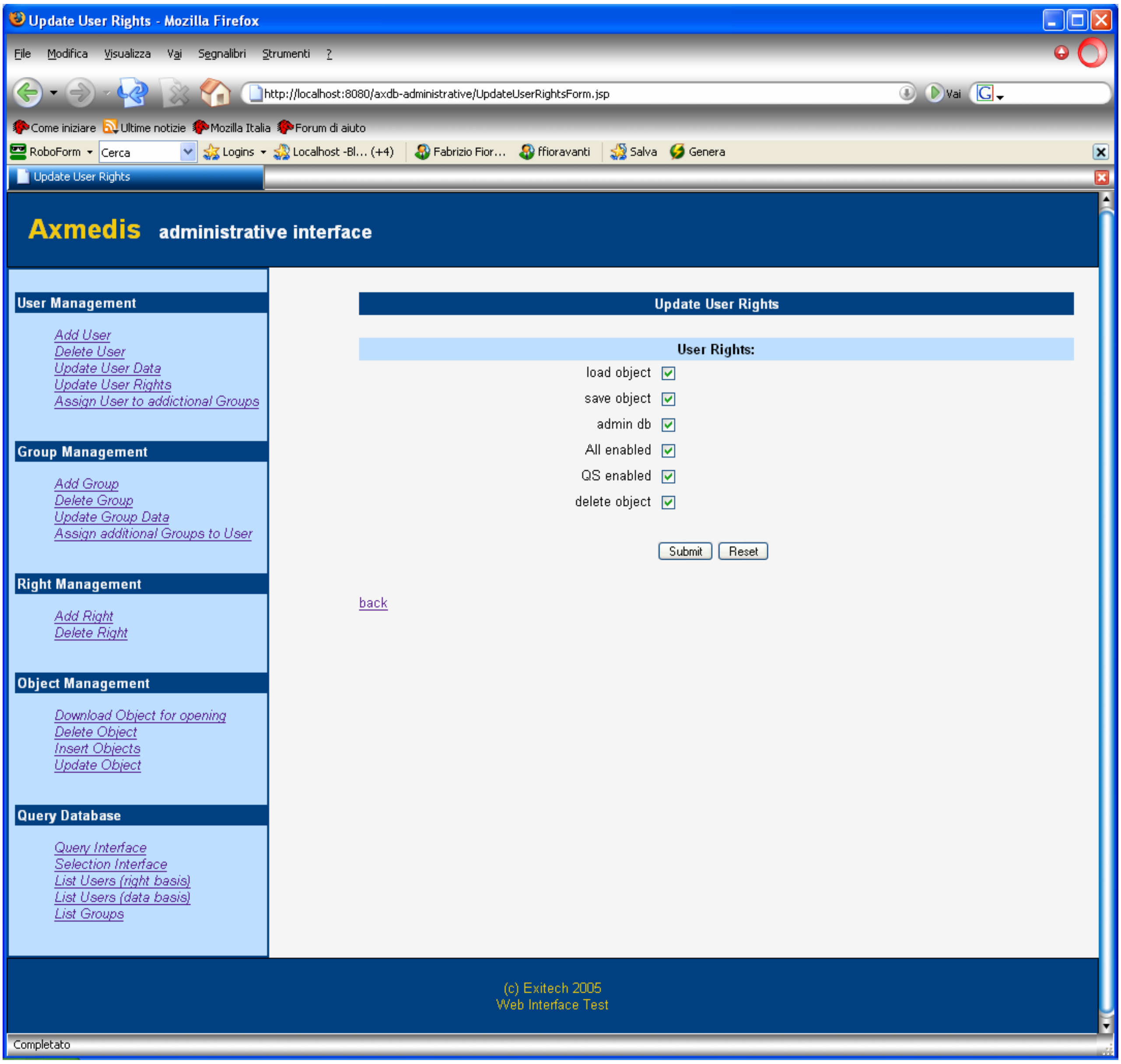


Update User Rights

User Id: fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)

fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)
admin (urn:axmedis:00000:fus:36158a05-0263-11da-999d-cd751e197f42)
siae (urn:axmedis:00000:fus:072bf5b4-8382-11db-b37a-87d371251fdc)
Fabrizio (urn:axmedis:00000:fus:8f6d10c6-207a-11db-bca8-8d91ffa728cf)
fabrizio fioravanti (urn:axmedis:00000:fus:6759d5a2-837e-11db-b37a-87d371251fdc)
test (urn:axmedis:00000:fus:764d73da-788d-11da-839b-a959fcc7178d)
fioravanti (urn:axmedis:00000:fus:75fd231a-7006-11db-bfc5-ab566c0d8590)

Once the user has been identified, it will be possible to change the rights that the user have. Current rights are checked, while available rights not currently active are not checked.


Update User Rights - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/UpdateUserRightsForm.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Update User Rights
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
Submit
Reset
back
(c) Exitech 2005
Web Interface Test
Completato

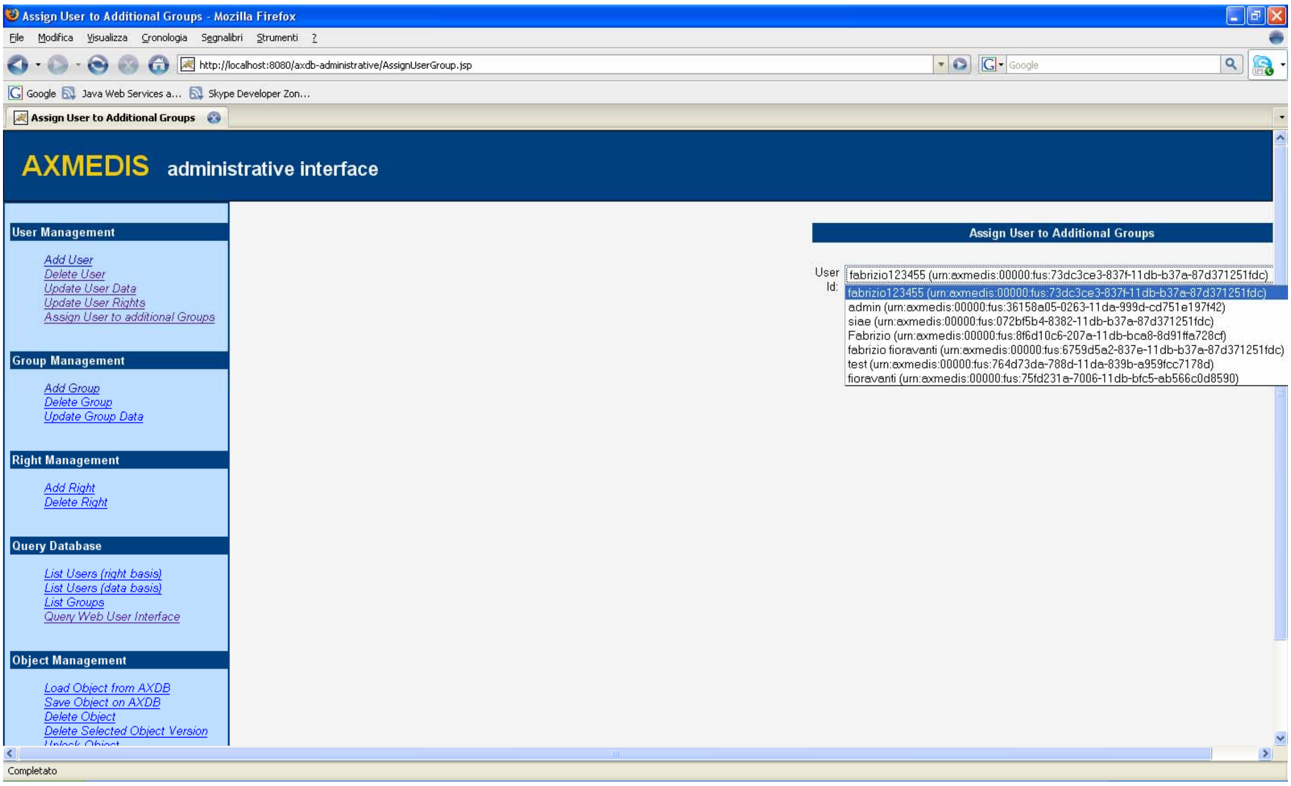
Update User Rights

User Rights:

load object ☑
save object ☑
admin db ☑
All enabled ☑
QS enabled ☑
delete object ☑

6.5.2.5 Assign user to additional groups

This page allows to assign to different groups a single user. In the next version a list of user will appear instead of the text input to insert the user ID.

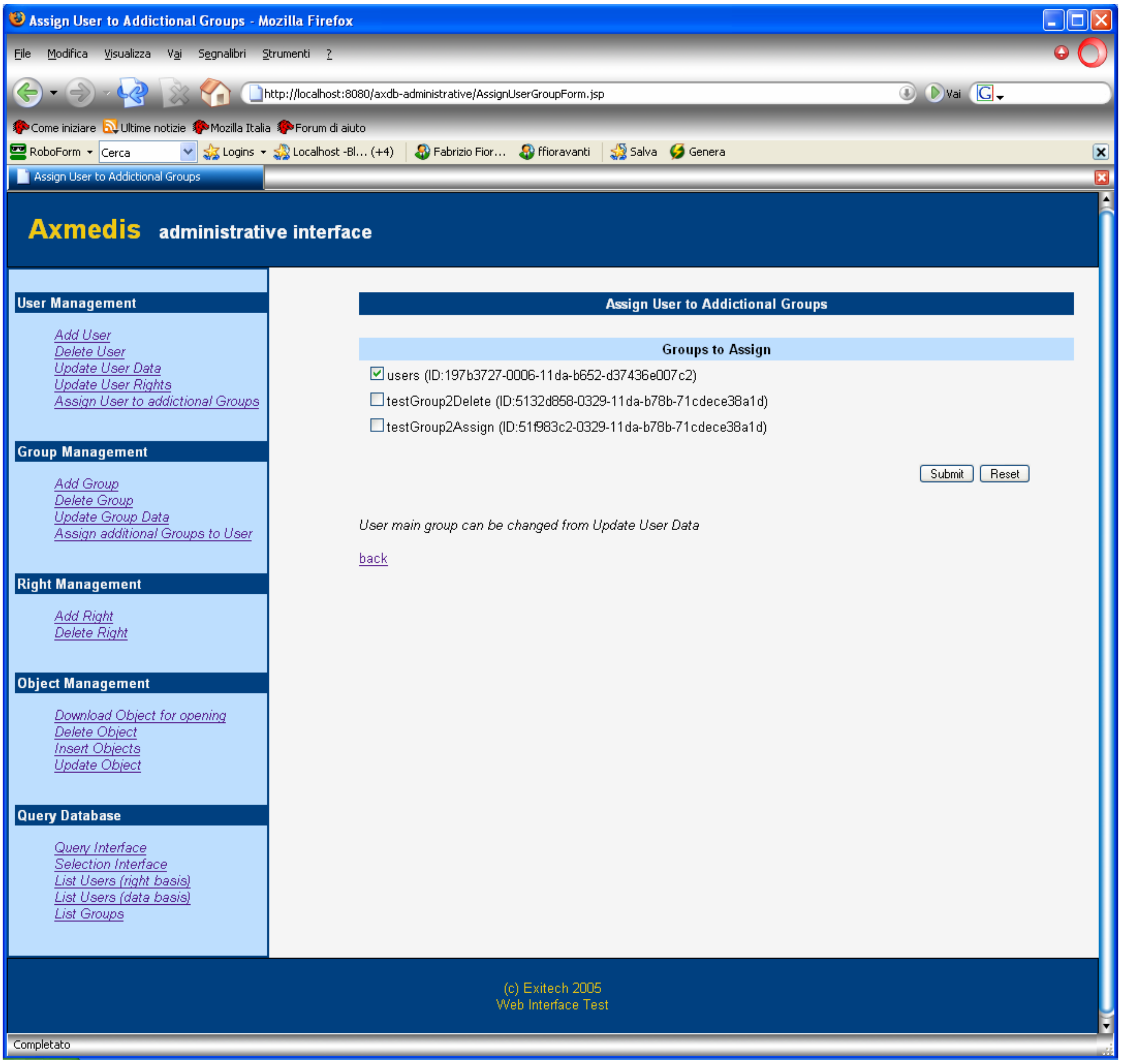


Assign User to Additional Groups

User Id: fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)

- fabrizio123455 (urn:axmedis:00000:fus:73dc3ce3-837f-11db-b37a-87d371251fdc)
- admin (urn:axmedis:00000:fus:36158a05-0263-11da-999d-cd751e197f42)
- siae (urn:axmedis:00000:fus:072bf5b4-8382-11db-b37a-87d371251fdc)
- Fabrizio (urn:axmedis:00000:fus:8f6d10c6-207a-11db-bca8-8d91ffa728cf)
- fabrizio fioravanti (urn:axmedis:00000:fus:6759d5a2-837e-11db-b37a-87d371251fdc)
- test (urn:axmedis:00000:fus:764d73da-788d-11da-839b-a959fcc7178d)
- fioravanti (urn:axmedis:00000:fus:75fd231a-7006-11db-bfc5-ab566c0d8590)

Once the user is selected it is possible to add groups to it.


Assign User to Addictional Groups - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/AssignUserGroupForm.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Assign User to Addictional Groups
Completato

Axmedis administrative interface

User Management

- Add User
- Delete User
- Update User Data
- Update User Rights
- Assign User to addictional Groups

Group Management

- Add Group
- Delete Group
- Update Group Data
- Assign additional Groups to User

Right Management

- Add Right
- Delete Right

Object Management

- Download Object for opening
- Delete Object
- Insert Objects
- Update Object

Query Database

- Query Interface
- Selection Interface
- List Users (right basis)
- List Users (data basis)
- List Groups

Assign User to Addictional Groups

Groups to Assign

☑ users (ID:197b3727-0006-11da-b652-d37436e007c2)
□ testGroup2Delete (ID:5132d858-0329-11da-b78b-71cdece38a1d)
□ testGroup2Assign (ID:51f983c2-0329-11da-b78b-71cdece38a1d)

User main group can be changed from Update User Data

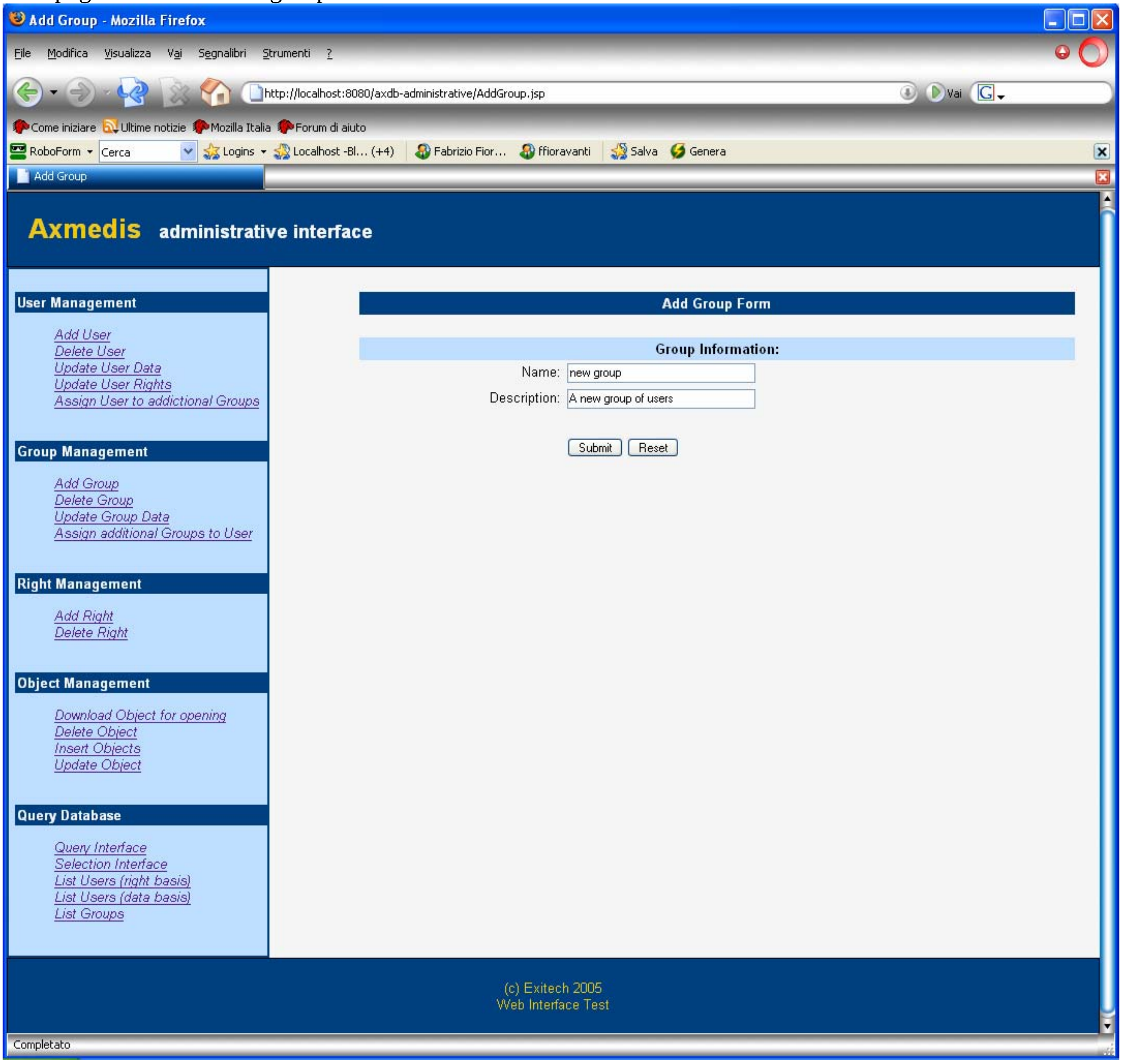
back

(c) Exitech 2005
Web Interface Test

6.5.3 Group Management

6.5.3.1 Add Group

This page allows to add a group.



Add Group Form

Group Information:

Name: new group

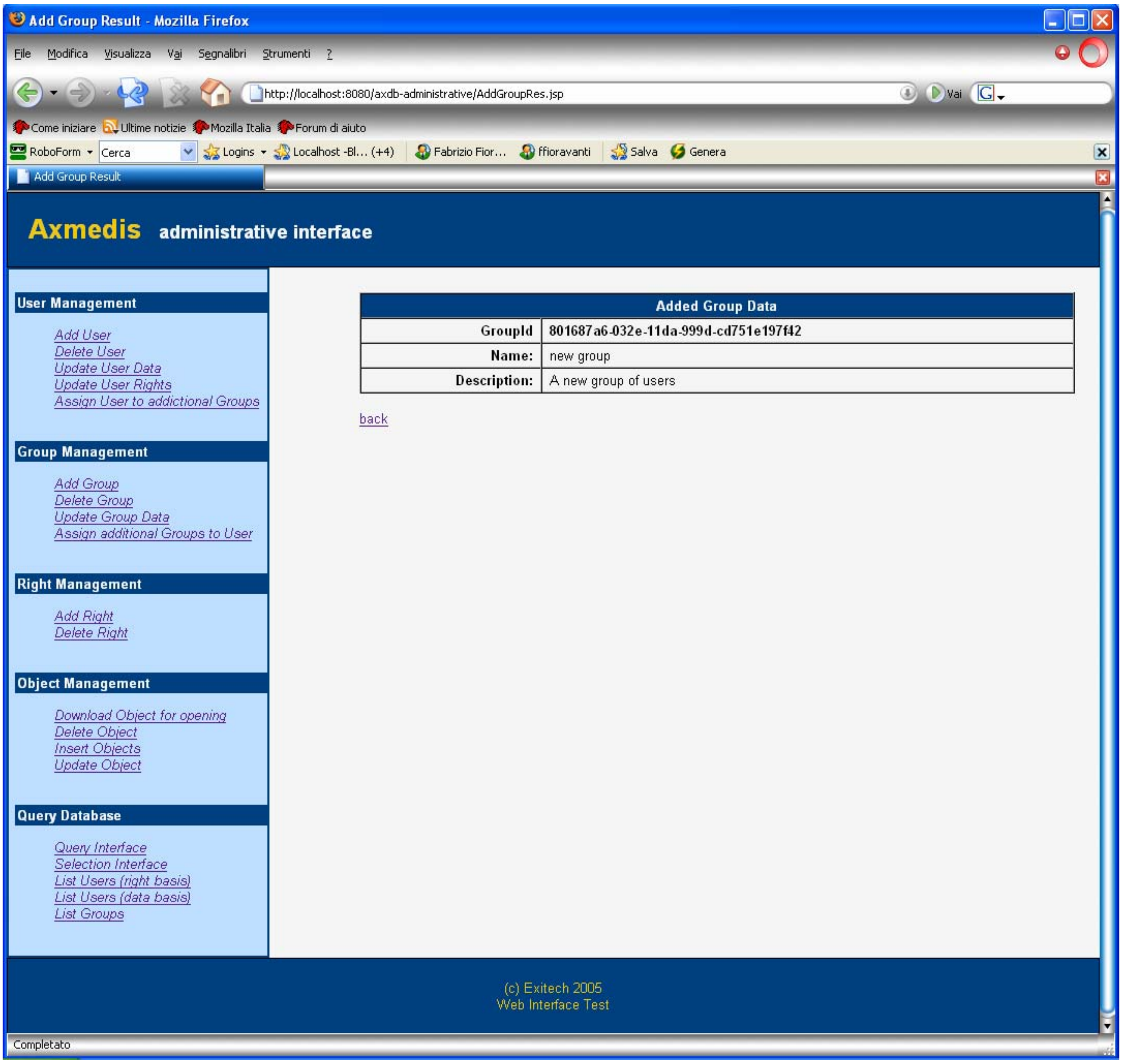
Description: A new group of users

After submit a confirmation page showing the automatically generated group ID will be shown.

Final version must have GroupID be prefixed by FGR- (that states for Factory Group).

In the final version the groups will be associated also to a set of rights that will be inherited by all the user belonging to a group.

These additional righst will not be modified by the user management but only by the group management.


Add Group Result - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/AddGroupRes.jsp
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Add Group Result

Axmedis administrative interface

User Management

- Add User
- Delete User
- Update User Data
- Update User Rights
- Assign User to addictional Groups

Group Management

- Add Group
- Delete Group
- Update Group Data
- Assign additional Groups to User

Right Management

- Add Right
- Delete Right

Object Management

- Download Object for opening
- Delete Object
- Insert Objects
- Update Object

Query Database

- Query Interface
- Selection Interface
- List Users (right basis)
- List Users (data basis)
- List Groups

Added Group Data	
GroupId	801687a6-032e-11da-999d-cd751e197f42
Name:	new group
Description:	A new group of users

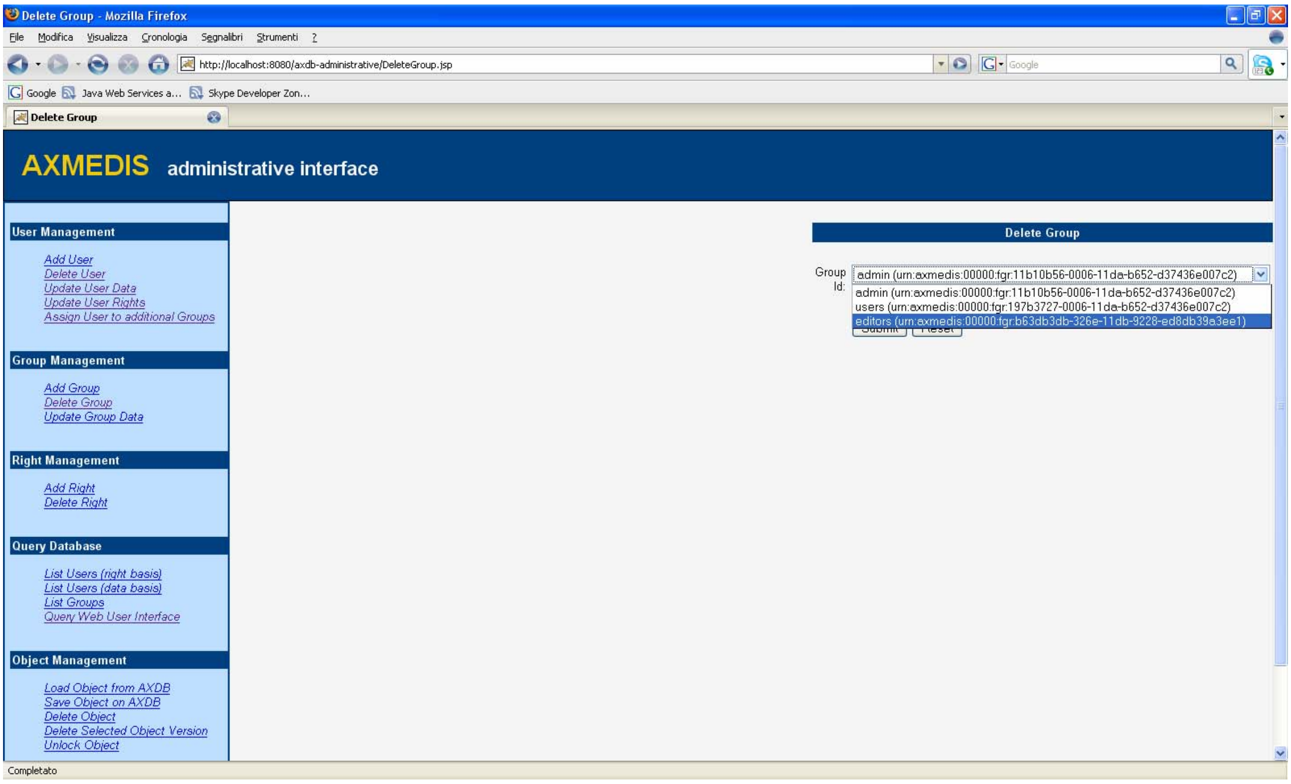
back

(c) Exitech 2005
Web Interface Test

Completato

6.5.3.2 Delete group

This page allows to delete a group. In the next version a list of groups will appear instead of the text input to insert the group ID.

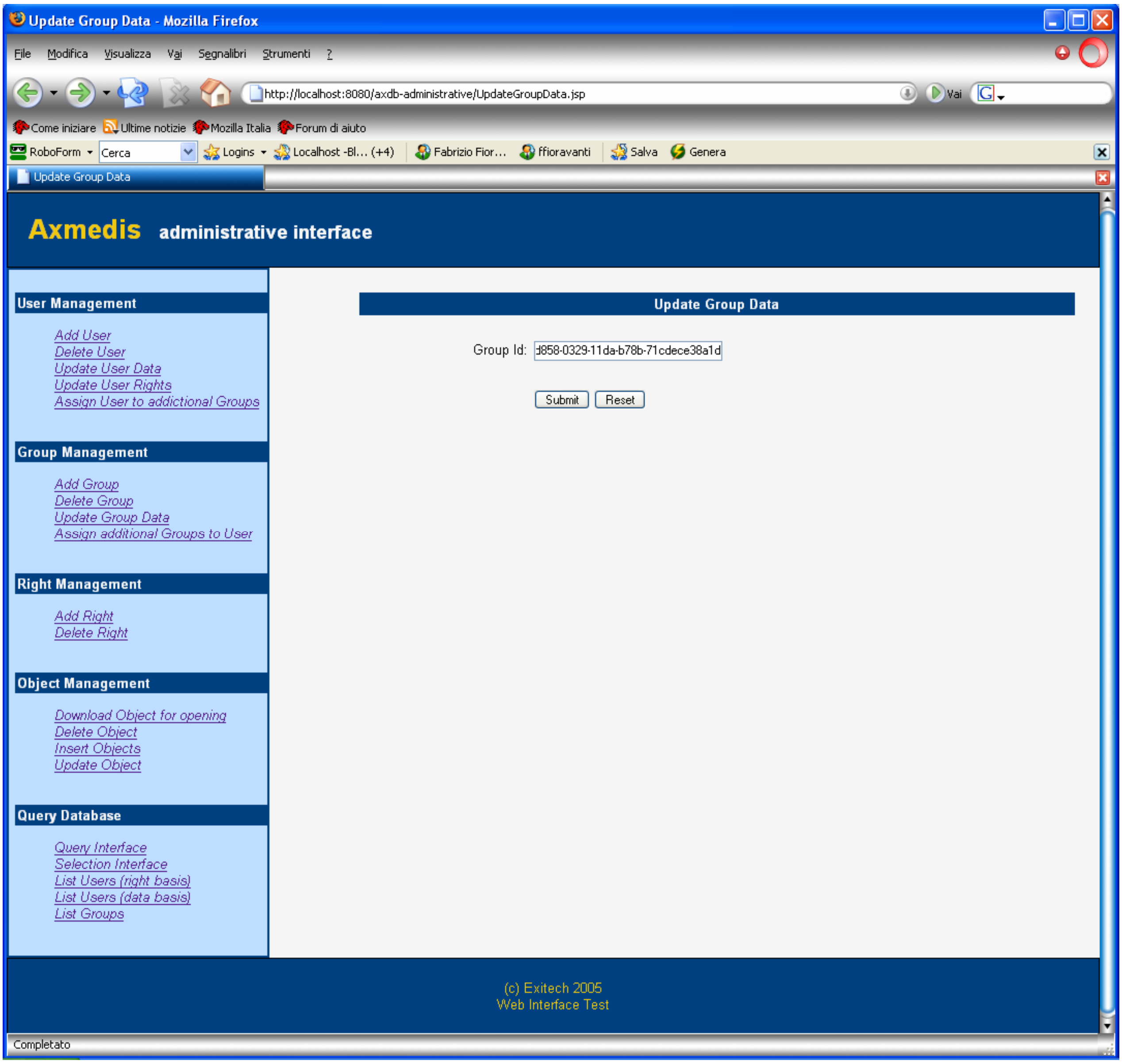


Group Id: admin (urn:axmedis:00000:fgr:11b10b56-0006-11da-b652-d37436e007c2)

admin (urn:axmedis:00000:fgr:11b10b56-0006-11da-b652-d37436e007c2)
users (urn:axmedis:00000:fgr:197b3727-0006-11da-b652-d37436e007c2)
editors (urn:axmedis:00000:fgr:b63db3db-326e-11db-9228-ed8db39a3ee1)

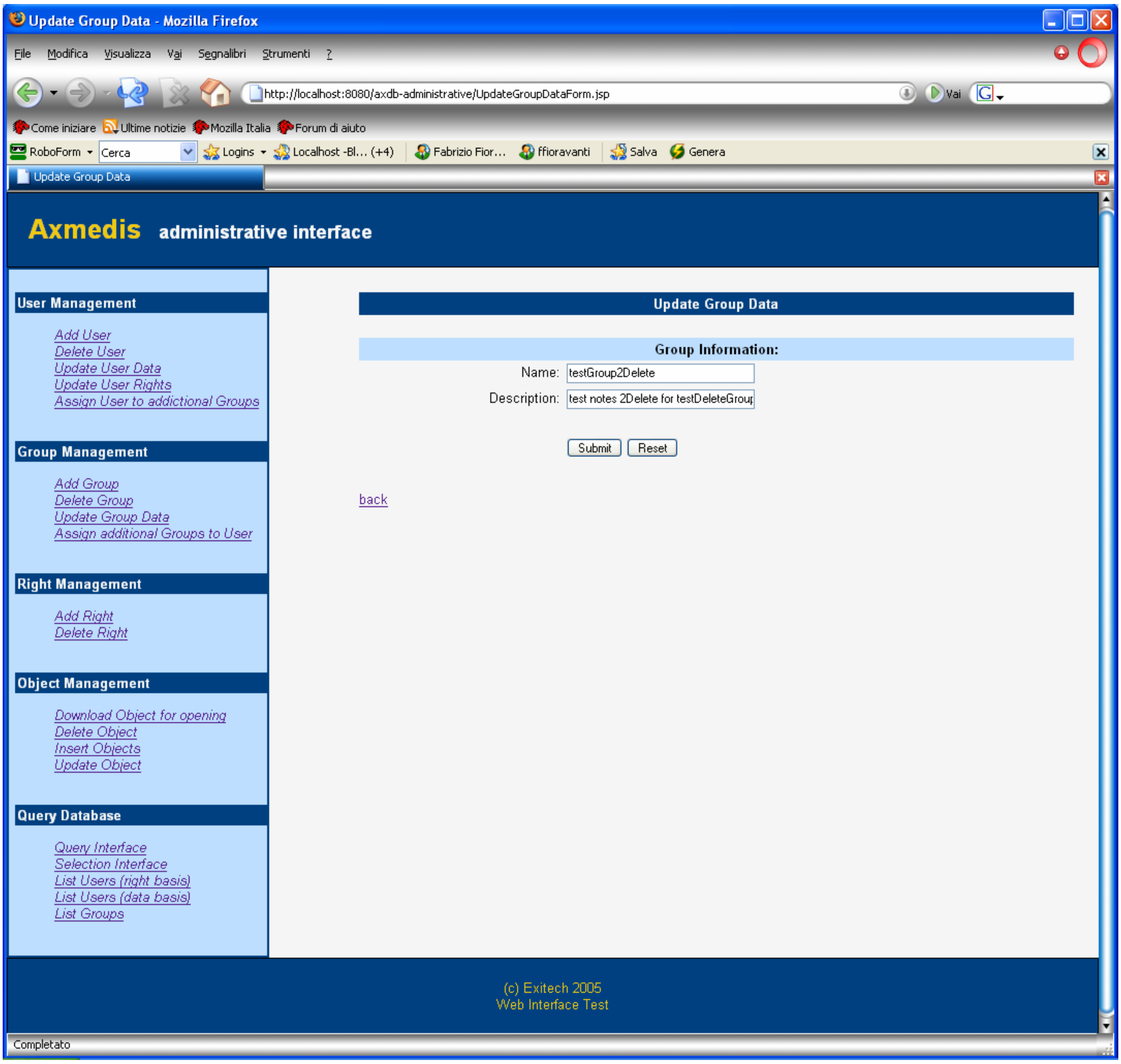
6.5.3.3 Update group data

This page allows to update a group related data.


Update Group Data - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/UpdateGroupData.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Update Group Data
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
Update Group Data
Submit
Reset
(c) Exitech 2005
Web Interface Test
Completato

Group Id: d858-0329-11da-b78b-71cdece38a1d

After choosing the group the modifiable data will appear.



Update Group Data - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/UpdateGroupDataForm.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Update Group Data

Axmedis administrative interface

User Management

- Add User
- Delete User
- Update User Data
- Update User Rights
- Assign User to addictional Groups

Group Management

- Add Group
- Delete Group
- Update Group Data
- Assign additional Groups to User

Right Management

- Add Right
- Delete Right

Object Management

- Download Object for opening
- Delete Object
- Insert Objects
- Update Object

Query Database

- Query Interface
- Selection Interface
- List Users (right basis)
- List Users (data basis)
- List Groups

Update Group Data

Group Information:

Name: testGroup2Delete

Description: test notes 2Delete for testDeleteGrour

Submit Reset

back

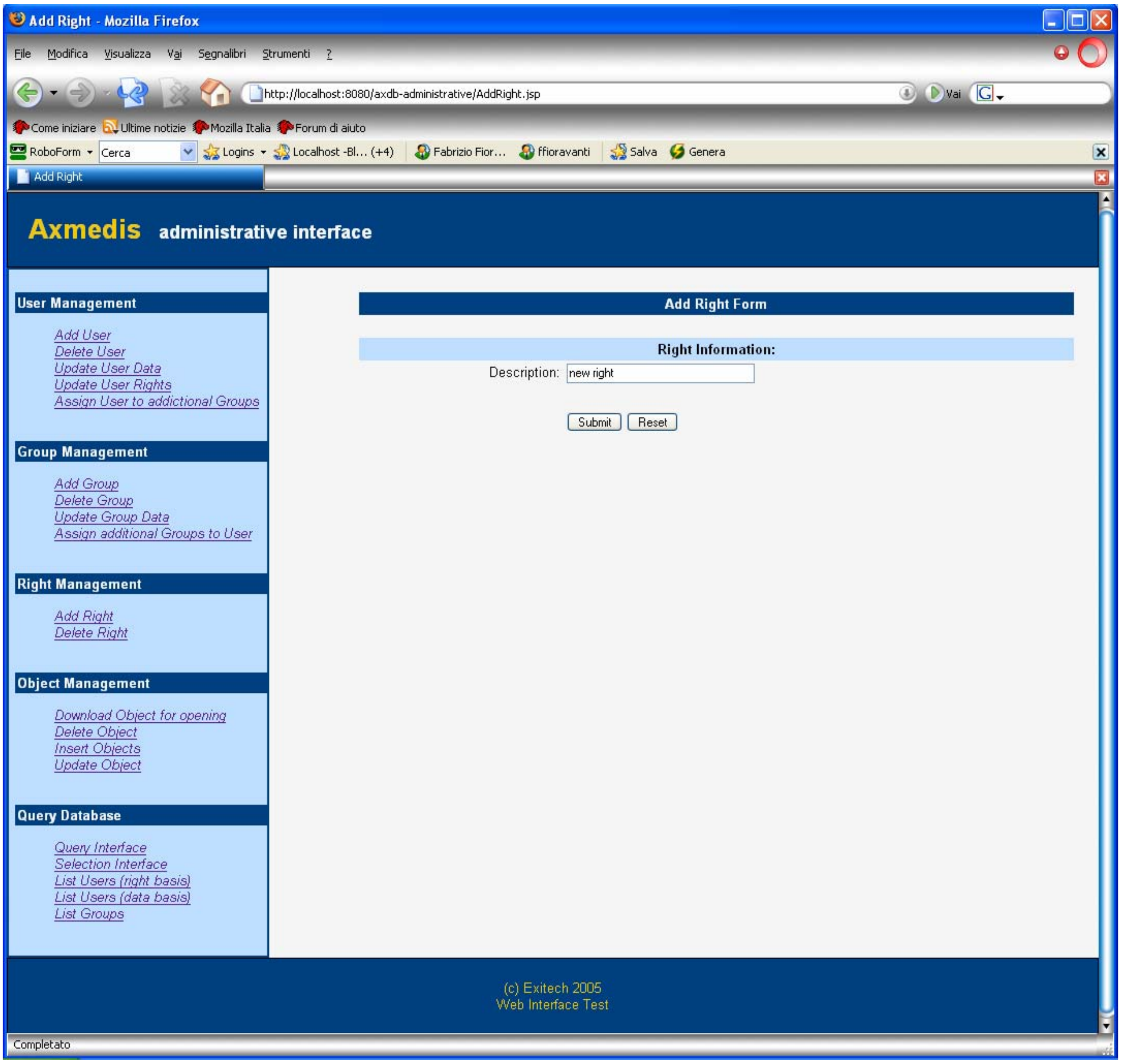
(c) Exitech 2005
Web Interface Test

6.5.4 Right Management

This section allows to add and delete rights that can be assigned to the users.

6.5.4.1 Add right

This page creates a new right.

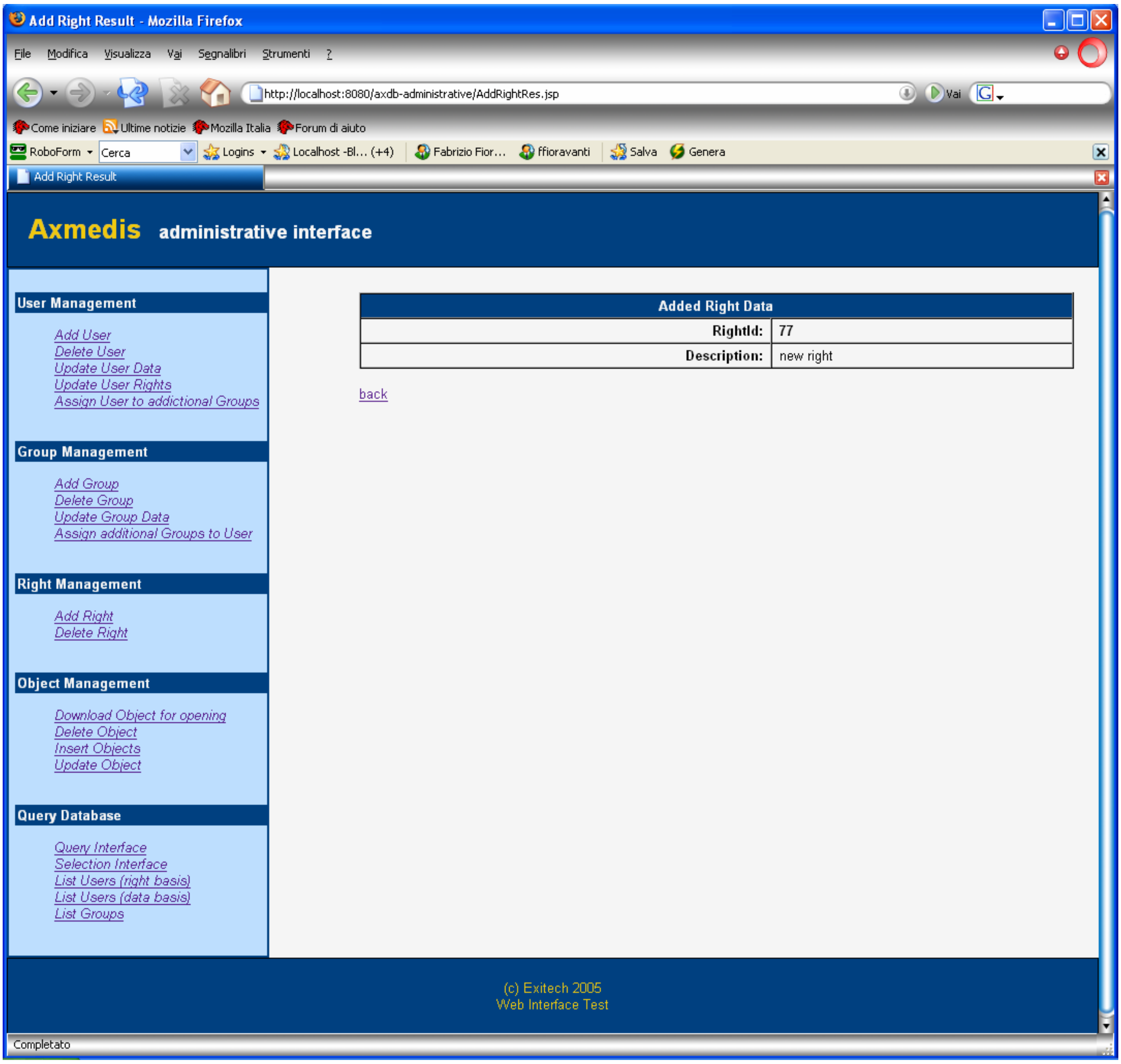

Add Right - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/AddRight.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Add Right
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
Submit
Reset
(c) Exitech 2005
Web Interface Test
Completato

Add Right Form

Right Information:

Description: new right

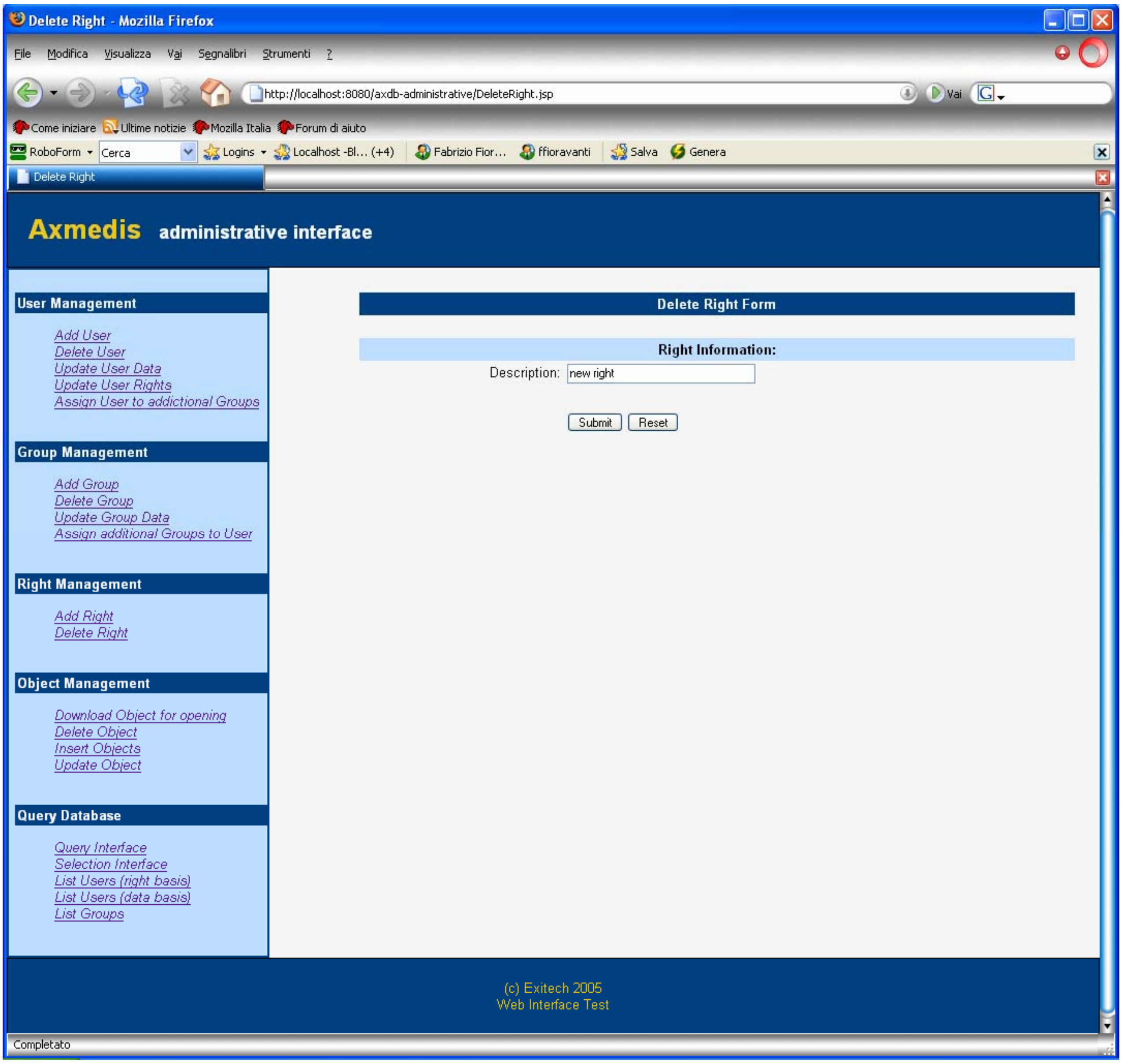
After the creation a summary screen is shown.


Add Right Result - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/AddRightRes.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Add Right Result
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
back
(c) Exitech 2005
Web Interface Test
Completato

Added Right Data	
RightId:	77
Description:	new right

6.5.4.2 Delete right

This page, given the right name, allows to eliminate such right from the DB.


Delete Right - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/DeleteRight.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
Delete Right
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
Submit
Reset
(c) Exitech 2005
Web Interface Test
Completato

Delete Right Form

Right Information:

Description: new right

6.5.5 Object Management

This section will allows to manage object (load/save/update/remove) and will be mainly based on Loader/Saver functionalities.

In addition to the menu item represented in the previous screenshoots, some other operations will be possible, such as:

- Mark an object as unavailable
- Delete one or more version of an object

6.5.6 Query Database

This section is useful for collecting information related to user groups and to access to the query and selection interfaces.

6.5.6.1 Query Interface

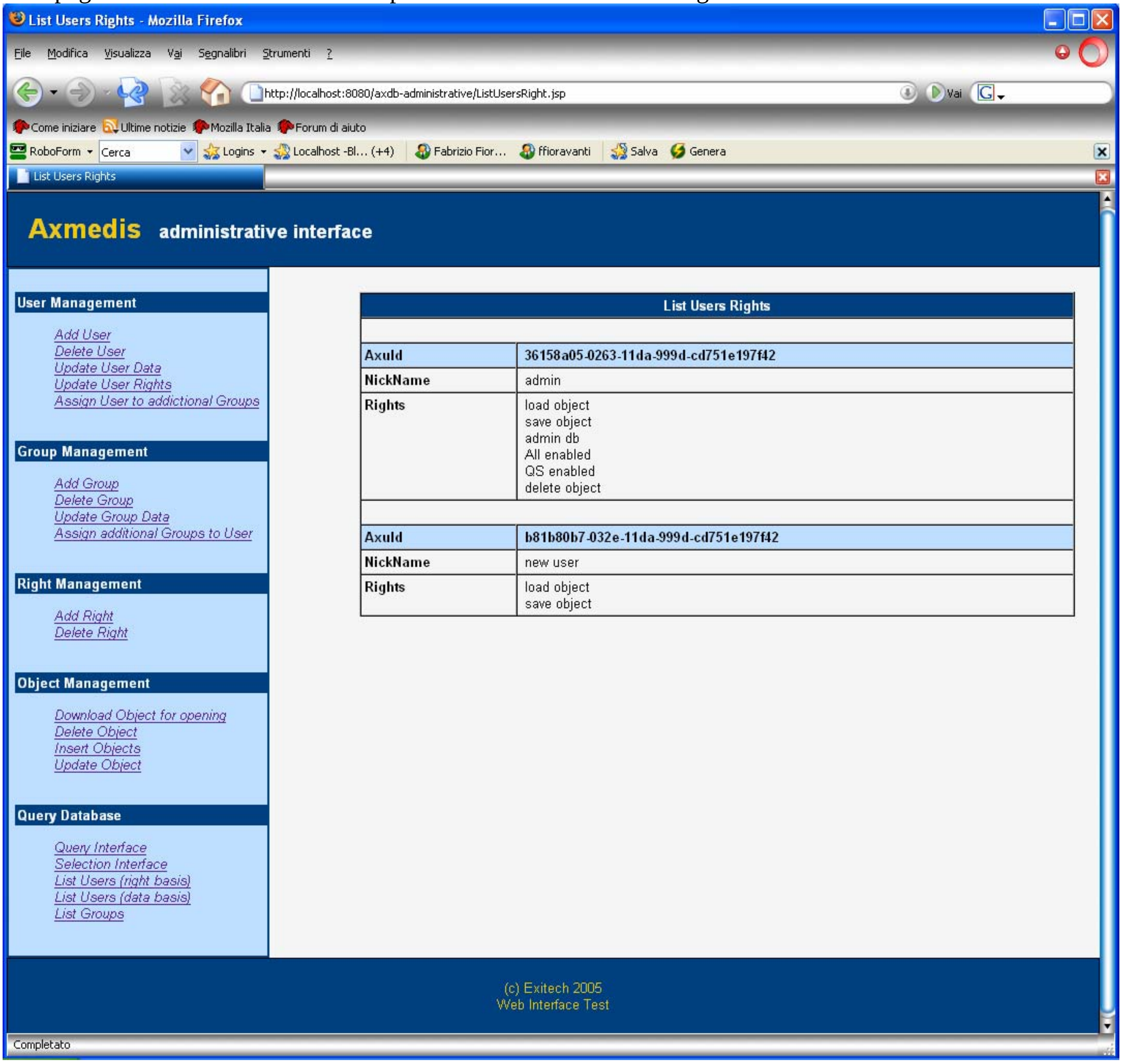
It is a link to the Web Query User Interface. See the interface in Query User Interface.

6.5.6.2 Selection Interface

It is a link to the Web Selection Interface. See the interface in Selection Interface.

6.5.6.3 List Users (right basis)

This page shows all the users that are present in the DB with their rights.

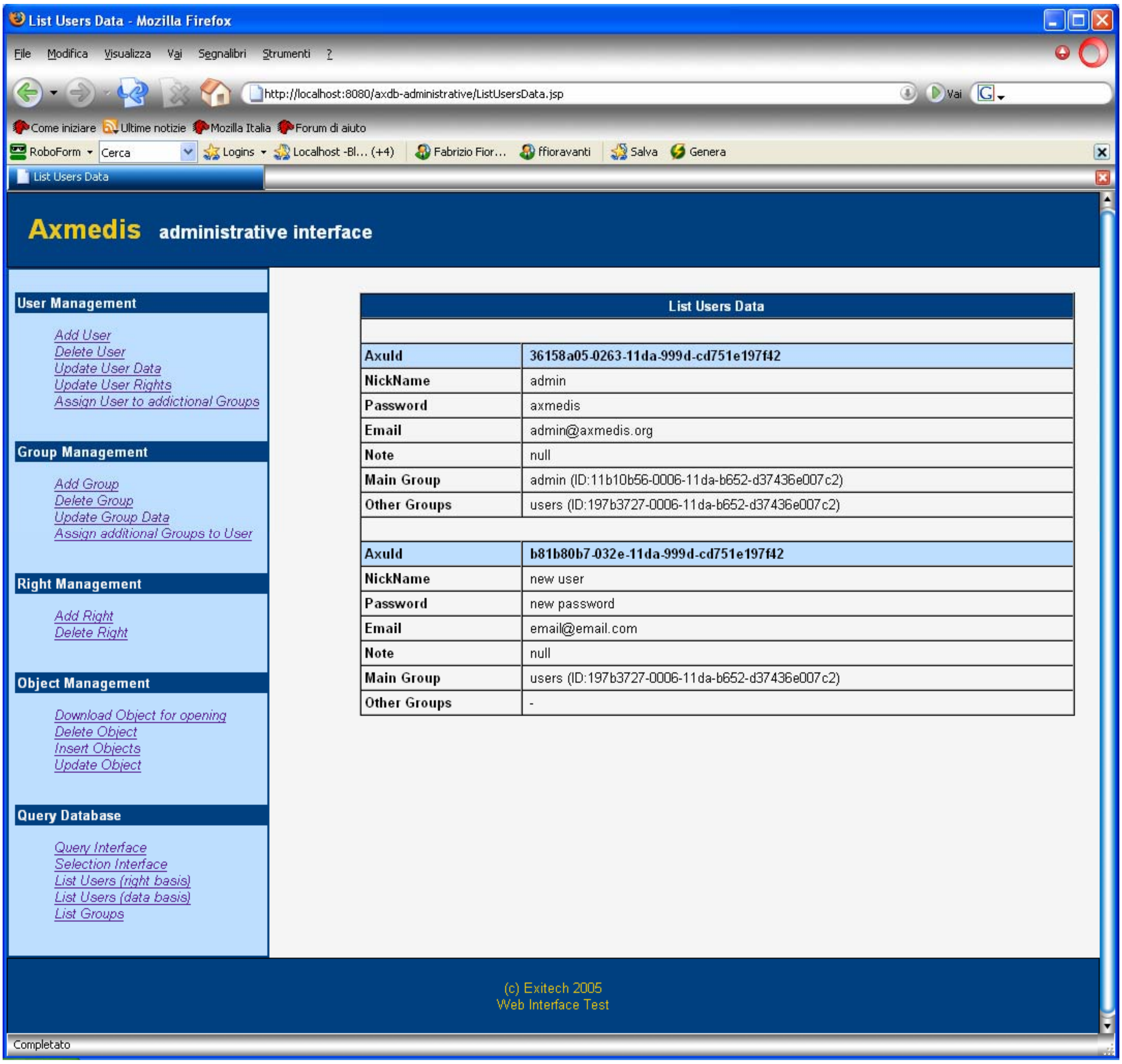

List Users Rights - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/ListUsersRight.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
List Users Rights
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
(c) Exitech 2005
Web Interface Test
Completato

List Users Rights

AxuId	36158a05-0263-11da-999d-cd751e197f42
NickName	admin
Rights	load object save object admin db All enabled QS enabled delete object
AxuId	b81b80b7-032e-11da-999d-cd751e197f42
NickName	new user
Rights	load object save object

6.5.6.4 List Users (data basis)

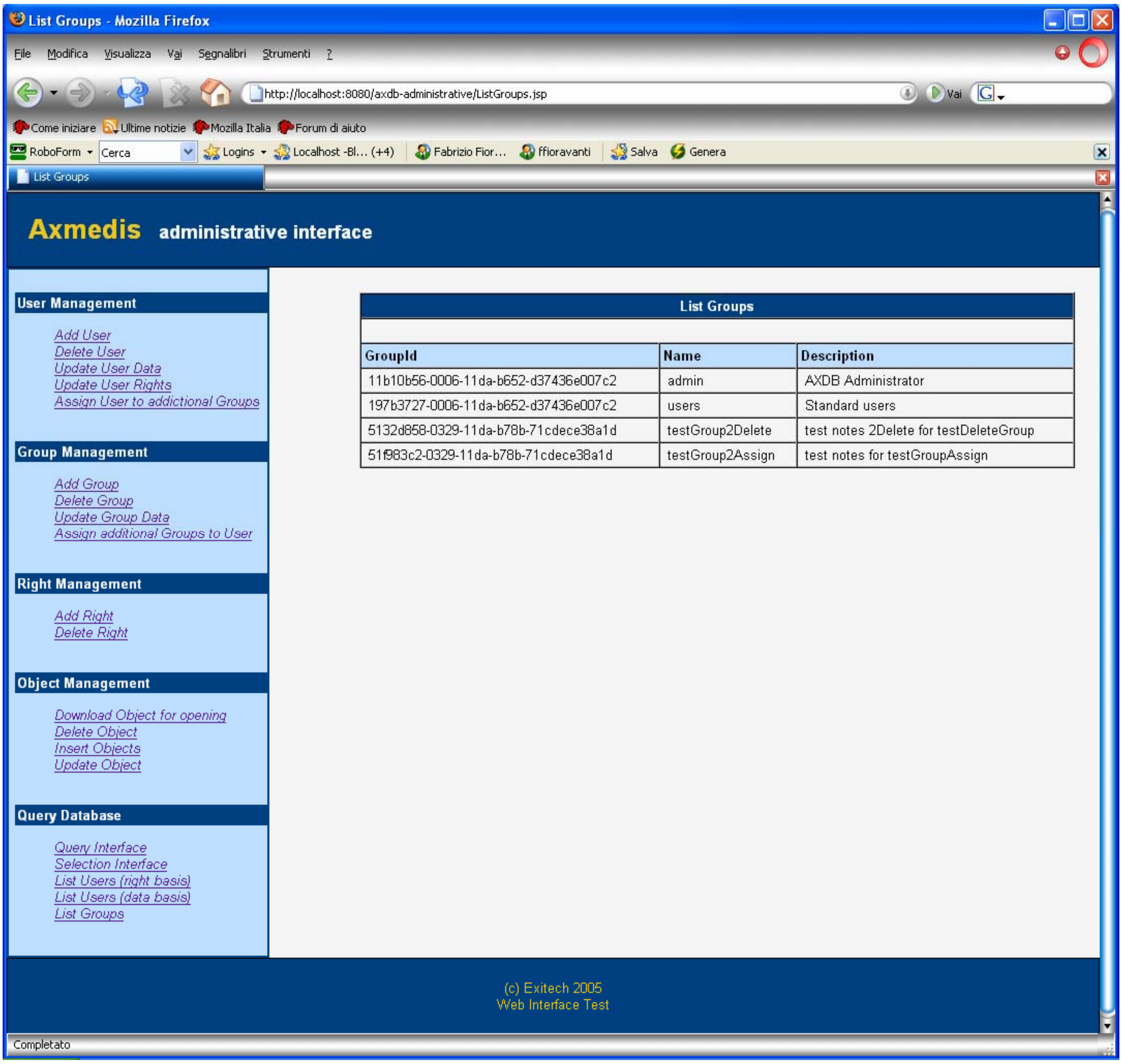
This page shows all the users that are present in the DB with their data.


List Users Data - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/ListUsersData.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
List Users Data
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
(c) Exitech 2005
Web Interface Test
Completato

List Users Data	
AxuId	36158a05-0263-11da-999d-cd751e197f42
NickName	admin
Password	axmedis
Email	admin@axmedis.org
Note	null
Main Group	admin (ID:11b10b56-0006-11da-b652-d37436e007c2)
Other Groups	users (ID:197b3727-0006-11da-b652-d37436e007c2)
AxuId	b81b80b7-032e-11da-999d-cd751e197f42
NickName	new user
Password	new password
Email	email@email.com
Note	null
Main Group	users (ID:197b3727-0006-11da-b652-d37436e007c2)
Other Groups	-

6.5.6.5 List groups

This page shows all the groups that are present in the DB with their data.



List Groups - Mozilla Firefox
File Modifica Visualizza Vai Segnalibri Strumenti ?
http://localhost:8080/axdb-administrative/ListGroups.jsp
Vai
Come iniziare Ultime notizie Mozilla Italia Forum di aiuto
RoboForm Cerca Logins Localhost -Bl... (+4) Fabrizio Fior... ffioravanti Salva Genera
List Groups
Axmedis administrative interface
User Management
Add User
Delete User
Update User Data
Update User Rights
Assign User to addictional Groups
Group Management
Add Group
Delete Group
Update Group Data
Assign additional Groups to User
Right Management
Add Right
Delete Right
Object Management
Download Object for opening
Delete Object
Insert Objects
Update Object
Query Database
Query Interface
Selection Interface
List Users (right basis)
List Users (data basis)
List Groups
(c) Exitech 2005
Web Interface Test
Completato

List Groups

GroupId	Name	Description
11b10b56-0006-11da-b652-d37436e007c2	admin	AXDB Administrator
197b3727-0006-11da-b652-d37436e007c2	users	Standard users
5132d858-0329-11da-b78b-71cdece38a1d	testGroup2Delete	test notes 2Delete for testDeleteGroup
51f983c2-0329-11da-b78b-71cdece38a1d	testGroup2Assign	test notes for testGroupAssign

6.6 Examples of usage

See the user manual in section 6.5

6.7 Integration and compilation issues

Since the application is a standard JAVA 1.5 application there is no particular issue

6.8 Configuration Parameters

6.8.1 axdb.properties file

Config parameter	Possible values
`axdbUrl`	`jdbc:mysql://mysqlhost/axdb-name?jdbcCompliantTruncation=false`
axdbUser	User to be used for accessing the database (it depends on your installation of AXDB)
axdbPwd	Password of the User to be used for accessing the database (it depends on your installation of AXDB)
`axdbMapping`	NOT REQUESTED FOR THIS MODULE. The property file is shared with other modules that requires this parameter

7 AXMEDIS Loader Saver (EXITECH)

Module/Tool Profile	
AXMEDIS Loader Saver	
Responsible Name	Grassi
Responsible Partner	EXITECH
Status (proposed/approved)	approved
Implemented/not implemented	Implemented
Status of the implementation	Completede, under maintenance
Executable or Library/module (Support)	
Single Thread or Multithread	Multithread
Language of Development	JAVA and C++
Platforms supported	Windows
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/WebServices/LoadeSaverWs/src/
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/repos/WebServices/LoadeSaverWs/bin/
Reference to the AXFW location of the demonstrator executable tool for public download	
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	Saver WS Endpoint: http://81.73.104.125:8080/LoadSaveWS/save WSDL: http://81.73.104.125:8080/LoadSaveWS/save?WSDL Loader WS Endpoint: http://81.73.104.125:8080/LoadSaveWS/load WSDL: http://81.73.104.125:8080/LoadSaveWS/load?WSDL loadListener Endpoint: http://81.73.104.125:8080/LoadSaveWS/CommitListener WSDL: http://81.73.104.125:8080/LoadSaveWS/CommitListener?WSDL saveListener Endpoint: http://81.73.104.125:8080/LoadSaveWS/commitlist WSDL: http://81.73.104.125:8080/LoadSaveWS/commitlist?WSDL User: test Password: test
Test cases (present/absent)	Media Club Files
Test cases location	http://www.axmedis.org/documenti/view_documenti.php?doc_id=1639
Usage of the	no

AXMEDIS configuration manager (yes/no)		
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved	-- --	
Major pending requirements	Indexing of optional fields	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section
Protocol Used	Shared with	Protocol name or reference to a section
Soap		
JNI		
Used Database name		
AXDB		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
		JNI, JWSDP, XALAN
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

7.1 General Description of the Module

AXMEDIS object Loader/Saver is a Web service that is capable of getting an AXMEDIS object (in the MPEG21 compliant format plus the additional information stored inside) and putting it in the database, that is the saving function; and it is also capable, given an Object ID (AXOID) to return the AXMEDIS object in the MPEG21 compliant format, that is the loading function.

The load and save function are always considered from the point of view of the user, so that Load means “load from AMXEDIS” and save means import inside AMXEDIS.

The AXMEDIS Loader/Saver interact, or better is part of, the AXMEDIS database Interface.

This module offers the services for saving in the database objects and for loading objects from the database. This module can be split in the Loader and Saver in order to have a better understanding of the different behavior of the two services.

Both services of this module are implemented as web-services and therefore in the following specification, the WSDL of the proposed service together with samples of the SOAP messages will be reported.

The services can operate in synchronous and asynchronous made. In asynchronous mode a Listener approach will be implemented, in the sense that the user, in order to be allowed to use asynchronous operation must offer a web-service with a known interface to receive the result of the operation.

The saver module will automatically set the version of the object inside the object as soon as the object is saved increasing the number currently present in the database.

7.1.1 Saver/Indexer (EXITECH, DSI)

Saver/Indexer module offers basically the commit service, both in sync and async mode.

During the saving process it is necessary to know which fields of the Descriptors have to be imported as optional fields. To cover this part of mapping that is very simple from the database side, during the installation and configuration of the AXMEDIS system, it will be created a file that will be read by the saver each time it has to import an axmedis object inside the AXDB.

The format proposed for the mapping is reported in section 20.

The web service definition is reported in section 26

Since a listener is defined in the asynchronous mode, it is necessary to draw the specification also of this web service that will be offered and implemented by a third party that uses the asynchronous commit. A sample of what shall be implemented is reported in section 27.

Since the whole interface in terms database interface has been realized in Java and since all the other web services have been realized and deployed in a JAVA environment, also the saver DB has been approached from the JAVA point of view.

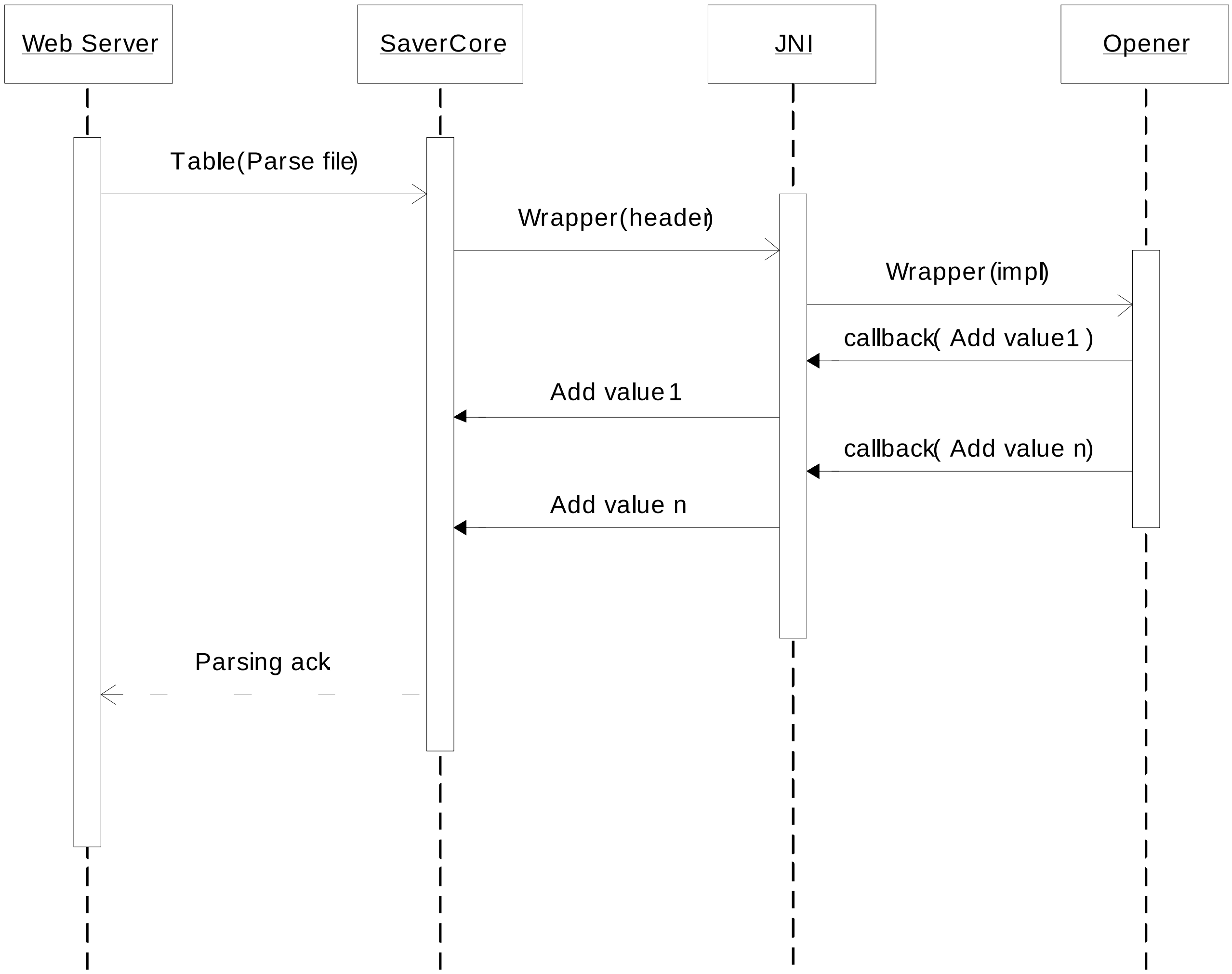
The main problem to be faced has been AXOM.lib that was written in C++ and the guarantee of security of the data during the process of storing in the database.

It has been decided to create all the logic that extracts metadata and opens the objects directly in C++ in a DLL that wrap the AXOM.

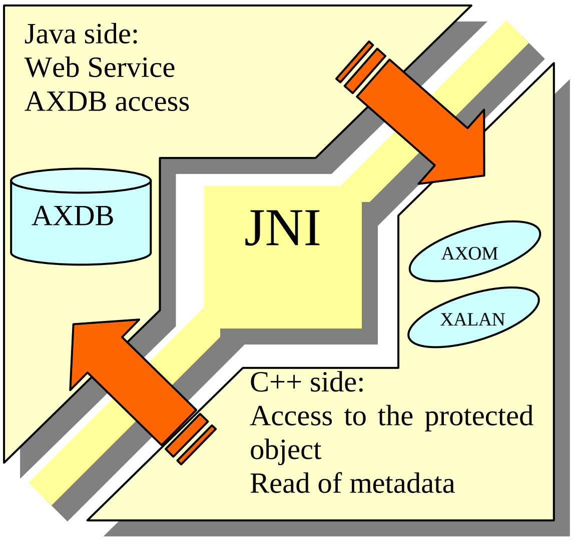
In order to access to the DLL functionality it has been decided to use JNI, so that the JAVA application has to invoke the C++ Wrapper class via JNI that performs all critical operations directly in C++ via the Opener class that calls back JAVA giving back the values of the metadata necessary for indexing. The JAVA side store all in the AXDB.

In order to retrieve faster the XML data needed, the XALAN C++ parser for Xpath resolution has been adopted.

The process is formalized by the following sequence diagram.



This approach allows a fast loading of the library since it is loaded only once from the application server and the resource is never released.
The following picture depicts the relationships among the different part and the roles covered by both side of the application JAVA (hosted in Tomcat Application server) and C++ contained in a DLL wrapping the AXOM.



7.1.2 Loader

loader module offers basically the checkuot service, both in sync and async mode.
The following table summarizes the methods of the Loader WebService a short description of the functionalities.

Loader WebService	
Method	***Description***
checkout_sync	This method allows a synchronous checkout of a predefined version of an object that will be provided in a URI communicated to the webservice client. The response arrives when the operation will be completed. The file in the URI will be removed as soon as it is downloaded, and in any case after a predefined timeout
checkout _async	This method allows a synchronous checkout of a predefined version of an object that will be provided in a URI communicated to the webservice client. The response will arrive when the operation will be accepted and the result will be communicated to a Listener specified below. The file in the URI will be removed as soon as it is downloaded, and in any case after a predefined timeout

The definition of the web service in terms of WSDL can be found in Section 28.

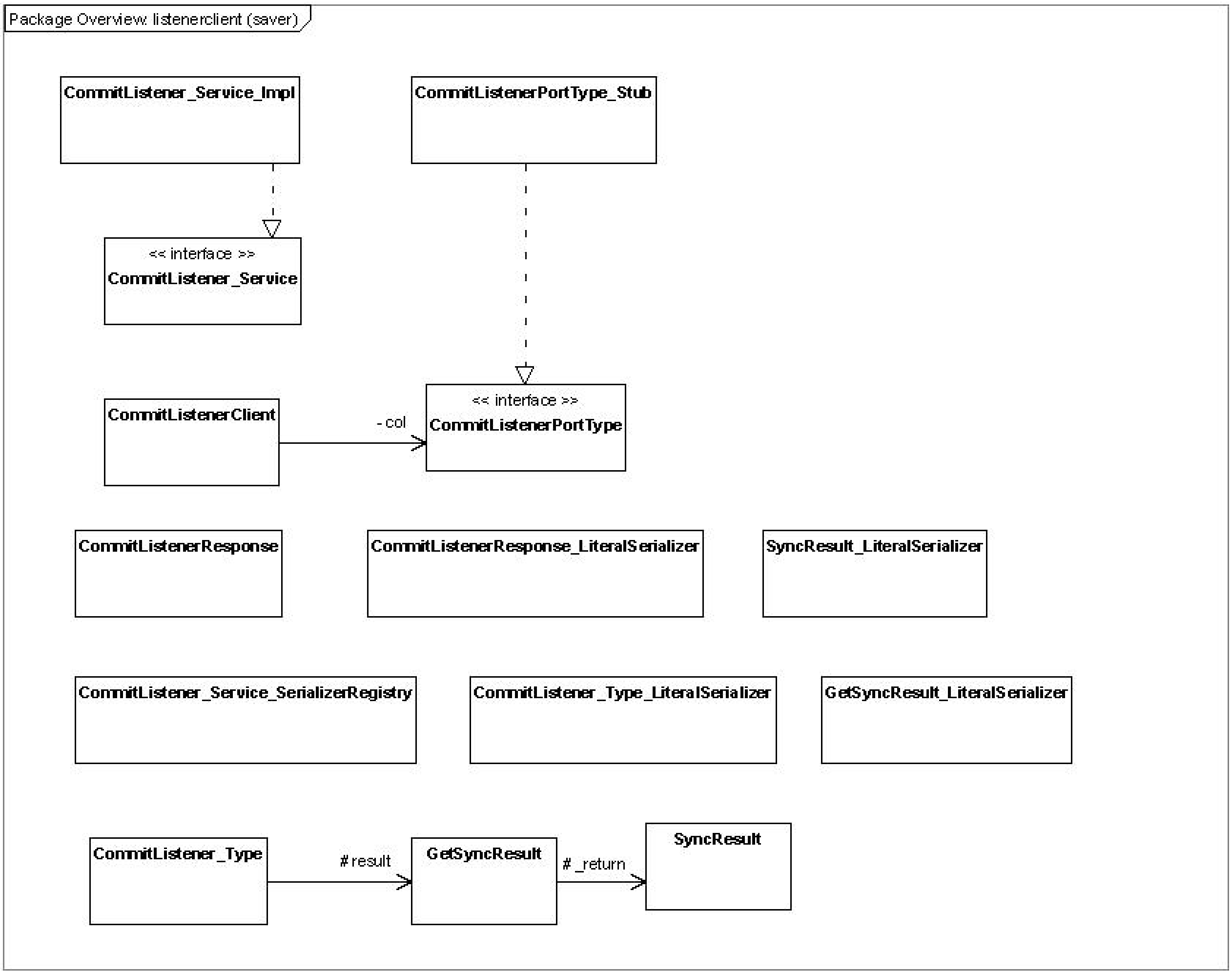
Since a listener is defined in the asynchronous mode, it is necessary to draw the specification also of this web service that will be offered and implemented by a third party that uses the asynchronous checkout. Refer to Section 29 for the definition in terms of WSDL.

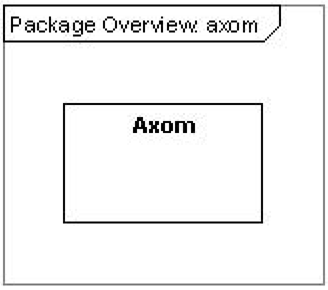
The main problem that drive the choice of language specification and technologies adoption is due to the fact that AXMEDIS Object Saver and Indexer must include an AXOM, that is written in C++. This implies that AXMEDIS Object Saver and Indexer itself must be written in C++.
The problem propagate in a cascade way to the modules that have a direct interface with this module that are:

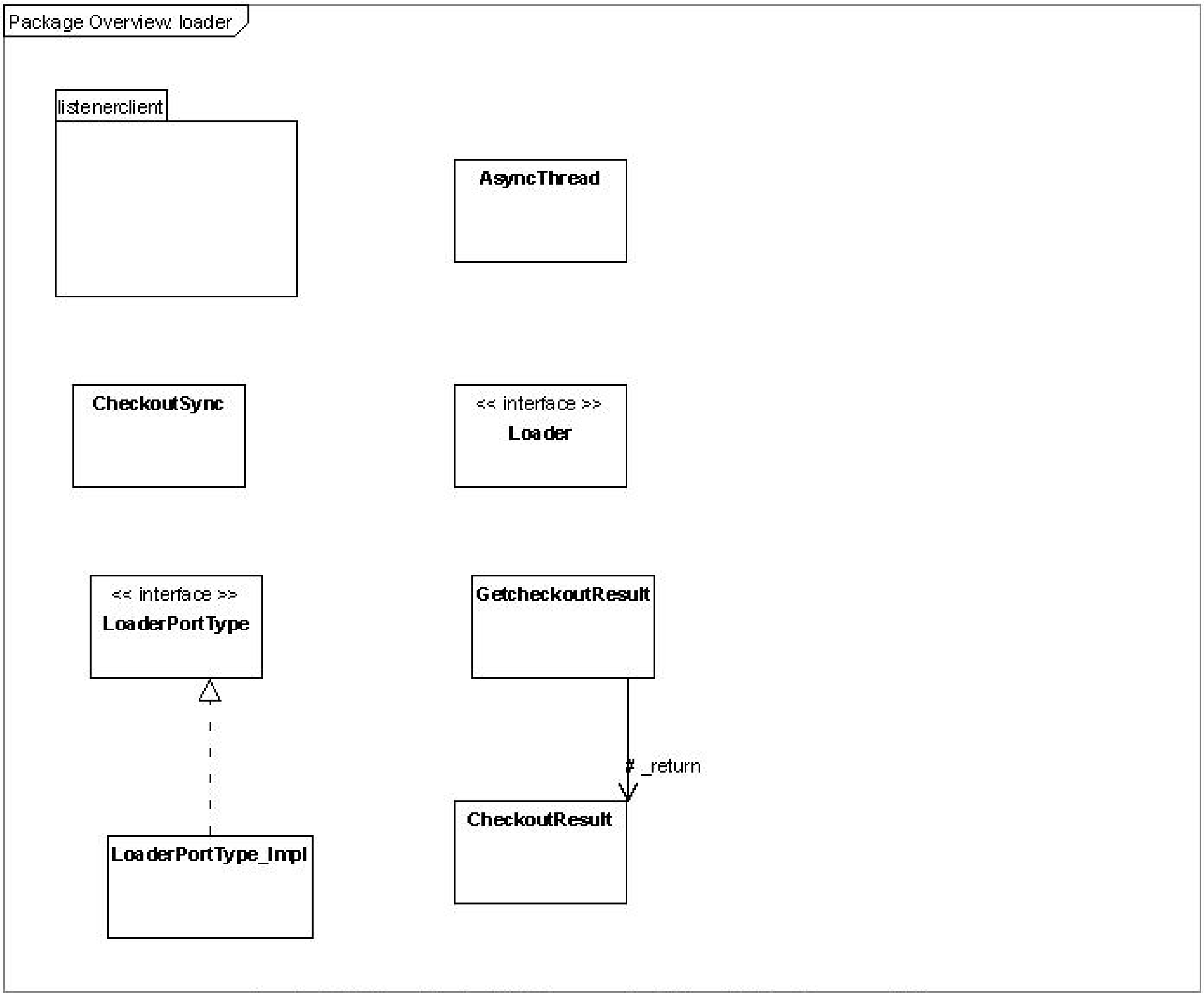
- AXMEDIS Database Interface
- AXMEDIS Save Web Service

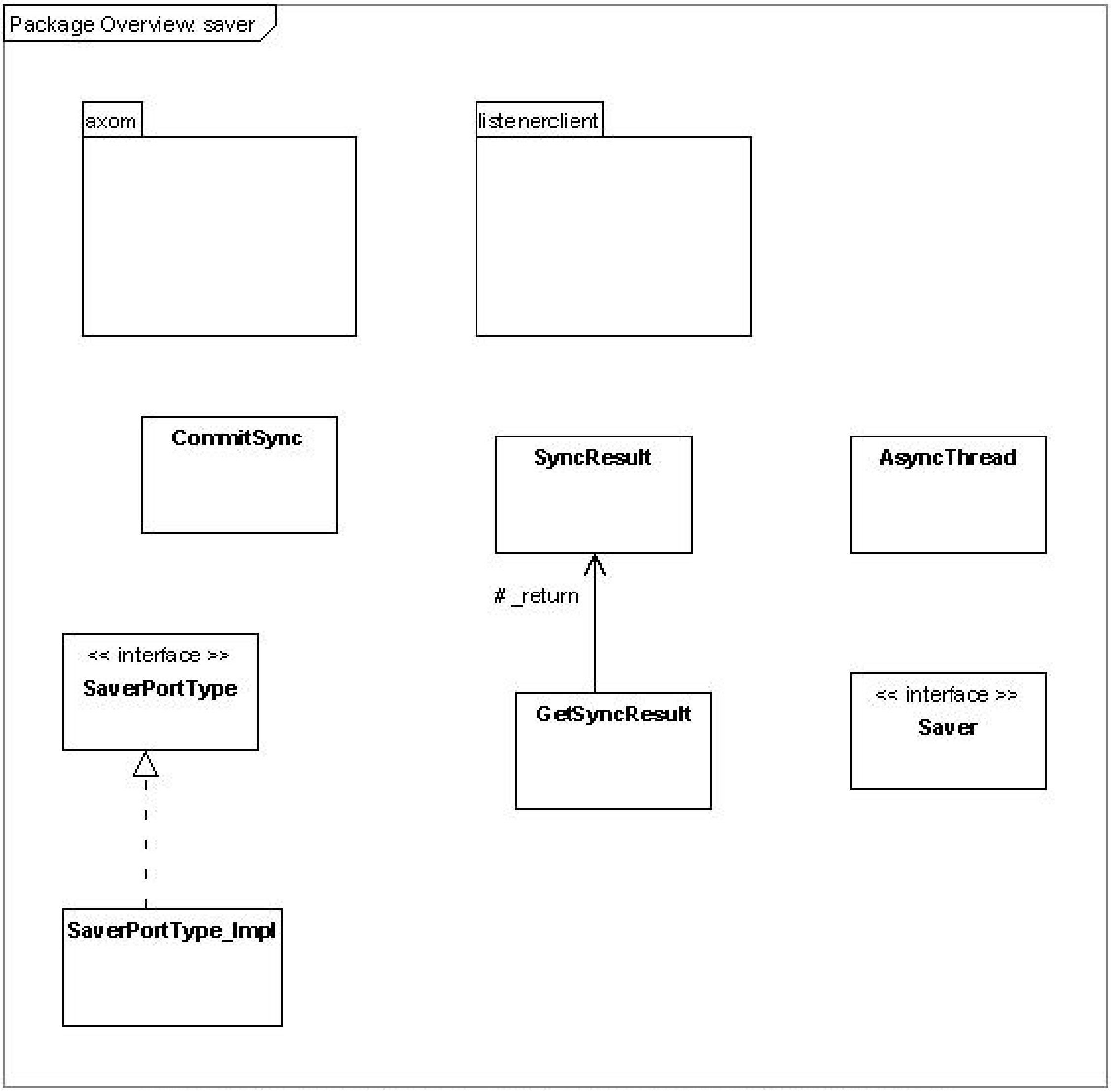
Since it is not feasible to write all the Database Interface in C++ for several reasons, it is better to implement all services in a JAVA package and export some methods also as a WebService, in order to allow modules that are not written in JAVA to easily communicate with the database.
It is better to implement the core part of the AXMEDIS Saves Web Service directly in C++ while top level part is realized in JAVA as the other web services.

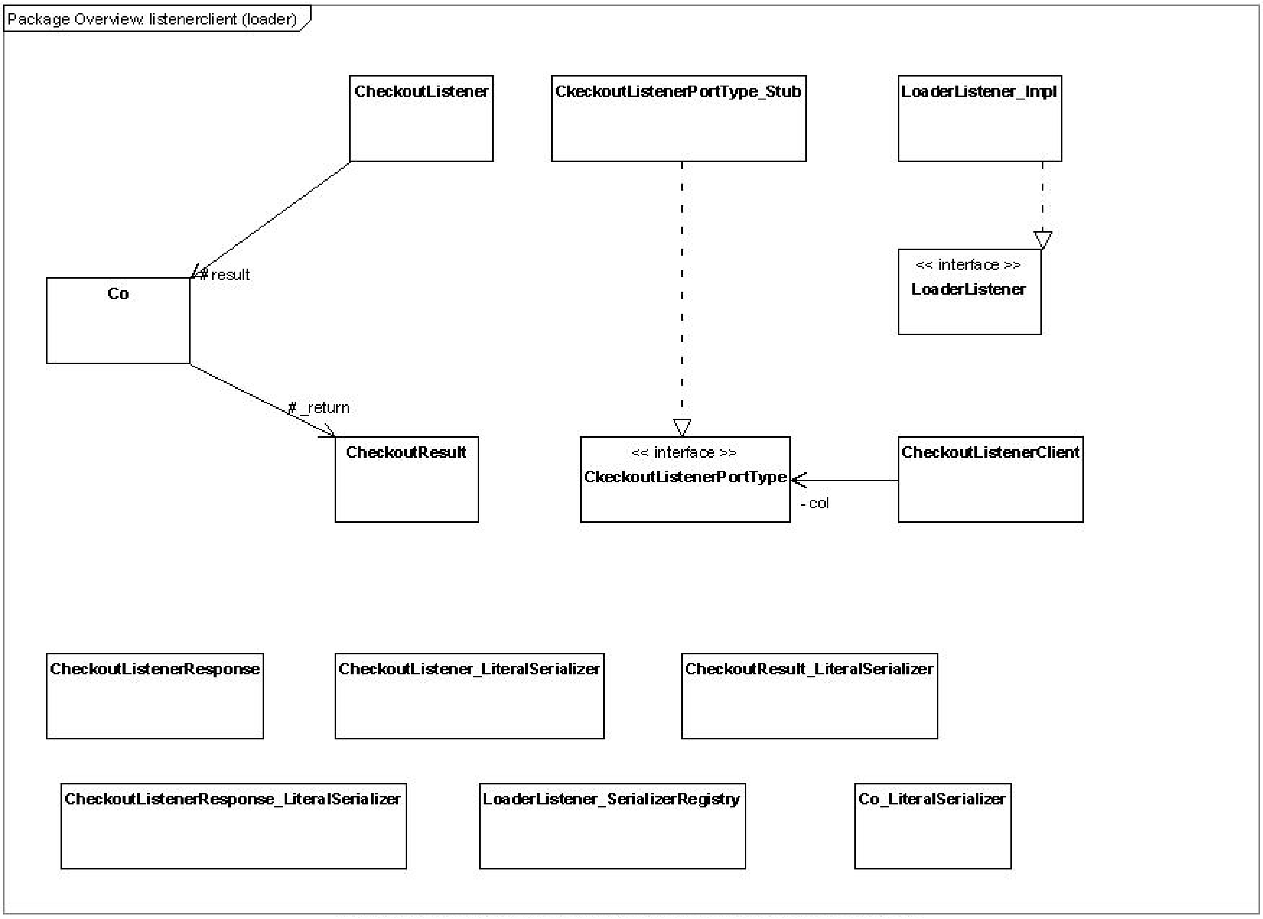
7.2 Module Design in terms of Classes


Package Overview: listenerclient (saver)
CommitListener_Service_Impl
CommitListenerPortType_Stub
<< interface >>
CommitListener_Service
CommitListenerClient
- col
<< interface >>
CommitListenerPortType
CommitListenerResponse
CommitListenerResponse_LiteralSerializer
SyncResult_LiteralSerializer
CommitListener_Service_SerializerRegistry
CommitListener_Type_LiteralSerializer
GetSyncResult_LiteralSerializer
CommitListener_Type
# result
GetSyncResult
# _return
SyncResult

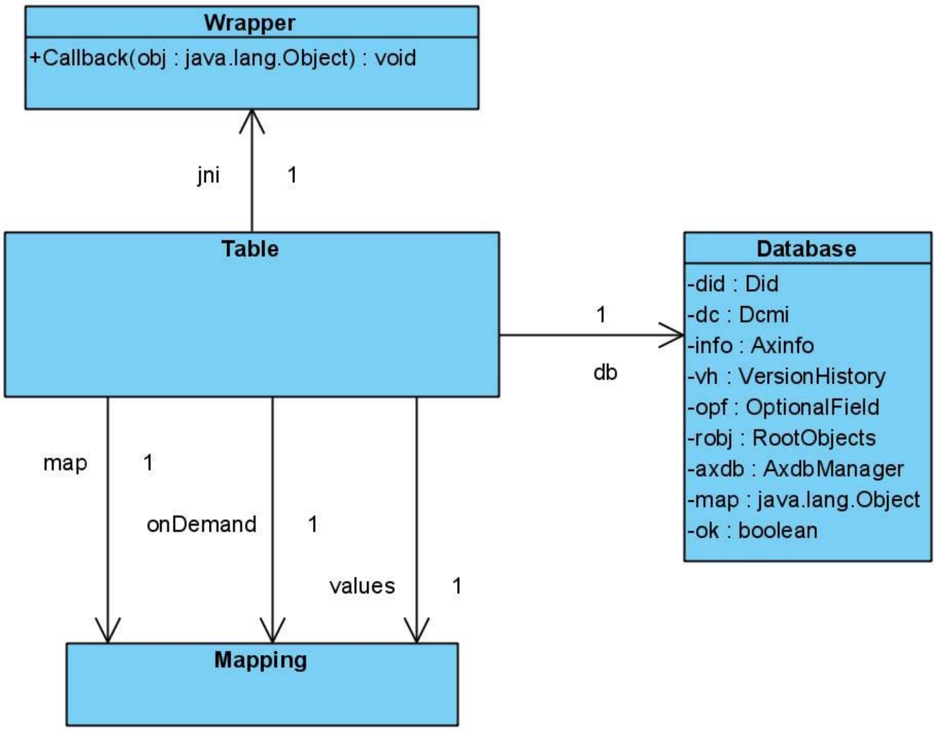

Package Overview: axom
Axom


Package Overview: loader
listenerclient
AsyncThread
CheckoutSync
<< interface >>
Loader
<< interface >>
LoaderPortType
GetcheckoutResult
# _return
LoaderPortType_Impl
CheckoutResult

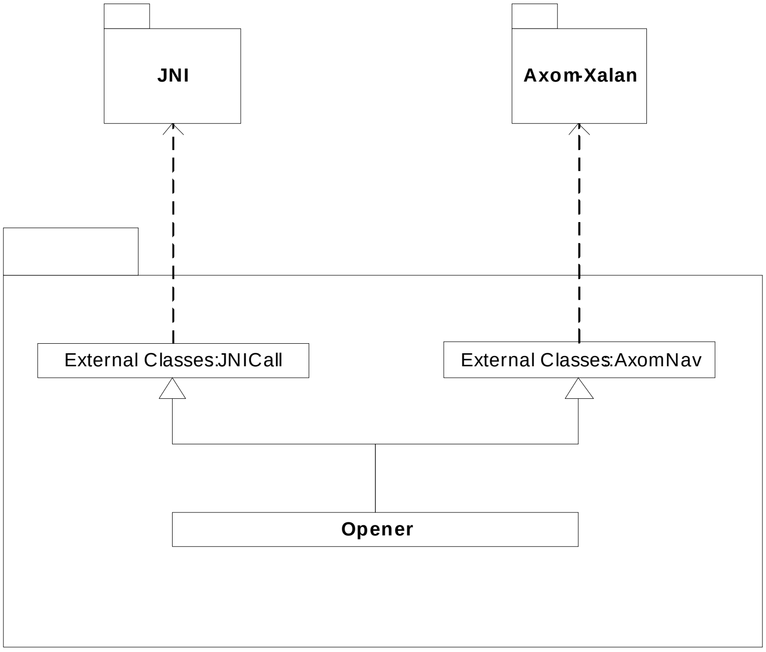

Package Overview: saver
axom
listenerclient
CommitSync
SyncResult
AsyncThread
# _return
<< interface >>
SaverPortType
GetSyncResult
<< interface >>
Saver
SaverPortType_Impl


Package Overview: listenerclient (loader)
CheckoutListener
CkeckoutListenerPortType_Stub
LoaderListener_Impl
<< interface >>
LoaderListener
Co
# result
# _return
CheckoutResult
<< interface >>
CkeckoutListenerPortType
- col
CheckoutListenerClient
CheckoutListenerResponse
CheckoutListener_LiteralSerializer
CheckoutResult_LiteralSerializer
CheckoutListenerResponse_LiteralSerializer
LoaderListener_SerializerRegistry
Co_LiteralSerializer

The application server creates a Table object that receives the AXMEDIS object to be analized. This object creates a temporary table that contains the metadata of the object obtained by the C++ parsing of the file.
When the parsing has to be started, the Table instance call a native C++ funciona named Callback of the Wrapper object.
The C++ code is is a DLL named jniwrap.dll and reads the metadata by using the function of the C++ libraries AXOM and XALAN.
The C++ side can insert in the temporary table the metadata values by the means of callback from C++ to the JAVA side.



The C++ wrapper once called create an instance of the Opener class that really parses the AXMEDIS object and insert data in the java side. The class axomNav is capable of navigating inside the object, while the capability of calling java operations is in the JNICall class, and therefore Opener inheriting from both has all the necessary features to perform the operations.



7.3 User interface description

This module has no user interface since it is a webservice

7.4 Technical and Installation information

In the following a draft installation manual for the web service is provided

7.4.1 Loader/Saver Web Service

In this section it is explained how to install and configure the Loader/Saver Web Service

7.4.2 Prerequisites

In this manual is it assumed that you install all in 1 WINDOWS server and that in this server you have:

- Mysql 4.1.10 or higher but not mysql 5.x installed and working
- Tomcat 5.5.x installed and working and respond on port 8080
- Apache (or IIS) installed and working and respond on port 80
- You have an FTP server installed on your server and the directory that is the root of the ftp service is $ROOTFTP
- Your machine has an IP address in your lan $IP_INTERNAL (say 192.168.1.81) and an internal DNS name $DNS_INTERNAL. If your server is visible from outside take note of the public ip $IP_PUBLIC and of the public dns name associated to the IP $DNS_PUBLIC

It is assumed also that:

- your Tomcat is installed in $TOMCAT (say C:\Programmi\Apache Software Foundation\Tomcat 5.5, or something like that);
- the document root of your web server is located in $HTDOCS (say C:\Programmi\Apache Group\Apache\htdocs, or something similar);
- your local copy of the AXMEDIS SVN repository is in $SVNROOT;
- you have got from the SVN repository the following files:
  - $SVNROOT\ Framework\doc\other\axdb\Mysql-4.1.10\axdbv4-clean.sql
  - $SVNROOT\\WebServices\LoadeSaverWs\lib\Axom.zip
  - $SVNROOT\\WebServices\QuerySupportWS\doc\test\Carl-Brisson-v2.axm
  - $SVNROOT\\WebServices\QuerySupportWS\doc\test\query.xml
  - $SVNROOT\WebServices\AxdbQSWs\bin\AXDBQuerySupportWs.war
  - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\axdb-mapping-v-1-4.xsd
  - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\LicenceWs.war
  - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\metadata-mapper.xml
  - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\QUERY-v1-6.xsd
  - $SVNROOT\WebServices\AxdbQSWs\lib\Tomcat-shared-libs.zip
  - $SVNROOT\WebServices\AxdbQSWs\lib\mysql-connector-java-3.1.7-bin.jar
  - $SVNROOT\WebServices\AxdbQSWs\lib\log4j-1.2.14.jar
  - $SVNROOT\WebServices\LoadeSaverWs\bin\LoadSaveWS.war
  - $SVNROOT\WebServices\LockUnlockWs\bin\LockUnlockWS.war
  - $SVNROOT\WebServices\QuerySupportWS\bin\MainQuerySupportWS.war
  - $SVNROOT\WebServices\QuerySupportWS\doc\test\CrawlerWS.war
  - $SVNROOT\WebServices\QuerySupportWS\doc\test\ExternalQuerySupportWS.war
  - $SVNROOT\WebServices\QuerySupportWS\doc\test\P2PWS.war
  - $SVNROOT\WebServices\QuerySupportWS\doc\test\QSClient
- Put all the libraries in the $SVNROOT\\WebServices\LoadeSaverWs\lib\Axom.zip in a Windows directory that is in the windows system path (or add the newly created directory to the existing path). In this directory identify all the jniwrap*.dll and select one for usage. Usually niwrap_version_F.dll is used for a factory environment, jniwap_version_VERSION.dll for a distribution environment and jniwrap_version_P2P.dll for a P2P environment.
- you have created under $HTDOCS the directories:
  - axmedis

If you made different assumption follow the guidelines and substitute the proposed path and names with the ones you chose.

7.4.3 Installation of AXMEDIS Database (AXDB)

See section 5.4.2

7.4.3.1 Web Service installation and configuration

If you have followed all the prerequisites you do not need to modify the axdb.properties file in **LoadSaveWS.war/WEB-INF/classes** directory by setting the correct values for axdbUrl, axdbUser, axdbPwd, that are the URL of the database, the user allowed to access to the AXDB and its password.
After that you have to edit axmpath.properties in the same directory and for the remote prefix property you have to the set place where the axm folders are stored, that is http://localhost/axmedis
Save, update the WAR and edit WEB-INF/classes/savercore_saver.properties and put in AxRepository property the path of your local WWW root repository followed by axmedis (say C:\inetpub\wwwroot\axmedis, or something similar).
Now we have to modify the other parameters but it is better to explain before what these parameters have been set for.
You can save an object in three different manners:

1) by selecting an axm file on your local system by identifying the file as file:///C:/somewhere/foo.axm
2) by giving to saver web service a remote http URL as http://myhost/mydir/foo.axm
3) by uploading on saver machine ftp server an axom file (say in ftp://user@server/upload/foo.axm) and calling the web service after the completion with the same filename ftp://user@server/upload/foo.axm

The system has to understand if you have uploaded the file on the ftp server that is local or if you have specified a remote ftp location (not supported at the moment). As stated in the prerequisites, your ftp server has an IP address in your lan $IP_INTERNAL (say 192.168.1.81) and an internal DNS name $DNS_INTERNAL (say mymachine.exitech.local). If your server is visible from outside you will have also its public ip $IP_PUBLIC and the public dns name associated to the IP $DNS_PUBLIC.
The parameters have to be set to the following:
FtpRepositoryIP=$IP_PUBLIC
FtpRepositoryDN=$DNS_PUBLIC
FtpLocalRepositoryIP=$IP_INTERNAL
FtpLocalRepositoryDN=$DNS_INTERNAL

After that you have to select the jniwarp dll that this instance of loader saver has to use and to put its name without .dll extension in the WEB-INF/classes/savercore_static.properties

In order to configure the loader web service you have to change the WEB-INF/classes/ws_axmpath.properties

An example of the file is reported below:

```
# Sample ResourceBundle properties file

#local prefix where the file are stored locally
localprefix=C\:\\Inetpub\\wwwroot\\axmedis

#remote prefix to be returned to the caller of Loader WebService
remoteprefix=http\://your-host-ip-or-name/axmedis/
```

localprefix is where in your installation the AXMEDIS files are put, while remoteprefis is how to have access to these files from external computer with trailing slash.

Close the WAR and deploy the newly created war in Tomcat.
By issuing http://localhost:8080/LoadSaveWS/load in your browser you should have something similar to the following table.

Web Services

Port Name	Status	Information
SaverPort	ACTIVE	Address: http://localhost:8081/LoadSaveWS/save WSDL: http://localhost:8081/LoadSaveWS/save?WSDL

		Port QName: {http://www.axmedis.org/saver.wsdl}SaverPortTypePort Remote interface: it.exitech.axmedis.ws.saver.SaverPortType Implementation class: it.exitech.axmedis.ws.saver.SaverPortType_Impl Model: http://localhost:8081/LoadSaveWS/save?model
CommitPort	ACTIVE	Address: http://localhost:8081/LoadSaveWS/commitlist WSDL: http://localhost:8081/LoadSaveWS/commitlist?WSDL Port QName: {http://www.someone.org/listener.wsdl}CommitListener Remote interface: it.exitech.axmedis.ws.commitlistener.CommitListenerPortType Implementation class: it.exitech.axmedis.ws.commitlistener.CommitListenerPortType_Impl Model: http://localhost:8081/LoadSaveWS/commitlist?model
LoaderPort	ACTIVE	Address: http://localhost:8081/LoadSaveWS/load WSDL: http://localhost:8081/LoadSaveWS/load?WSDL Port QName: {http://www.axmedis.org/loader.wsdl}LoaderPortTypePort Remote interface: it.exitech.axmedis.ws.loader.LoaderPortType Implementation class: it.exitech.axmedis.ws.loader.LoaderPortType_Impl Model: http://localhost:8081/LoadSaveWS/load?model
CheckuotPort	ACTIVE	Address: http://localhost:8081/LoadSaveWS/ws/CheckuotPort WSDL: http://localhost:8081/LoadSaveWS/ws/CheckuotPort?WSDL Port QName: {http://www.someone.org/listener.wsdl}CkeckoutListenerPortTypePort Remote interface: it.exitech.axmedis.ws.loaderlistener.CkeckoutListenerPortType Implementation class: it.exitech.axmedis.ws.loaderlistener.CkeckoutListenerPortType_Impl Model: http://localhost:8081/LoadSaveWS/ws/CheckuotPort?model

7.4.3.2 Test of loader/saver webservice

In order to test the loader/saver WebService you can deploy the axdb-administrative.war application that allows you to save an axm file.
Put a test.axm file in the c:\ of the system that is hosting your saver and by using the axdb-administrative store it in the database using as the file path file:///c:/test.axm
The result of the saving will be shown on the web page and if all is OK, then your saver service is OK.
In order to test loader you need the axoid of the object that can be obtained from the DB with the following SQL statement or read direcyly inside the object

```
SELECT axoid
FROM `did`
ORDER BY version DESC
```

Use the load object function of axdb-administrative to verify if the loader configuration has been completed correctly.

7.5 Draft User Manual

No user manual is provided since it is a backoffice product that requires only installation and configuration hints reported above and below this section.

7.6 Examples of usage

Example of usages are provided in the installation manual in section 7.4

7.7 Integration and compilation issues

Loader and saver web service uses the AXOM library to open the object. AXOM library is wrapped inside a DLL that is used together with other DLL by the web Service. Regarding the JAVA part the services are compatible with JDK 1.5 and installable as they are in Tomcat 5.5 after the configuration steps reported in section 7.4

7.8 Configuration Parameters

All the configuration parameters have been explained in section 7.4.3.1

8 Protection Models for AXMEDIS object Repository (UPC)

Module/Tool Profile		
Protection Model for AXMEDIS object Repository		
Responsible Name	Rubén Barrio	
Responsible Partner	UPC	
Status (proposed/approved)	Approved	
Implemented/not implemented	Implemented	
Status of the implementation	Completed, under maintenance	
Executable or Library/module (Support)	Library	
Single Thread or Multithread	Single Thread	
Language of Development	C++	
Platforms supported	Windows	
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/Framework/licensemodel	
Reference to the AXFW location of the demonstrator executable tool for internal download		
Reference to the AXFW location of the demonstrator executable tool for public download		
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any		
Test cases (present/absent)		
Test cases location		
Usage of the AXMEDIS configuration manager (yes/no)	no	
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved		
Major pending requirements		
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
PMSServer		
Formats Used	Shared with	format name or reference to a section
	DRMEditor&Viewer	

Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
AxmedisLicenses		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not
Xerces		
WxWidgets		
Mysql++		

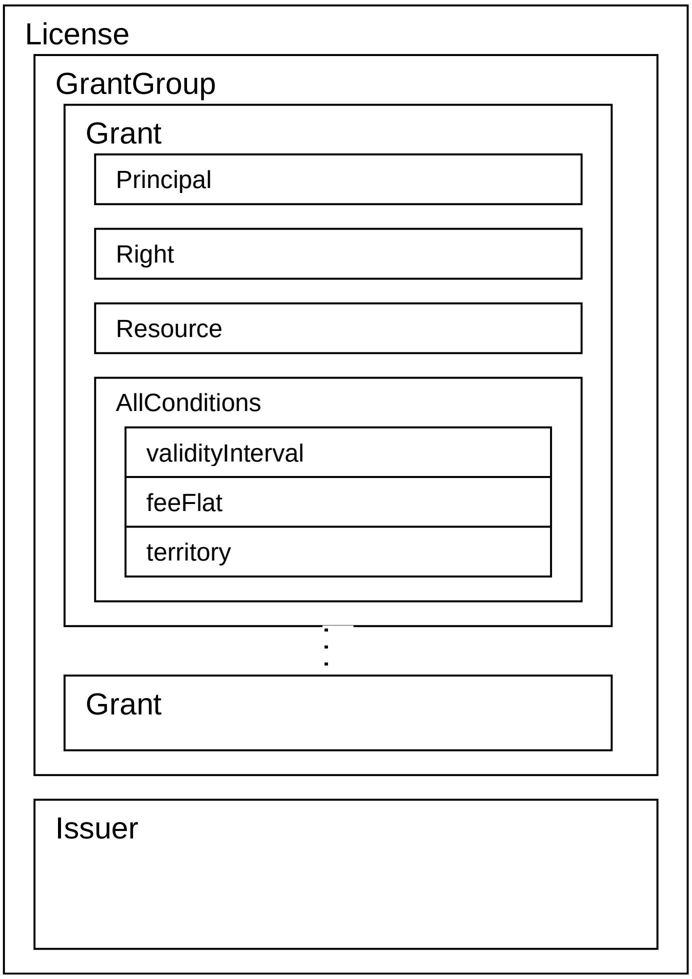
8.1 General Description of the Module

8.1.1 From REL licenses to ER model

In order to express DRM rules associated to AXMEDIS objects it has been decided to use MPEG-21 REL as primary rights expression language. Nevertheless, this is not the only possible language that can be used for expressing DRM rules, as stated in the DoW and the research guidelines document. We have planned to consider also ODRL (Open Digital Rights Language) as a language for expressing licenses.

For this reason, we propose a simple but scalable solution for expressing licenses into the UML and ER diagrams explained in more detail in next sections. This structure will simplify moving from one REL language to another (for instance from MPEG-21 REL to ODRL), expressing as many conditions as we want and also adding new conditions, if they appear.

The approach we have followed is to impose a common structure for licenses in order to simplify the parsing from the XML-based license into the ER structure. This structure for a final user license is shown in the next figure. Distributor licenses follow a similar structure.



REL License structure

For expressing the different types of conditions we can have, we have defined the following information:

- ConditionType: It indicates which kind of condition we are expressing.
- Five Tvalue fields and two NValue fields (more can be added if desired), whose values depend on the conditionType. See section “Information inside the conditions table” for details.

Using this structure we can easily define new conditions and implement support classes in the corresponding programming language depending on the condition.

We can also make complex queries over the defined ER diagram, only asking for different ConditionType, TvalueN and NvalueN.

Other possible approach for expressing conditions could have been to define a different table for each condition. We have discarded this approach, as conditions expressed in MPEG-21 REL have a lot of different possible and optional values, and making queries over such a structure could be even more complex than using only one table.

8.1.1.1 UML diagram for Licenses

The model that we are proposing allows storing the licenses in a relational database. All MPEG-21 REL Licenses can be stored in the database once parsed properly. With this model we are no restricting the syntactic structure of the licenses, but we need to transform them in a common format in order to facilitate searches.

The figure in this section shows the UML class diagram that we are proposing for representing the licenses. We explain the diagram next.

Each *license* can have an *AXLID* element that contains a unique identifier for the license, one or more *issuer* element(s) and a *GrantGroup*, a *status* element contains the status of the license, e.g. revoked, a *substLic* element that contains the license that replaces the revoked one, and a *inventory* element that contains the variables defined in the license that can be referenced within this license.

Each *GrantGroup* contains a set of *Grants* and the *forAll* element where the variables or patterns are defined within this GrantGroup are placed.

Each *Grant* contains the information of the *right* granted, the *resource*, the *principal* and an optional set of *conditions* related to that right, and the *forAll* element where the variables or patterns are defined within this grant are placed.

In addition, we have to realise that a *resource* can be a *GrantGroup* (in case of Distributor Licenses).

8.1.2 Proposal of PAR description based on REL information

Possible Available Rights (PAR) are a simplified version of licenses, as they provide the information of all the possible rights that a user has over an object.

We propose an UML diagram and an ER diagram for representing them, as described in more detail in next sections.

8.1.3 UML diagram for PAR

The figure shows the UML class diagram that we are proposing for representing the PAR. We explain it next.

Each *PAR* can have a *PARID* element that contains a unique identifier for the PAR, a *GrantGroup*, a *status* element contains the status of the license, e.g. valid, a *licensingURL* element that contains the reference to the service / distributor site that can issue a license for this PAR, if any, and a *inventory* element that contains the variables defined in the PAR that can be referenced within this PAR.

Each *GrantGroup* contains a set of *Grants* and the *forAll* element where the variables or patterns are defined within this GrantGroup are placed.

Each *Grant* contains the information of the *right* granted, the *resource* and an optional set of *conditions* related to that right, and the *forAll* element where the variables or patterns are defined within this grant are placed.

In addition, we have to realise that a *resource* can be a *GrantGroup* (in case of distribution of PAR, that is, licenses).

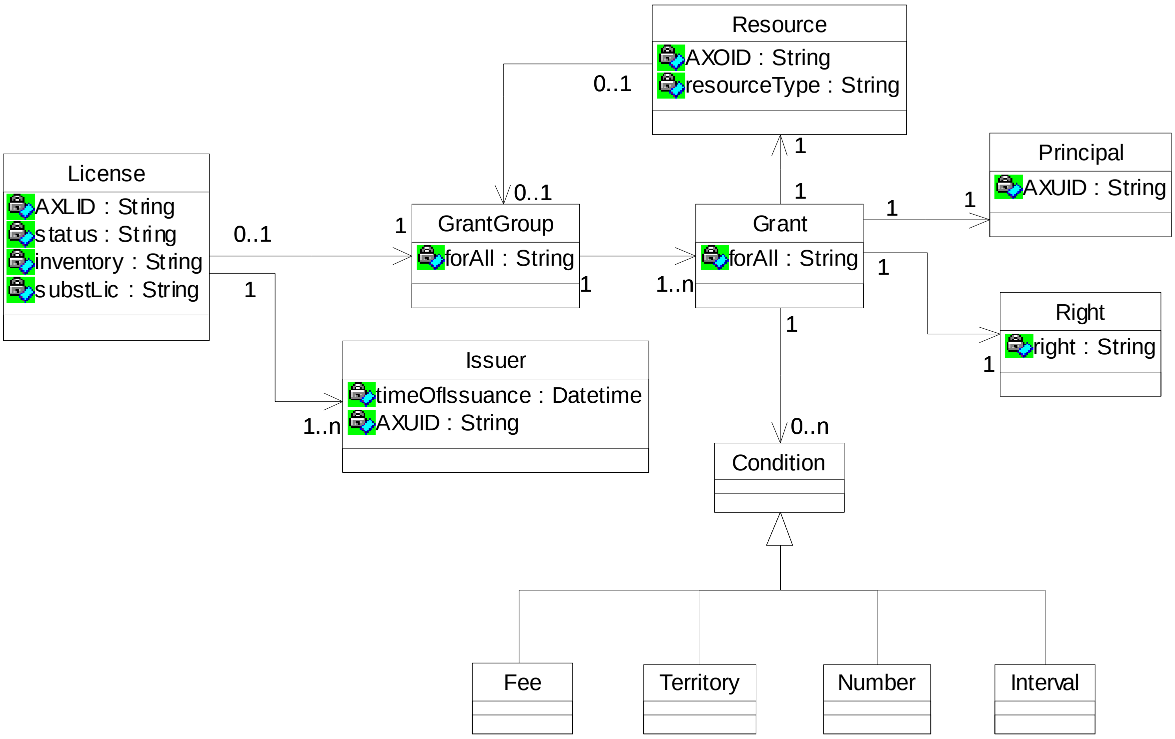
8.1.3.1 Pricing for Final users

The pricing of a given action that a final user wants to perform can be expressed in several places: PAR, distribution license and final user license.

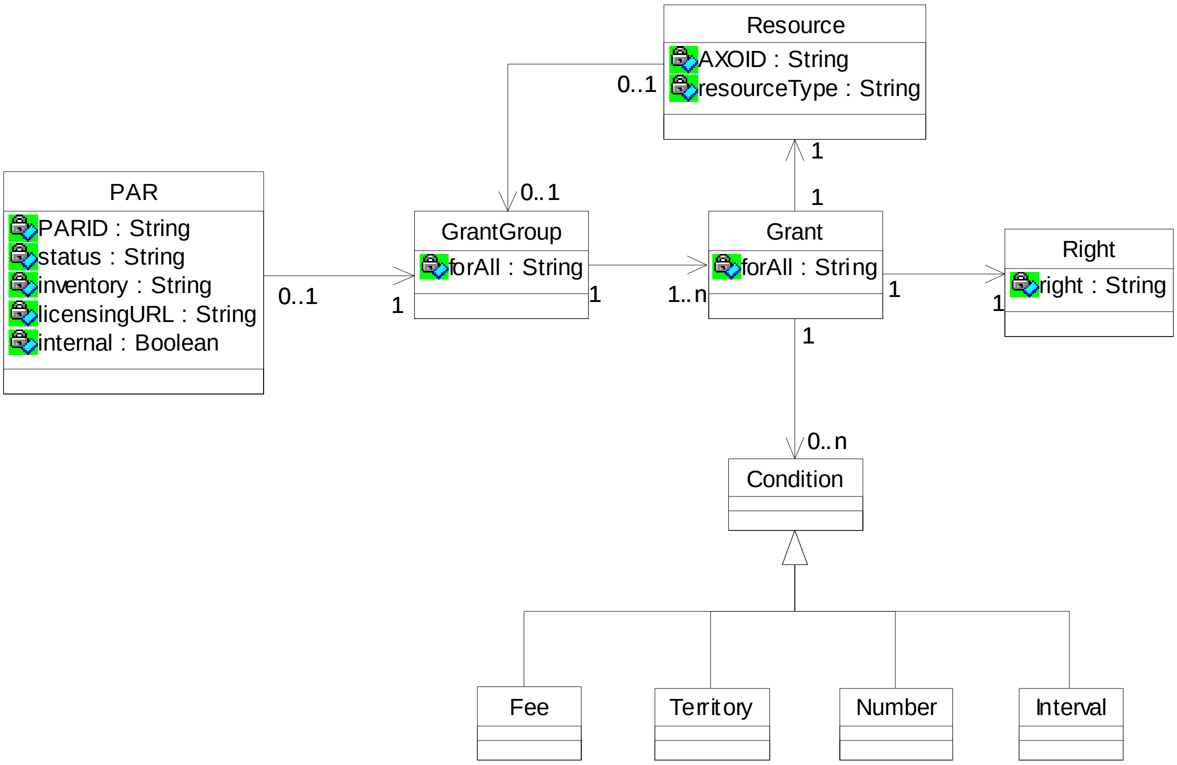
Depending on this, the user can be informed through the PMS client that the right he wants to exercise will cost X euros, before buying the right or before exercising it after he has bought it.

8.2 Module Design in terms of Classes

In this section, the UML diagram for supporting licenses and PAR is shown.



UML diagram for licenses



UML diagram for PAR

8.3 Examples of usage

8.3.1.1 Example of information inside the License Database

In this section we present two examples of MPEG-21 REL licenses, for final users and distributors. Then, we present how this license information is stored in the different tables.

Lic1: Example of final user license

```
<?xml version="1.0" encoding="UTF-8"?>
<r:license xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS" xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-NS"
xmlns:sx="urn:mpeg:mpeg21:2003:01-REL-SX-NS" xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:mpeg:mpeg21:2003:01-REL-R-NS rel-
r.xsd urn:mpeg:mpeg21:2003:01-REL-SX-NS rel-sx.xsd urn:mpeg:mpeg21:2003:01-REL-MX-NS rel-mx.xsd">
    <r:grantGroup>
        <r:grant>
            <r:keyHolder>
                <r:info>
                    <dsig:KeyValue>
                        <dsig:RSAKeyValue>
                            <dsig:Modulus>KtdToQQyzA==</dsig:Modulus>
                            <dsig:Exponent>AQABAA==</dsig:Exponent>
                        </dsig:RSAKeyValue>
                    </dsig:KeyValue>
                </r:info>
            </r:keyHolder>
            <mx:play/>
            <mx:diReference>
                <mx:identifier>urn:a1-AXOID1-f</mx:identifier>
            </mx:diReference>
            <r:allConditions>
                <r:validityInterval>
                    <r:notBefore>2005-01-01T01:00:00</r:notBefore>
                    <r:notAfter>2005-04-01T01:00:00</r:notAfter>
                </r:validityInterval>
                <sx:exerciseLimit>
                    <r:serviceReference>
                        <sx:uddi>
                            <sx:serviceKey>
                                <sx:uuid>ee1398c0-8abe-11d7-a735-b8a03c50a862</sx:uuid>
                            </sx:serviceKey>
                        </sx:uddi>
                    </r:serviceReference>
                    <sx:count>150</sx:count>
                </sx:exerciseLimit>
            </r:allConditions>
        </r:grant>
        <r:grant>
            <r:keyHolder>
                <r:info>
                    <dsig:KeyValue>
                        <dsig:RSAKeyValue>
                            <dsig:Modulus>KtdToQQyzA==</dsig:Modulus>
                            <dsig:Exponent>AQABAA==</dsig:Exponent>
                        </dsig:RSAKeyValue>
                    </dsig:KeyValue>
                </r:info>
            </r:keyHolder>
            <mx:adapt/>
            <mx:diReference>
                <mx:identifier>urn:a1-AXOID1-f</mx:identifier>
            </mx:diReference>
            <r:allConditions>
                <sx:feeFlat>
                    <r:serviceReference>
                        <sx:uddi>
                            <sx:serviceKey>
                                <sx:uuid>D04951E4-332C-4693-B7DB-D3D1D1C20844</sx:uuid>
                            </sx:serviceKey>
                        </sx:uddi>
                    </r:serviceReference>
                    <sx:rate>
                        <sx:amount>5.00</sx:amount>
                        <sx:currency>iso:EUR</sx:currency>
                    </sx:rate>
                    <sx:to>
                        <sx:paymentService>
                            <serviceReference>
                                <sx:uddi>
                                    <sx:serviceKey>
```

```
                                        <sx:uuid>D04951E4-332C-4693-B7DB-D3D1D1C20844</sx:uuid>
                                    </sx:serviceKey>
                                </sx:uddi>
                            </serviceReference>
                        </sx:paymentService>
                    </sx:to>
                </sx:feeFlat>
            </r:allConditions>
        </r:grant>
    </r:grantGroup>
    <r:issuer>
        <r:keyHolder>
            <r:info>
                <dsig:KeyValue>
                    <dsig:RSAKeyValue>
                        <dsig:Modulus>X0j9q99yzA==</dsig:Modulus>
                        <dsig:Exponent>AQABAA==</dsig:Exponent>
                    </dsig:RSAKeyValue>
                </dsig:KeyValue>
            </r:info>
        </r:keyHolder>
        <r:details>
            <r:timeOfIssue>2005-01-01T01:00:00</r:timeOfIssue>
        </r:details>
    </r:issuer>
</r:license>
```

Lic2: Example of distributor license

```
<?xml version="1.0" encoding="UTF-8"?>
<r:license          xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS"          xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-NS"
xmlns:sx="urn:mpeg:mpeg21:2003:01-REL-SX-NS"                        xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"    xsi:schemaLocation="urn:mpeg:mpeg21:2003:01-REL-R-NS    rel-
r.xsd urn:mpeg:mpeg21:2003:01-REL-SX-NS rel-sx.xsd urn:mpeg:mpeg21:2003:01-REL-MX-NS rel-mx.xsd">
    <r:grantGroup>
        <r:grant>
            <r:keyHolder>
                <r:info>
                    <dsig:KeyValue>
                        <dsig:RSAKeyValue>

    <dsig:Modulus>g8NRYMG307NqJgmZG8TlUOp+9sQjsAai+hlLpkBiLaf4RhvjS3pD0dvy1YosEjKL8mk/KTGniC+pY4ia5kLBy
Q==</dsig:Modulus>
                            <dsig:Exponent>AQABAA==</dsig:Exponent>
                        </dsig:RSAKeyValue>
                    </dsig:KeyValue>
                </r:info>
            </r:keyHolder>
            <r:issue/>
            <r:grantGroup>
                <r:grant>
                    <mx:play/>
                    <mx:diReference>
                        <mx:identifier>urn:a1-AXOID2-f</mx:identifier>
                    </mx:diReference>
                    <r:allConditions>
                        <sx:territory>
                            <sx:location>
                                <sx:country>iso:ES</sx:country>
                            </sx:location>
                        </sx:territory>
                        <sx:feePerUse>
                            <sx:rate>
                                <sx:amount>1.00</sx:amount>
                                <sx:currency>iso:EUR</sx:currency>
                            </sx:rate>
                            <sx:to>
                                <sx:paymentService>
                                    <serviceReference>
                                        <sx:uddi>
```

```
                                            <sx:serviceKey>
                                                <sx:uuid>D04951E4-332C-4693-B7DB-D3D1D1C20844</sx:uuid>
                                            </sx:serviceKey>
                                        </sx:uddi>
                                    </serviceReference>
                                </sx:paymentService>
                            </sx:to>
                        </sx:feePerUse>
                    </r:allConditions>
                </r:grant>
                <r:grant>
                    <mx:adapt/>
                    <mx:diReference>
                        <mx:identifier>urn:a1-AXOID2-f</mx:identifier>
                    </mx:diReference>
                    <r:allConditions>
                        <sx:exerciseLimit>
                            <r:serviceReference>
                                <sx:uddi>
                                    <sx:serviceKey>
                                        <sx:uuid>ee1398c0-8abe-11d7-a735-b8a03c50a862</sx:uuid>
                                    </sx:serviceKey>
                                </sx:uddi>
                            </r:serviceReference>
                            <sx:count>3</sx:count>
                        </sx:exerciseLimit>
                    </r:allConditions>
                </r:grant>
            </r:grantGroup>
        </r:grant>
    </r:grantGroup>
    <r:issuer>
        <r:keyHolder>
            <r:info>
                <dsig:KeyValue>
                    <dsig:RSAKeyValue>
                        <dsig:Modulus>X0j9q99yzA==</dsig:Modulus>
                        <dsig:Exponent>AQABAA==</dsig:Exponent>
                    </dsig:RSAKeyValue>
                </dsig:KeyValue>
            </r:info>
        </r:keyHolder>
        <r:details>
            <r:timeOfIssue>2005-01-01T01:00:00</r:timeOfIssue>
        </r:details>
    </r:issuer>
</r:license>
```

Example of values inside Licenses database:

Licenses		
AXLID	**Status**	**TimeOfIssuance**
Lic1	Valid	2005-01-01T01:00:00
Lic2	Valid	2005-01-01T01:00:00

Issuers	
AXLID	**AXUID**
Lic1	Issuer1
Lic2	Issuer2

GrantGroups	
AXLID	**GrantGroupID**
Lic1	GG1
Lic2	GG1

Lic2	GG2

Grants									
AXLID	**Grant Group ID**	**GrantID**	**Right**	**ResourceType**	**Resource**	**GrGrID_Res**	**Principal**	**ResType**	**ResSubType**
Lic1	GG1	Gr1	Play	Resource	AXOID1	- null -	Prin1	1	0
Lic1	GG1	Gr2	Adapt	Resource	AXOID1	- null -	Prin1	1	0
Lic2	GG1	Gr1	Issue	GrantGroup	- null -	GG2	Prin2	1	0
Lic2	GG2	Gr1	Play	Resource	AXOID2	- null -	- null -	1	0
Lic2	GG2	Gr2	Adapt	Resource	AXOID2	- null -	- null -	1	0

Conditions								
AXLID	**GrantGroupID**	**GrantID**	**ConditionID**	**ConditionType**	**Tvalue1**	**Tvalue2**	**Tvalue3**	**Nvalue1**
Lic1	GG1	Gr1	Con1	ValidityInterval	2005-01-01T01:00:00	2005-04-01T01:00:00		
Lic1	GG1	Gr1	Con2	exerciseLimit	Service_Referece1.xml			150
Lic1	GG1	Gr2	Con1	feeFlat	Service_Referece2.xml	iso:EUR	TO_1.xml	5
Lic2	GG2	Gr1	Con1	territory	<sx:country>iso:ES</sx:country>			
Lic2	GG2	Gr1	Con2	feePerUse		iso:EUR	TO_2.xml	1
Lic2	GG2	Gr2	Con1	exerciseLimit	Service_Referece1.xml			3

The fields *ServiceReference* and *To* contain an XML node as described in MPEG-21 REL. These fields are not used in searches, so we don’t need to extend the information within them to other fields in the table. But, we need the information to perform an authorisation.

8.4 Formal description of LicenseDBManager

This module is completely described in DE 3.1.2.2.14, as LicenseManager.

9 PAR Manager (UPC)

Module/Tool Profile	
PAR Manager	
Responsible Name	Rubén Barrio
Responsible Partner	UPC
Status (proposed/approved)	Approved
Implemented/not implemented	Implemented
Status of the implementation	Completed, under maintenance
Executable or Library/module (Support)	Executable
Single Thread or Multithread	Single Thread
Language of Development	C++
Platforms supported	Windows
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/WebServices/PARManager
Reference to the AXFW location of the demonstrator executable tool for internal download	
Reference to the AXFW location of the demonstrator executable tool for public download	
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	
Test cases (present/absent)	
Test cases location	
Usage of the AXMEDIS configuration manager (yes/no)	no
Usage of the AXMEDIS Error Manager (yes/no)	no
Major Problems not solved	
Major pending requirements	

Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
PMSServer		
Formats Used	Shared with	format name or reference to a section
Protocol Used	Shared with	Protocol name or reference to a

		section
Used Database name		
AxmedisLicenses		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not
Xerces		
WxWidgets		
Mysql++		

9.1 General Description of the Module

PAR Manager provides the interface for creation and management of Potential Available Rights in AXMEDIS. The PARs are stored in the corresponding database for searching purposes. The PARQuerySupport will consult the PAR database to get the results of the user queries.

The PAR Manager is a Web Service with three basic operations, this web service has not SSL support because this service is inside a private network and it is used by registered members.

The metods of the PARManager are:

PAR Manager	
Methods	***Description***
getPAR	Retrieves a PAR from the PAR Manager
removePAR	Deletes a PAR from the database
sendPAR	Sends a PAR to the server, here is verified

9.2 Proposal of PAR description based on REL information

Possible Available Rights (PAR) are a simplified version of licenses, as they provide the information of all the possible rights that a user has over an object.

We propose an UML diagram and an ER diagram for representing them, as described in more detail in next sections.

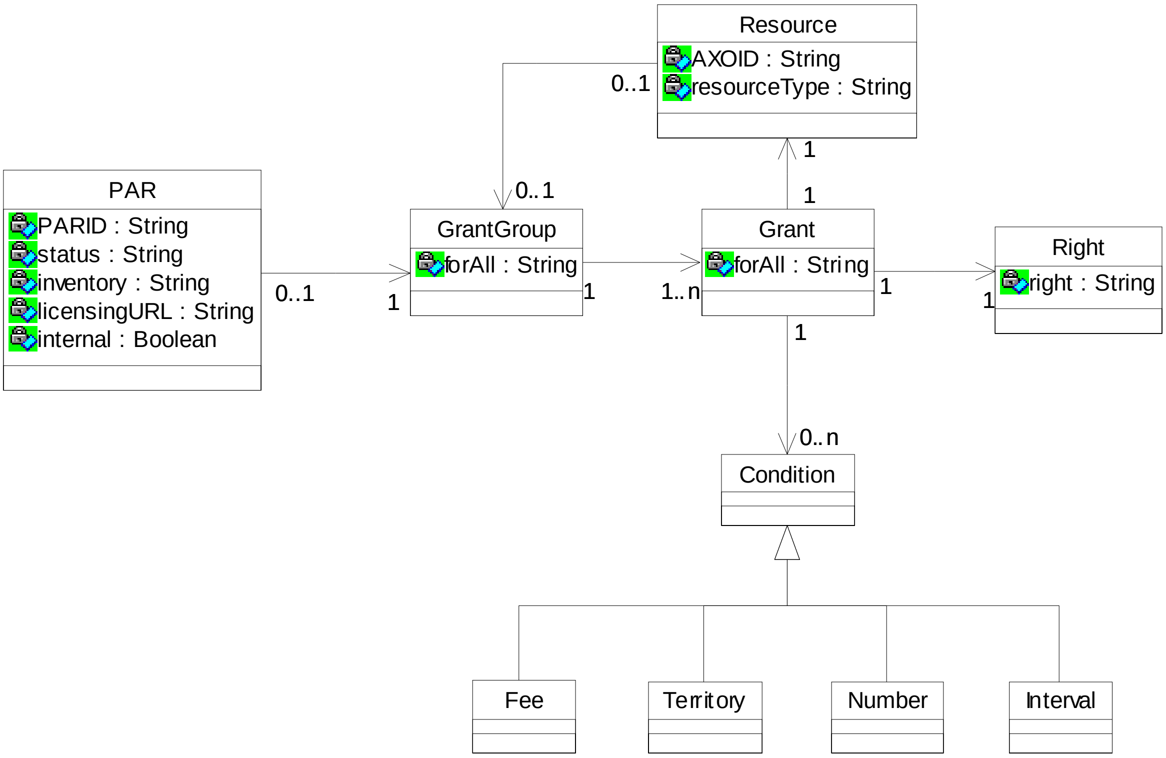
9.3 UML diagram for PAR

The figure shows the UML class diagram that we are proposing for representing the PAR. We explain it next.
Each *PAR* can have a *PARID* element that contains a unique identifier for the PAR, a *GrantGroup*, a *status* element contains the status of the license, e.g. valid, a *licensingURL* element that contains the reference to the service / distributor site that can issue a license for this PAR, if any, and a *inventory* element that contains the variables defined in the PAR that can be referenced within this PAR.

Each *GrantGroup* contains a set of *Grants* and the *forAll* element where the variables or patterns are defined within this GrantGroup are placed.

Each *Grant* contains the information of the *right* granted, the *resource* and an optional set of *conditions* related to that right, and the *forAll* element where the variables or patterns are defined within this grant are placed.

In addition, we have to realise that a *resource* can be a *GrantGroup* (in case of distribution of PAR, that is, licenses).



UML diagram for PAR

9.4 Instalation Process

Create a new database in MYSQL with the par.sql file. This SQL script is provided in this document in the following sections.

Create a user in the MYSQL database that have access to make selects and inserts in the created database.

Modify the licman.ini file to provide the parameters for the services

```
host=DatabaseServerIP
pardatabase=Name of the database created
user=Username for the MYSQL
password=Password for the MYSQL username
bindaddress= IP address in which the PARManager will be listening
port=Port in which the PARManager will be listening
```

Example:

```
host=193.145.45.173
pardatabase=axmedispar
user=axmedis
password=axmedis
bindaddress=193.145.45.225
port=8585
```

Now you can Run PARManager.exe, but take care of having in the ./schemas the MPEG-21 Schemas for making the verification of PARS.

9.5 Operation Details

PAR Manager	
Method	getPAR
Description	This function retrieves the PAR stored in the PAR database. It retrieves the PAR with the AXOID set as a parameter.
Input parameters	String AXOID: AXOID
Output parameters	String, the PAR in XML

PAR Manager	
Method	sendPAR
Description	This function stores a PAR in the PAR database.
Input parameters	String PARXML: the PAR in XML format String AXOID: the PAR AXOID
Output parameters	String: result of the operation

PAR Manager	
Method	removePAR
Description	This function removes a PAR from PAR database.
Input parameters	String AXOID: the AXOID of the PAR that will be erased
Output parameters	String: result of the operation

9.6 SQL for creating the PAR Database

```
CREATE DATABASE /*!32312 IF NOT EXISTS*/ `axmedispar`;
USE `axmedispar`;

DROP TABLE IF EXISTS `licenses`;
CREATE TABLE `licenses` (
  `AXLID` varchar(100) NOT NULL default '',
  `Status` varchar(50) default NULL,
  `SubstLic` varchar(100) default NULL,
  `Inventroy` varchar(100) default NULL,
  `LicenseXML` text,
  `TimeOfIssuance` varchar(50) default NULL,
  PRIMARY KEY  (`AXLID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `grantgroups`;
CREATE TABLE `grantgroups` (
  `AXLID` varchar(100) NOT NULL default '',
  `GrantGroupID` varchar(100) NOT NULL default '',
  `forAll` text,
  PRIMARY KEY  (`AXLID`,`GrantGroupID`),
  CONSTRAINT `FK_GG1` FOREIGN KEY (`AXLID`) REFERENCES `licenses` (`AXLID`) ON
DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `grants`;
CREATE TABLE `grants` (
  `AXLID` varchar(100) NOT NULL default '',
  `GrantGroupID` varchar(100) NOT NULL default '',
  `GrantID` varchar(100) NOT NULL default '',
  `RightName` varchar(50) default NULL,
  `ResourceType` varchar(25) default NULL,
  `AXOID` varchar(250) default NULL,
  `GrGrID_Res` varchar(100) default NULL,
  `forAll` text,
  `ResType` int(10) unsigned default '0',
  `ResSubType` int(10) unsigned default '0',
  PRIMARY KEY  (`AXLID`,`GrantGroupID`,`GrantID`),
  KEY `AXLID` (`AXLID`,`GrGrID_Res`),
  CONSTRAINT `FK_G1` FOREIGN KEY (`AXLID`, `GrantGroupID`) REFERENCES
`grantgroups` (`AXLID`, `GrantGroupID`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `FK_G2` FOREIGN KEY (`AXLID`, `GrGrID_Res`) REFERENCES
`grantgroups` (`AXLID`, `GrantGroupID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `conditions`;
CREATE TABLE `conditions` (
  `AXLID` varchar(100) NOT NULL default '',
  `GrantGroupID` varchar(100) NOT NULL default '',
  `GrantID` varchar(100) NOT NULL default '',
  `ConditionID` varchar(100) NOT NULL default '',
  `ConditionType` varchar(50) default NULL,
  `TValue1` varchar(1024) default NULL,
  `TValue2` varchar(1024) default NULL,
  `TValue3` varchar(1024) default NULL,
  `TValue4` varchar(1024) default NULL,
  `TValue5` varchar(1024) default NULL,
  `NValue1` float default NULL,
  `NValue2` float default NULL,
  `NValue3` float default NULL,
  `NValue4` float default NULL,
  `NValue5` float default NULL,
  PRIMARY KEY  (`AXLID`,`GrantGroupID`,`GrantID`,`ConditionID`),
  KEY `ConditionType` (`ConditionType`),
  CONSTRAINT `FK_C1` FOREIGN KEY (`AXLID`, `GrantGroupID`, `GrantID`) REFERENCES
`grants` (`AXLID`, `GrantGroupID`, `GrantID`) ON DELETE CASCADE ON UPDATE
CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS `parameters`;
CREATE TABLE `parameters` (
  `ID` int(11) NOT NULL default '0',
  `LicenseID` int(11) default NULL,
  PRIMARY KEY  (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `parameters` (`ID`,`LicenseID`) VALUES
 (1,1);
```

10 History of AXMEDIS Objects (EXITECH)

Module/Tool Profile		
History of AXMEDIS Objects		
Responsible Name	Fioravanti	
Responsible Partner	EXITECH	
Status (proposed/approved)	approved	
Implemented/not implemented	Implemented	
Status of the implementation	Completed, under maintenance	
Executable or Library/module (Support)	---	
Single Thread or Multithread	Multithread	
Language of Development	JAVA	
Platforms supported	All supported by JDK 1.5	
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/Framework/source/axdb/	
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/repos/Framework/bin/axdb/	
Reference to the AXFW location of the demonstrator executable tool for public download		
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any		
Test cases (present/absent)	absent	
Test cases location		
Usage of the AXMEDIS configuration manager (yes/no)	no	
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved	-- --	
Major pending requirements	-- --	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section

Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

10.1 General Description of the Module

This module has no sense without the general database interface and it is consider as separated only to focus the attention on problems related to versioning in the large.

For all the AXMEDIS objects it is necessary to track and store different versions of each object. Object can differ for several aspects. The main aspects are:

- Content itself that has changed (the DIDL is not modified but the external reference is a different object;
- Part of the content (also the DIDL can be modified if a new Item is added for example);
- Metadata associated to the object (i.e. DIDL is the same, but AXINFO is changed or other metadata have been modified).

AXMEDIS has to track all version and has to be capable of having immediately available the last version of the object for indexing and for fast retrieval, and has to be capable of retrieving one of the previous version of the object, also if not indexed in the database.

It is necessary, in the Database Interface, to have methods for listing versions, recovering different versions of an object and to update objects generating a new version.

Moreover it is necessary also to lock and unlock the objects/versions in the database in order to allow people to work on an item to produce something new.

It is necessary to distinguish between the different situation in which a version is updated:

1. The content of the object has been modified, but apart from the external reference no other change has been introduced: in this case a new version is created with reference to the new content, while the previous object (full AXMEDIS object) with the previous external references are tracked in the system;

2. Part of the content and at least the DIDL are changed: the object is imported in the system main database in order to have the indexing of the new object online. The old version is stored together with the external references;
3. The AXINFO or metadata have been changed: the new metadata are mapped in the DB and the old object is stored together with the old external references.

The problem related to the name of duplicated external references has to be considered, since if I have for example the content FlorencePicture.jgp that is part of one AXMEDIS object and I have updated such object with a new version of the FlorencePicture.jgp (maybe different in resolution, filtering or whatever) keeping the same name, it is necessary to be able to distinguish between these versions in order to give back the correct object.

In the resource is embedded in the XML object the problem do not exists since the whole object is stored together with versioning information related to the versions that are not the current.

It is possible to create a hierarchical structure on the disc in order to store the physical contents; such structure can be very simply implemented by having a main directory with an unique ID (i.e the AXOID) that in the database is referenced to the content with a set of subdirectories in which for each version only the new version of the content are stored; if a version is missing in the hierarchy it means that for that version no update of the content has been made.

```
AXOID ------ 1 ----- FlorencePicture.jpg
           |--- 1 ----- FlorencePicture.jpg
```

In general all objects are stored externally and we have to take care of size of embedded object and also of time related to transfer of object for example for transcoding.

All the operations described in this section are in charge to Saver/Indexer module, that is the entry point to the database and therefore the best place where taking care of new versions added to the DB.

10.1.1 Locking service

As introduced above, it is several time necessary to lock objects in the database if we want to modify them.

The locking mechanism is somehow more complex since it has an aspects bound to versioning.

We can choose to lock a single version (in order to set it as “unavailable”) or to lock all the versions of an axoid to allow a modification of the version of the object.

The lock for a specific version is mainly useful for administrative purposes in order to avoid a load of an object that for some reasons (missing rights, problems, and so on) cannot be viewed.

When a single version is locked it cannot be loaded, and therefore load web service will return a specific error.

When an user wants to edit an object with the aim of creating a new version of the object, he/she MUST acquire a lock on the AXOID and therefore on all the versions and then can load one version (that is not “version-locked”) in order to generate a new version.

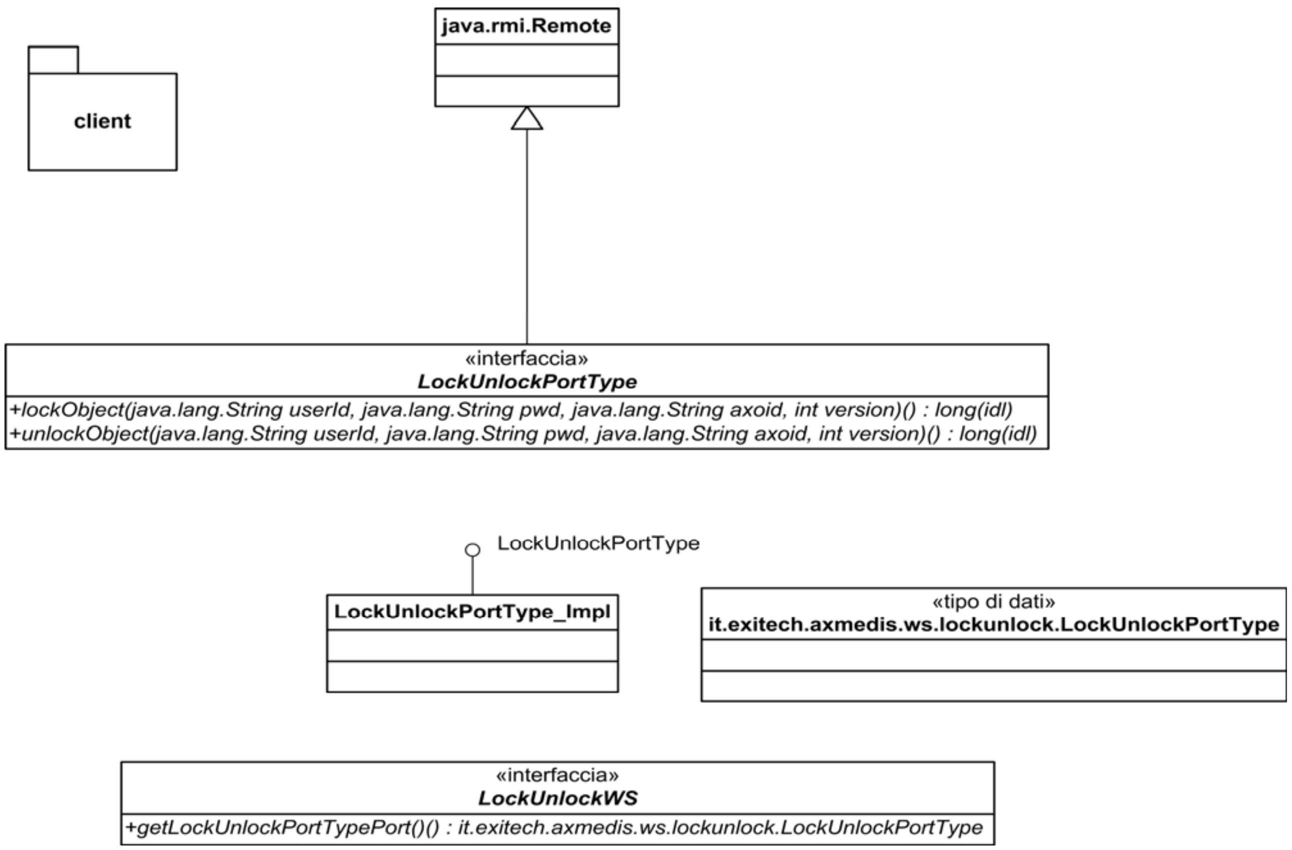
In order to summarize we have the following possibility:

- A specific version of an AXOID is locked: no one can load that version
- An user load a not locked version of an AXOID: the user cannot generate a new version
- An user acquire a lock on a version of an AXOID and load a not locked version: the user can generate a new version

To this end a webservice has been created that allow an user to lock and unlock an object. In order to lock/unlock the AXOID, it is necessary to user version=0 as a parameter,

10.2 Module Design in terms of Classes

Please refer to section 4.2 for all classes related to the versioning, while for classes the following diagram is the reference


client
java.rmi.Remote
«interfaccia»
LockUnlockPortType
+lockObject(java.lang.String userId, java.lang.String pwd, java.lang.String axoid, int version)() : long(idl)
+unlockObject(java.lang.String userId, java.lang.String pwd, java.lang.String axoid, int version)() : long(idl)
LockUnlockPortType
LockUnlockPortType_Impl
«tipo di dati»
it.exitech.axmedis.ws.lockunlock.LockUnlockPortType
«interfaccia»
LockUnlockWS
+getLockUnlockPortTypePort()() : it.exitech.axmedis.ws.lockunlock.LockUnlockPortType

10.3 User interface description

This module as all the database part has no user interface apart from that provided by AXMEDIS Web Administrative Interface

10.4 Technical and Installation information

The module is comprised inside the AXMEDIS database interface and all the issues related to the main module are still valid for this one. Moreover we have the locking service that is a web service whose WSDL is reported in section 37, that allows the user to lock and unlock objects in the AXDB. An object locked by an user can be released by the same user or by an administrator.

In this section it is reported a draft configuration manual for installing the service as it is provided by EXITECH in the SVN repository of the project.

10.4.1 Prerequisites

In this manual is it assumed that you install all in 1 WINDOWS server and that in this server you have:

- Mysql 4.1.10 or higher but not mysql 5.x installed and working
- Tomcat 5.5.x installed and working and respond on port 8080

It is assumed also that:

- your Tomcat is installed in $TOMCAT (say C:\Programmi\Apache Software Foundation\Tomcat 5.5, or something like that);
- your local copy of the AXMEDIS SVN repository is in $SVNROOT;
- you have got from the SVN repository the following files:
    - https://cvs.axmedis.org/repos/WebServices/LockUnlockWs/bin/Tomcat5.5/LockUnlockWS.war

If you made different assumption follow the guidelines and substitute the proposed path and names with the ones you chose.

10.4.2 Installation of AXMEDIS Database (AXDB)

See section 5.4.2

10.4.3 AXMEDIS Database Support Web Services

10.4.3.1 Installation of the WebServices

The installation of the web services identified in this section is very easy since the WAR file in https://cvs.axmedis.org/repos/WebServices/LockUnlockWs/bin/Tomcat5.5/LockUnlockWS.war can be deployed as it is in Tomcat 5.5.x after installing in $TOMCAT/shared/libs the libraries in $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\Tomcat-shared-libs.zip.

If the parameters have been set as stated in section 5.4.2, nothing has to be changed, otherways it is necessary to modify the axdb.properties file to reflect the configuration of the system as stated in section 5.8.1.

10.4.3.2 Verification of the Installation

If the installation and therefore the deployment has been correctly done, in the Tomcat Manager a screen similar to the following will appear:

Web Services

Port Name	Status	Information	
LockUnlockWSd	ACTIVE	Address:	http://localhost:8080/LockUnlockWS/lockunlock
		WSDL:	http://localhost:8080/LockUnlockWS/lockunlock?WSDL
		Port QName:	{http://www.exitech.it/axmedis/LockUnlockWS}LockUnlockPortTypePort
		Remote interface:	it.exitech.axmedis.ws.lockunlock.LockUnlockPortType
		Implementation class:	it.exitech.axmedis.ws.lockunlock.LockUnlockPortType_Impl
		Model:	http://localhost:8080/LockUnlockWS/lockunlock?model

10.5 Draft User Manual

No user manual apart from the specification of API already reported in AXMEDIS database interface sections.

10.6 Examples of usage:

The web service has two methods: the lock and the unlock. A call to the lock with a defined AXOID and version will put a lock flag on the file, so that if some one tries to issue a lock it will be refused. The same user that has issued the lock or an administrator can call the unlock method to release the lock.

10.7 Integration and compilation issues

The module is comprised inside the AXMEDIS database interface and all the issues related to the main module are still valid for this one.

10.8 Configuration Parameters

Config parameter	Possible values
`axdbUrl`	`jdbc:mysql://localhost/axdb-test?jdbcCompliantTruncation=false`
`axdbUser`	`axdbuser`
`axdbPwd`	`mkzamk`

11 AXMEDIS Query Support (EXITECH)

Module/Tool Profile		
AXMEDIS Query Support		
Responsible Name	Fioravanti	
Responsible Partner	EXITECH	
Status (proposed/approved)	approved	
Implemented/not implemented	implemented	
Status of the implementation	Completed, under maintenance	
Executable or Library/module (Support)	Web service	
Single Thread or Multithread	Multithread	
Language of Development	JAVA	
Platforms supported	All supported by JDK 1.5 and Tomcat 5.5	
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/WebServices/AxdbQSWs/source/ https://cvs.axmedis.org/repos/WebServices/QuerySupportWS/source/	
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/repos/WebServices/AxdbQSWs/bin/ https://cvs.axmedis.org/repos/WebServices/QuerySupportWS/bin/	
Reference to the AXFW location of the demonstrator executable tool for public download	none	
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	MainQuerySupport Endpoint: http://81.73.104.125:8080/MainQuerySupportWs/mqs WSDL: http://81.73.104.125:8080/MainQuerySupportWs/mqs?WSDL AXDBQuerySupport Endpoint: http://81.73.104.125:8080/AXDBQsWS/QuerySupport WSDL: http://81.73.104.125:8080/AXDBQsWS/QuerySupport?WSDL	
Test cases (present/absent)	present	
Test cases location	https://cvs.axmedis.org/repos/WebServices/QuerySupportWS/doc/test/	
Usage of the AXMEDIS configuration manager (yes/no)	no	
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved	-- --	
Major pending requirements	-- --	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section

Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

11.1 General Description of the Module

The query support is a combination of query mechanisms for retrieving content objects among the indexed information, in the AXMEDIS database (for content processing, composition, formatting, etc.), on the P2P AXEPTool, on the content collector referencing the content contained in the CMSs (Crawled Results Integrated Database). From the AXMEDIS database interface it has to be possible to make valid queries for all the environments and read unified results. From the AXEPTool or from the Collector the queries can be performed only on their environment.

The query support allows the specification of technical/professional query including metadata, technical information, business and licensing aspects, content based, DRM rules, etc.

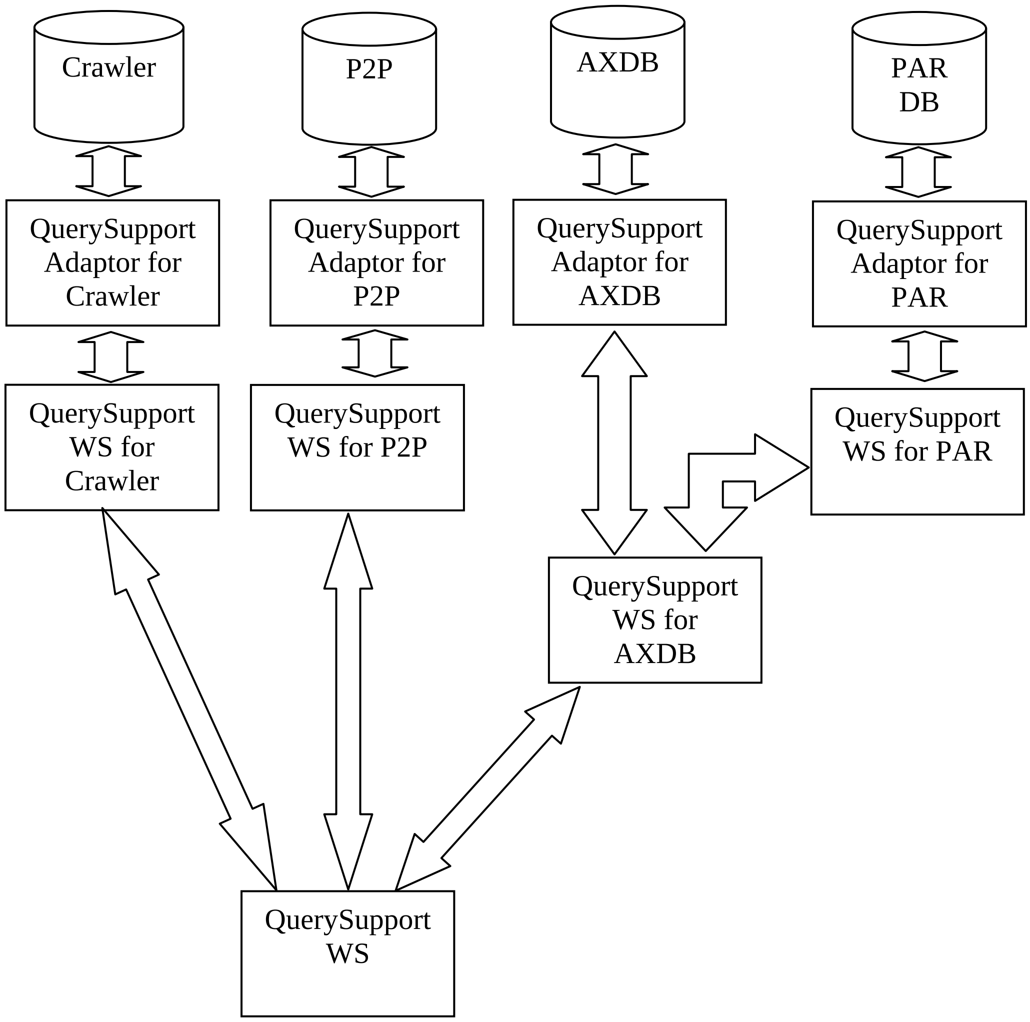
11.1.1 Query Support general architecture

In the architecture it is evidenced that the Query support offers a service to the other tools of the system by the means of a Web service that is able to receive query, distribute such queries on the different channels and recollecting aggregated data in order to communicate back the results.

The query Support Web Service Interface communicate with the backend of the AXDB for gathering information directly on the object stored in the AXMEDIS database, while access via a web service interface to the crawling system and to the AXEPTOOL. The web service interface and therefore the WSDL offered by the AXEPTOOL and by the Crawling system must be the same as that offered by Query Support in order to have a unique format for data exchanging.

The Query User Interface access via web service to the Query support in order to send query and gather information.

The Architecture for query support can be summarized at a functional level also according to the details reported in the following picture where the relationships among the different entities are formalized.
In particular it has been evidenced the relation between the AXDB QS WS and the DB of PAR and Internal PAR. The same relation can be applied also to the other components


Crawler
P2P
AXDB
PAR DB
QuerySupport Adaptor for Crawler
QuerySupport Adaptor for P2P
QuerySupport Adaptor for AXDB
QuerySupport Adaptor for PAR
QuerySupport WS for Crawler
QuerySupport WS for P2P
QuerySupport WS for PAR
QuerySupport WS for AXDB
QuerySupport WS

The general query support Web Service forwards the query to the sub-systems involved in the query. Each subsystem receive the query and P2P and AXDB also forward the same query to the PAR DB if the query involve also PAR.
The result from the subsystem is put in AND with the PAR results and sent back to QuerySupport WS. Query Support Web Service built a result with the OR of all the results and return back it to the user.

11.1.2 Query Format and language (EXITECH)

The query support interface with the rest of the AXMEDIS component is realized by the means of a WebService interface. This fact implies that the messages (that are the queries and the results of the queries) that are exchanged among the different AXMEDIS component must be in XML format.
To this end, in this section a basic schema for defining how to express a query and a query result is defined. These schemas will be the foundation also for the WebService WSDL.

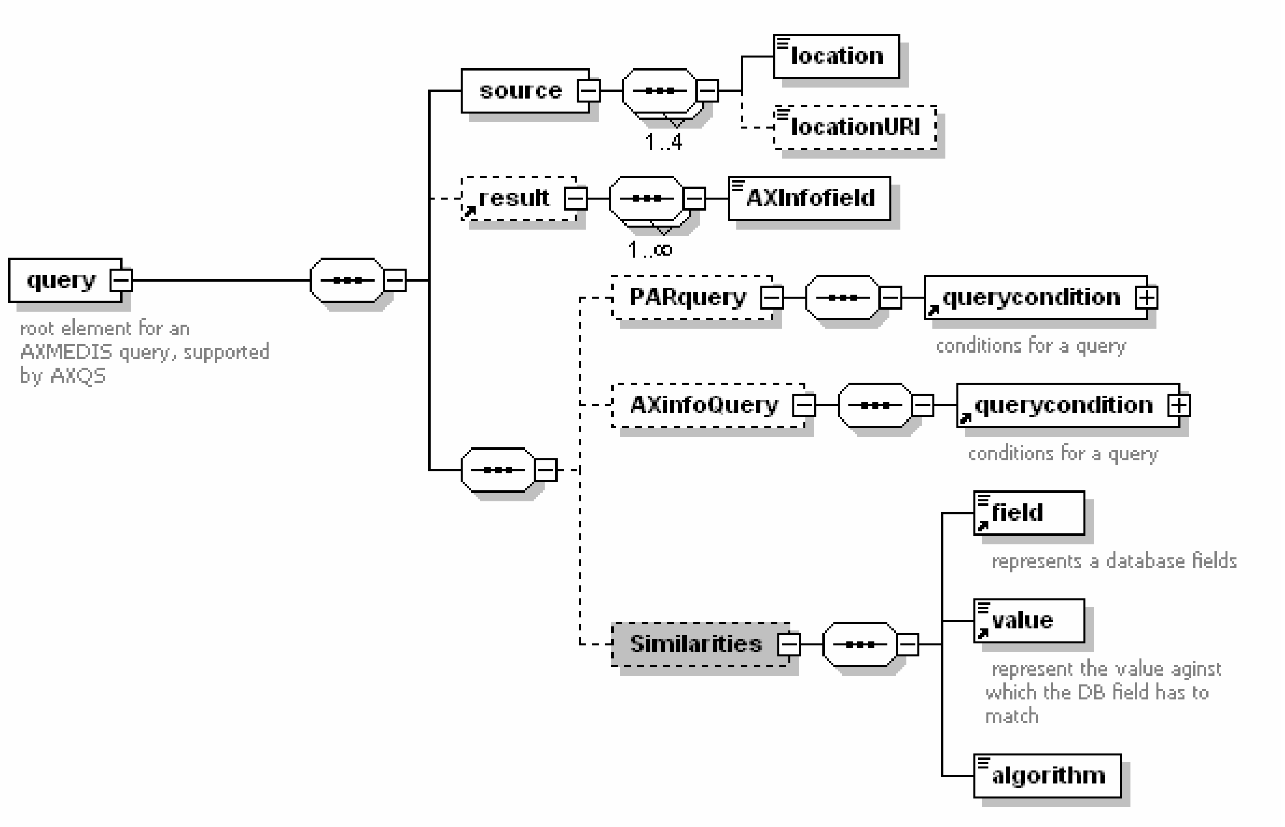
11.1.2.1 Schema for an AXMEDIS query

In this section the general schema for representing an AXMEDIS query is reported and commented.

The schema of the query is mainly composed by three parts. In the first part the location on which the query has to be distributed is reported, in the second parts a list of fields that the query should give as the result is contained and in the third and final part, the conditions under which a query has to be executed are imposed. The three different parts are separately commented after the textual and graphic representation of the Query schema in sections

The proposed schema is reported in section 21 while a more fashionable graphic format that can be useful to have a general overview with the related explanations are reported in this section.

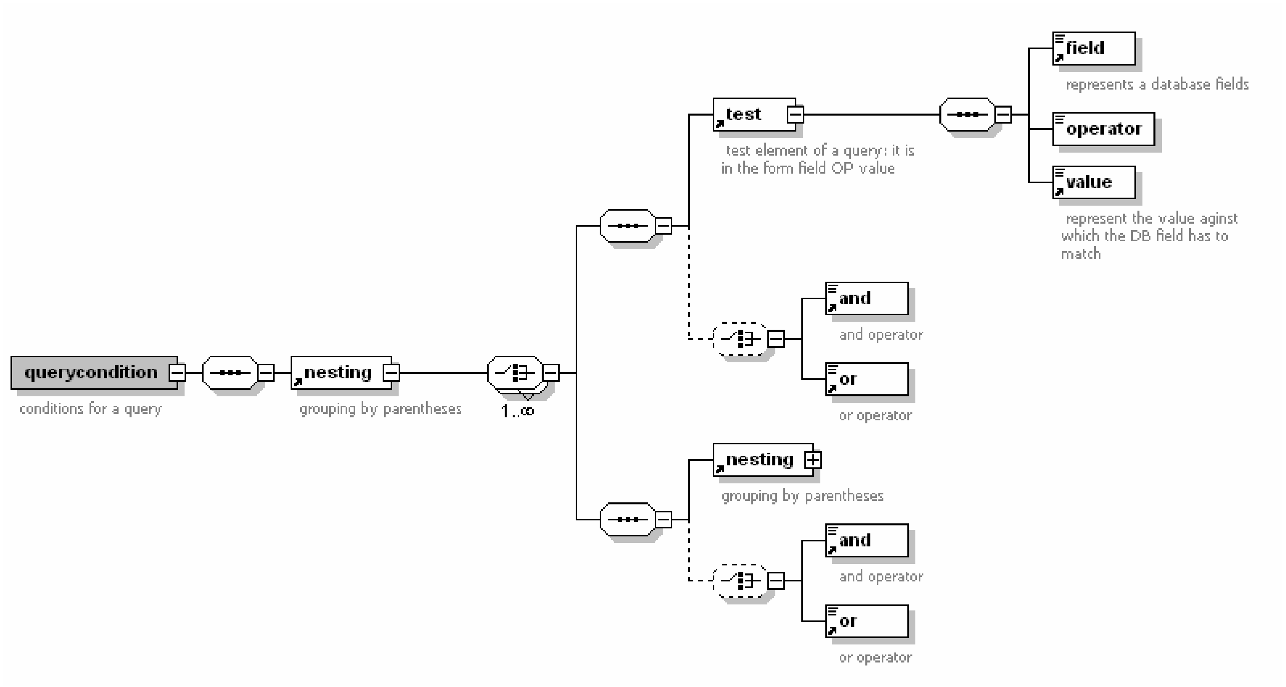
According to the last revision of the query model and of the way in which query can be submitted by the user, it can be noted that user performs queries on Descriptors and PAR/InternalPar for obtaining a list of results that satisfy the criteria. The DRM part is managed in a second step of the query, where from the licence DB, all the user that have a licence to sell a licence to the user according to the PAR limitation imposed, are automatically extracted, so that the user the performed the query can select the most appropriate to its needs. For these reasons the limitation on DRM that were present in the previous query model have been removed, moving this functionality to the automatic generation of a list of possible licensors.



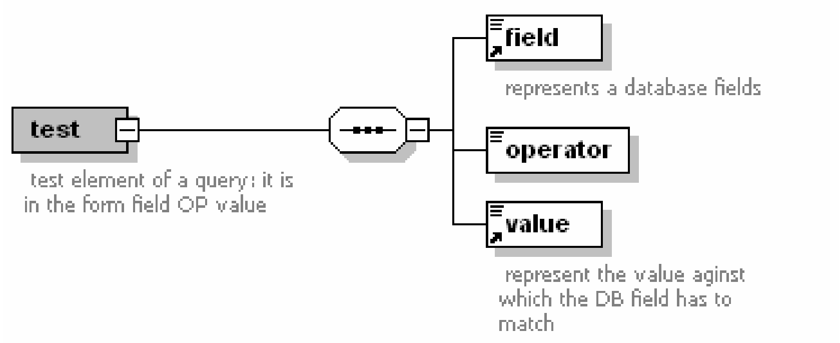
In this first diagram we suppose that a query can be performed on different sources, returning different AXINFO fields and that the query conditions can be expressed on three different information sources:

- PARquery, that are the info related to Possible Available Rights
- DRMMquery, that are the information about licensing
- AXinfoQuery, that are the info related to the medatada of the AXMEDIS object.

For each one of these groups a query condition applies and the groups are merged on an AND basis. Querycondition is reported in the following schema:



Querycondition is then a set of nesting elements that represents basically the parentheses levels.
A nesting can be nested in another nesting element of it can be a part of the real query, in the case it is part of a real query it is a combination of test items that are reported in the following:


test
test element of a query: it is in the form field OP value
field
represents a database fields
operator
value
represent the value aginst which the DB field has to match

Test is the basic element where a fields can be compared to a value by the means of an operator that will be discussed in more details in the following.
Test element has also an attribute that allows to specify if it must be negated or not.

It is very important to note how the field has to be specified in order to obtain information from the query. Each field is defined in a mapping file described in section 20. The value that must be put in the text part of field is the concatenation between the descriptor and the DescriptorXPath.
The current defined mapping are evidenced in the following table, while for the optional fields the namespace will be used as Descriptor and the Xpath from the starting of the descriptor as Descriptor XPath.

field value	Description
AXINFO: accessModeId	Id of the access Mode of the object
AXINFO:axcid	Id of the creator
AXINFO:axdid	Id of the distributor
AXINFO:creationDate	date of creation of the object
AXINFO:governedObject	flag to assert if the object is governed

AXINFO:lastModificationDate	last modification date of the object
AXINFO:objectStatus	status of the object
AXINFO:ownerId	id of the owner of the object
AXINFO:protectedObject	flag to assert if the object is protected
AXINFO:version	version of the object
AXINFO:workItemId	id of the work item
AXINFO:uri	uri of the object to be downloaded
AXINFO:axoid	axoid of the object
DCMI: accrualMethod	accrualMethod of the DCMI
DCMI: accrualPeriodicity	accrualPeriodicity of the DCMI
DCMI: accrualPolicy	accrualPolicy of the DCMI
DCMI: audience	audience of the DCMI (mathes also all the refinements)
DCMI: audience_educationLevel	educationLevel refinement of the audience DCMI
DCMI: audience_mediator	mediator refinement of the audience DCMI
DCMI: contributor	contributor of the DCMI
DCMI: coverage	coverage of the DCMI (mathes also all the refinements)
DCMI: coverage_spatial	spatial refinement of the coverage DCMI
DCMI: coverage_temporal	temporal refinement of the coverage DCMI
DCMI: creator	creator of the DCMI
DCMI: date	date of the DCMI (mathes also all the refinements)
DCMI: date_available	available refinement of the date DCMI
DCMI: date_created	created refinement of the date DCMI
DCMI: date_dateAccepted	dateAccepted refinement of the date DCMI
DCMI: date_dateCopyrighted	dateCopyrighted refinement of the date DCMI
DCMI: date_dateSubmitted	dateSubmitted refinement of the date DCMI
DCMI: date_issued	issued refinement of the date DCMI
DCMI: date_modified	modified refinement of the date DCMI
DCMI: date_valid	valid refinement of the date DCMI
DCMI: description	description of the date DCMI (mathes also all the refinements)
DCMI: description_abstract	abstract refinement of the description DCMI
DCMI: description_tableOfContents	tableOfContents refinement of the description DCMI
DCMI: format	format of te DCMI (mathes also all the refinements)
DCMI: format_extent	extent refinement of the format DCMI
DCMI: format_medium	medium refinement of the format DCMI
DCMI: identifier	identifier of the DCMI (mathes also all the refinements)
DCMI: identifier_bibliographicCitation	bibliographicCitation refinement of the identifier DCMI
DCMI: instructionalMethod	instructionalMethod of the DCMI
DCMI: language	language of the DCMI
DCMI: provenance	provenance of the DCMI
DCMI: publisher	publisher of the DCMI
DCMI: relation	relation of the DCMI (mathes also all the refinements)
DCMI: relation_conformsTo	conformsTo refinement of the relation DCMI
DCMI: relation_hasFormat	hasFormat refinement of the relation DCMI
DCMI: relation_hasPart	hasPart refinement of the relation DCMI
DCMI: relation_hasVersion	hasVersion refinement of the relation DCMI
DCMI: relation_isFormatOf	isFormatOf refinement of the relation DCMI
DCMI: relation_isPartOf	isPartOf refinement of the relation DCMI
DCMI: relation_isReferencedBy	isReferencedBy refinement of the relation DCMI
DCMI: relation_isReplacedBy	isReplacedBy refinement of the relation DCMI
DCMI: relation_isRequiredBy	isRequiredBy refinement of the relation DCMI
DCMI: relation_isVersionOf	isVersionOf refinement of the relation DCMI
DCMI: relation_references	references refinement of the relation DCMI
DCMI: relation_replaces	replaces refinement of the relation DCMI

DCMI: relation_requires	requires refinement of the relation DCMI
DCMI: rights	rights of the DCMI (mathes also all the refinements)
DCMI: rights_accessRights	accessRights refinement of the rights DCMI
DCMI: rights_license	license refinement of the rights DCMI
DCMI: rightsHolder	rightsHolder of the DCMI
DCMI: source	source of the DCMI
DCMI: subject	subject of the DCMI
DCMI: title	title of the DCMI (mathes also all the refinements)
DCMI: title_alternative	alternative refinement of the title DCMI
DCMI: type	type of the DCMI

The mapping used at the moment is reported in the following file for having a complete view:

```
<?xml version="1.0" encoding="UTF-8"?>
<AxdbMapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="axdb-mapping-v-1-2.xsd">
        <Map>
                <Descriptor>AXINFO</Descriptor>
                <DescriptorXPath>axoid</DescriptorXPath>
                <Table>did</Table>
                <FieldName>axoid</FieldName>
                <Description>AXOID of the object</Description>
        </Map>
        <Map>
                <Descriptor>AXINFO</Descriptor>
                <DescriptorXPath>uri</DescriptorXPath>
                <Table>did</Table>
                <FieldName>xmlObjectUri</FieldName>
                <Description>URI of the object</Description>
        </Map>
        <Map>
                <Descriptor>AXINFO</Descriptor>
                <DescriptorXPath>version</DescriptorXPath>
                <Table>did</Table>
                <FieldName>version</FieldName>
        </Map>
        <Map>
                <Descriptor>AXINFO</Descriptor>
                <DescriptorXPath>accessModeId</DescriptorXPath>
                <Table>axinfo</Table>
                <FieldName>accessModeId</FieldName>
        </Map>
        <Map>
                <Descriptor>AXINFO</Descriptor>
                <DescriptorXPath>axdid</DescriptorXPath>
                <Table>axinfo</Table>
                <FieldName>axdid</FieldName>
        </Map>
        <Map>
                <Descriptor>AXINFO</Descriptor>
```

```
            <DescriptorXPath>axcid</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>axcid</FieldName>
        </Map>
        <Map>
            <Descriptor>AXINFO</Descriptor>
            <DescriptorXPath>objectStatus</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>objectStatusId</FieldName>
        </Map>
        <Map>
            <Descriptor>AXINFO</Descriptor>
            <DescriptorXPath>protectedObject</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>protectedObject</FieldName>
        </Map>
        <Map>
            <Descriptor>AXINFO</Descriptor>
            <DescriptorXPath>governedObject</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>governedObject</FieldName>
        </Map>
        <Map>
            <Descriptor>AXINFO</Descriptor>
            <DescriptorXPath>workItemId</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>workItemId</FieldName>
        </Map>
        <Map>
            <Descriptor>AXINFO</Descriptor>
            <DescriptorXPath>ownerId</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>ownerId</FieldName>
        </Map>
        <Map>
            <Descriptor>AXINFO</Descriptor>
            <DescriptorXPath>creationDate</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>creationDate</FieldName>
        </Map>
        <Map>
            <Descriptor>AXINFO</Descriptor>
            <DescriptorXPath>lastModificationDate</DescriptorXPath>
            <Table>axinfo</Table>
            <FieldName>lastModificationDate</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>contributor</DescriptorXPath>
            <Table>dcmi</Table>
```

```
                <FieldName>contributor</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>coverage</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>coverage</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>coverage_spatial</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>coverage_spatial</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>coverage_temporal</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>coverage_temporal</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>creator</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>creator</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_available</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_available</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_created</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_created</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_dateAccepted</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_dateAccepted</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_dateCopyrighted</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_dateCopyrighted</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_dateSubmitted</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_dateSubmitted</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_issued</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_issued</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_modified</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_modified</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>date_valid</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>date_valid</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>description</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>description</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>description_abstract</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>description_abstract</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>description_tableOfContents</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>description_tableOfContents</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
```

```
            <DescriptorXPath>format</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>format</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>format_extent</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>format_extent</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>format_medium</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>format_medium</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>identifier</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>identifier</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>identifier_bibliographicCitation</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>identifier_bibliographicCitation</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>language</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>language</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>publisher</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>publisher</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_conformsTo</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_conformsTo</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_hasFormat</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_hasFormat</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_hasPart</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_hasPart</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_hasVersion</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_hasVersion</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_isFormatOf</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_isFormatOf</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_isPartOf</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_isPartOf</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_isReferencedBy</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_isReferencedBy</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_isReplacedBy</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_isReplacedBy</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_isRequiredBy</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_isRequiredBy</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_isVersionOf</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_isVersionOf</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_references</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_references</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_replaces</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_replaces</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>relation_requires</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>relation_requires</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>rights</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>rights</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>rights_accessRights</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>rights_accessRights</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>rights_license</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>rights_license</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>source</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>source</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>subject</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>subject</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>title</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>title</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>title_alternative</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>title_alternative</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>type</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>type</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>accrualMethod</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>accrualMethod</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>accrualPeriodicity</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>accrualPeriodicity</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>accrualPolicy</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>accrualPolicy</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>audience</DescriptorXPath>
            <Table>dcmi</Table>
            <FieldName>audience</FieldName>
        </Map>
        <Map>
            <Descriptor>DCMI</Descriptor>
            <DescriptorXPath>audience_educationLevel</DescriptorXPath>
            <Table>dcmi</Table>
```

```
                <FieldName>audience_educationLevel</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>audience_mediator</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>audience_mediator</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>instructionalMethod</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>instructionalMethod</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>provenance</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>provenance</FieldName>
        </Map>
        <Map>
                <Descriptor>DCMI</Descriptor>
                <DescriptorXPath>rightsHolder</DescriptorXPath>
                <Table>dcmi</Table>
                <FieldName>rightsHolder</FieldName>
        </Map>
</AxdbMapping>
```

Query sources

The AXMEDIS query can be distributed on four different locations that are: CRAWLER, AXDB, AXEPTOOL, and other remote query supports. This is the way the source tag has between 1 and 4 locations that can have a value in the enumeration CRAWLER, AXEPTOOL, AXDB, QS and have also an optional URL for covering the need of a remote Query Support that is needed by the Kiosk, when the kiosk will look in the kiosk factory if an object is preseeent or not. This new addition allows also to create a network of query supports that are cooperatively in an hierarchy tree.

Query result

The second part of the schema is used to describe the fields that the query should return together with the AXOID (or the temporary AXOID in the case of CRAWLER) that is considered a mandatory field. If the result tag is not present, then only the list of AXOID will be returned; otherwise at least one field must be specified. After the field list an optional array of sorting parameter can be inserter; each sortby tag has an optional attribute (not shown in the graphic version) that by default assumes the asc=true value, that means ascending sorting. If imposed to false, it means that sorting is descending.
The field tag will be limited to the possible list of fields, once the database schema will be defined and a selecting among the possible fields to be queried will be defined.

Query conditions

The most relevant part of a query is the section related to query conditions that is of course optional, but if present must have the structure reported in the previous XML schema. All the field tag must contains the database field that must match, all value field must contain the value against which the field is checked and operator is one of the following:

- GT: greater than
- LT: less than
- GE: greater equal
- LE: less equal
- EQ: Equal
- NE: Not equal
- STARTWITH: field must start with the value
- ENDWITH: field must end with the value
- CONTAINS: field must contain in any position the value

The field tag will be limited to the possible list of fields, once the database schema will be defined and a selecting among the possible fields to be queried will be defined.

Similarities

Similarities are a particular way of making a specific similarity request on the set ob objects selected by the previous queryconditions. By now only one similarity field as been inserted in order to reduce the complexity of the query. Similarities can be addressed by regular expressions (i.e. for searching a partially known AXOID) or by more advanced algorithms to be defined on the basis of the needs of the project. For the moment no algorithm has been implemented. This part was suggested by UR but has never been supported or requested by any other partner. This section can be considered as a space for further development.

XML Query example file

In this example the query proposed as a sample by ILABS that is:

SELECT * **FROM** * **WHERE ((**AUTHOR **.EQ.** ”BOTTICELLI” **.AND. (** MEDIA **.EQ.** “VIDEO” **.OR.** MEDIA **.EQ.** “AUDIO” **.OR.** MEDIA **.EQ.** “TEXT”**)) .AND.** (IPR **.EQ.** “FREE” **.AND.** COST .LT. “10.00”**))**

Apart from the fields for which we suppose to have only the AXOID and

```
<?xml version="1.0" encoding="UTF-8"?>
<query xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="QUERY-v1-6.xsd">
        <source>
                <location>CRAWLER</location>
                <location>AXEPTOOL</location>
                <location>AXDB</location>
        </source>
        <AXinfoQuery>
                <querycondition>
                        <nesting>
                                <nesting>
                                        <nesting>
                                                        <test>
                                                                <field>AUTHOR</field>
                                                                <operator>EQ</operator>
                                                                <value>BOTTICELLI</value>
                                                        </test>
                                        </nesting>
                                        <and/>
                                        <nesting>
                                                        <test>
                                                                <field>MEDIA</field>
                                                                <operator>EQ</operator>
                                                                <value>VIDEO</value>
                                                        </test>
                                                        <or/>
                                                        <test>
                                                                <field>MEDIA</field>
                                                                <operator>EQ</operator>
                                                                <value>AUDIO</value>
                                                        </test>
                                                        <or/>
                                                        <test>
                                                                <field>MEDIA</field>
                                <operator>EQ</operator>
                                <value>TEXT</value>
                            </test>
                    </nesting>
                </nesting>
                <and/>
                <nesting>
                        <test>
                            <field>IPR</field>
                            <operator>EQ</operator>
                            <value>FREE</value>
                        </test>
                        <and/>
                        <test>
                            <field>COST</field>
                            <operator>LT</operator>
                            <value>10.00</value>
                        </test>
                </nesting>
            </nesting>
        </querycondition>
    </AXinfoQuery>
</query>
```

Examples can be as complex as we want since the possibility of combining with AND and OR the number of desired nesting levels.
Another sample of a simpler query is reported in the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<query xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation=" QUERY-v1-6.xsd">
    <source>
        <location>AXDB</location>
    </source>
    <AXinfoQuery>
        <querycondition>
            <nesting>
                <test>
                    <field>PIPPO</field>
                    <operator>EQ</operator>
                    <value>3</value>
                </test>
                <and/>
                <nesting>
                    <test>
                        <field>PAPERINO</field>
                        <operator>EQ</operator>
                        <value>1</value>
                    </test>
                    <or/>
                    <test>
                        <field>PLUTO</field>
                        <operator>GT</operator>
                        <value>1</value>
                    </test>
                </nesting>
            </nesting>
        </querycondition>
    </AXinfoQuery>
</query>
```

11.1.2.2 Schema for an AXMEDIS query result

In this section the schema for the results that are returned back after a query is reported and discussed.
It is important to track if the result is coming from a source (i.e. CRAWLER, that means that some automatic operation for the creation of an AXMEDIS object starting from the content in the CMS have to be performed

before using the object) or from a remote channel such as AXEPTOOL that means that the content is not immediately available, but has to be downloaded from a remote peer.
The information related to the source of information must came together with the fields requested by the user in the query.

The proposed schema is reported in section 22, in text format and in a more fashionable graphic format that can be useful to have a general overview.

The schema of the result is quite simple since it is an optional list of AXMEDIS objects (AXObject) and for each object the information source is reported (AXEPTOOL, CRAWLER, AXDB, QS) together with the information that are relevant for the source, the list of peers that have the object together with the AXIOD for AXEPTOOL, the AXOID for AXDB and the temporary AXOID if the object is recovered by the crawler.
For each AXObject a set of extrainfo can be optionally reported. Each extrainfo tag is comprised of a couple of field and value. In this case also, when the AXDB schema will be stable we will be able to limit the valid content for field tag.
Errofields has been added as a support for the user in order to notify which fieldname present in the query were not present in the target source for query results. The elimination of such fields can provoke a not empty result.
RootObject element contains the list of elements that contain the target AXOID and that are downloadable or accessible in some way.

XML results example file

In this section are collected sample XML file compliant to the proposed schema that can be useful to understand the practical structure of the results of a query.

```
<?xml version="1.0" encoding="UTF-8"?>
    <AXObject>
        <AXEPTOOL>
            <peers>
                <peerID>12234</peerID>
                <peerID>56454</peerID>
                <peerID>12784</peerID>
            </peers>
            <AXOID>12248766-gftr</AXOID>
        </AXEPTOOL>
        <extrainfo>
            <field>Author</field>
            <value>Ramazzotti</value>
        </extrainfo>
        <extrainfo>
            <field>YearOfPublication</field>
            <value>1996</value>
        </extrainfo>
        <extrainfo>
            <field>Duration</field>
            <value>315</value>
        </extrainfo>
    </AXObject>
    <AXObject>
        <CRAWLER>
            <TAXOID>5383999-temp</TAXOID>
        </CRAWLER>
        <extrainfo>
            <field>Author</field>
            <value>Ramazzotti</value>
        </extrainfo>
        <extrainfo>
            <field>YearOfPublication</field>
            <value>1997</value>
        </extrainfo>
        <extrainfo>
```

```
                        <field>Duration</field>
                        <value>417</value>
                </extrainfo>
        </AXObject>
        <AXObject>
                <AXDB>
                        <AXOID>35423912843.fkhwgywe</AXOID>
                </AXDB>
                <extrainfo>
                        <field>Author</field>
                        <value>Ramazzotti</value>
                </extrainfo>
                <extrainfo>
                        <field>YearOfPublication</field>
                        <value>1995</value>
                </extrainfo>
                <extrainfo>
                        <field>Duration</field>
                        <value>234</value>
                </extrainfo>
        </AXObject>
</queryresults>
```

11.1.3 Selection Format and storage (EXITECH)

Selections are collections of symbolic queries and AXMEDIS Objects IDs. Selections are then a set comprised of symbolic queries not expanded or actualized such as S1= {Q1, Q2} or a selection of AXMEDIS Object IDs with some symbolic queries such as S2={Q3, AXOB1, AXOB444, AXOB3412}. Selection can be expanded or actualized so that S1 at time t1 can be S1@t1= {AXO1-1, AXO1-2, AXO1-3, AXO2-1, AXO2-2, AXO2-3, AXO2-4, AXO2-5, AXO2-6, AXO2-7, AXO2-8, AXO2-9}, while the same query at time t2 can be S1@t2= { AXO1-3, AXO2-1, AXO2-2, AXO2-3, AXO2-4, AXO2-5, AXO2-6, AXO2-8, AXO2-9, AXO3-1, AXO3-2}

A selection can be formalized in XML starting from what has been stated for query considering that in a selection only AXMEDIS objects and symbolic queries can be found. Considering the selection in a deeper detail, it is evident that only objects that are in AXDB can be part of the selection together with symbolic queries. This means that object extracted from the crawling system need to be transformed in AXMEDIS object to be inserted in a selection with their AXOID, and objects got from AXEPTOOL need to be imported in the AXDB before being used put in a selection.

The formal model for a selection can be summarized as a collection of queries and of AXMEDIS objects that can be recovered in the AXDB, and therefore a collection of AXOIDs.

11.1.3.1 XML Selection Schema

A selection is a collection of zero or more query with zero or more AXOID. It is necessary that at least an AXOID or a query is inside the selection.

11.1.3.2 XML Selection Samples

```
<?xml version="1.0" encoding="UTF-8"?>
<selection                              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Documents\Docs\Exitech\AXMEDIS\Deliverables\Specific
ation\AXMEDIS-DE3-1-2-AXFW\Current-PART-E\Selection-v1-1.xsd"                name="TEST"
timestamp="2005-01-20T18:20:46.275+01:00">
        <AXOID>3y7932469236</AXOID>
```

```
    <AXOID>824375832741723</AXOID>
    <query>
        <source>
            <location>CRAWLER</location>
        </source>
        <AXinfoQuery>
            <querycondition>
                <nesting>
                    <test>
                        <field>AUTHOR</field>
                        <operator>STARTWITH</operator>
                        <value>MOZ</value>
                    </test>
                </nesting>
            </querycondition>
        </AXinfoQuery>
    </query>
</selection>
```

11.1.3.3 XML Selection Storage

Selection are created by the user of the system or of the tool and therefore the logical place in which they can be stored is in the user profile. For the specification see section **Errore. L'origine riferimento non è stata trovata.**

11.1.4 Query Support WEB Service Interface (EXITECH)

The Web service technology is a widely adopted methods for exchanging information among servers distributed in a LAN/WAN environment. According to www.w3.org the definition of web service is:

"A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."

The Query Support web service interface is the module that offers an external interface for querying the AXMEDIX database (as well as AXEPTool and Crawling system) and its interface must be suitably defined by a WSDL that have to take care of the general structure of *AXMEDIS data structure schema*, in terms of query fields that can be adopted.

This module will have also the need for a database for supporting the operation of query distribution and Integration.

The ER diagram for the database section related to query support web interface is reported in section 18.5.3, while in the following the explanation of the tables that have been projected is commented.

Once the Query Support Web Service Interface receives a query, it passes the query to the Query Distribution module that stores the query in the Received Table with the reception timestamp and generating an unique ID for the query.

Once the query has been processed and optimized for the channels, a set of different queries are issued on the different channels. For each query a record in the Distributed table is created. The record comprises the queryID, the timestamp and the channel over which the query has been issued.

Each time a channel provide an answer asynchronously

With respect to the query, the Query Result Integration module write a record in the Integrated Table, setting the QueryID, the Channel from which the result has been received, the timestamp and the result.

Results will be back to the query issuer in two manners:

- Asynchronous: each time a result is obtained from one of the channel it is passed back to the requester
- Synchronous: Integrator waits to have all results before sending back the results set to the requester.

The database has to support both modes.
Since we have a webservice, we have two choices for sending and receiving a query, the first is to map inside a soap message and therefore also in the programming language the complex type that has been modelled for the query. This can be an hard task to be done since the query structure is very complex, but allows a formal verification on the query.
The second approach consists in putting the XML query in a string (a simple type for a SOAP message) that in easily mapped both in C++ and Java. This string can then be parsed by an XML parser, but it is impossible to check the correctness of the soap message with a schema before it is processed.
From a preliminary check with the identified library for webservices in C++, it has been evident that the mapping in C++ is very hard to be obtained and requires several manipulation to bring to an acceptable code that can be processed by gsoap parser for client and server side and for , and therefore, at least in this phase, where the query structure will be probably revised it is preferred to adopt a string approach. This approach will be revised in the seconfd phase of the specification when the model for the query and for the query results will be more stable.
The same approach will be used for results also both in synchronous and asynchronous mode.
The overview of the web service is detailed below, while the specification is reported in section 33.

Query_Support	
Method	***Description***
Make_Query_Sync	Method that allows to synchronously issue a query and get the corresponding result.
Make_Query_ASync	Method that allows to asynchronously issue a query and get the corresponding result on a listener specified by the user.

Since a listener is defined in the asynchronous mode, it is necessary to draw the specification also of this web service that will be offered and implemented by a third party that uses the asynchronous query result mechanism. Its specification is reported in section 34.

11.1.5 Query Distribution (EXITECH)

Query distribution is a module that is capable of distributing a query to different subsystems that are able to accept queries from the query support in order to collect result in a different phase by the means of the Query Result Integration module.
This module collects queries from the Query Support WebService Interface, assign a unique ID to the query in order to have the capability of collecting in the future the results, and store in temporary DB the queries and the associated IDs with the channel over the query has been issued.
This module in mainly a web service client of the tool it interfaces:

- Collector Engine Query Support interface
- AXMEDIS database manager
- AXEPTOOL Query support Interface

All these tools have to export a WebService interface that accept a query and must be capable of issuing notifications to the Query Result Integrator, by the means of a web service call, that will be discussed in the paragraph related to the Integrator.
In order to simplify the work, it is suggested to adopt the same WSDL provided by the query support to exchange results in response to a query both for synchronous and asynchronous model.

In the case, the synchronous mode will be requested, Query Distribution module will save directly on the DB the results or will call directly the Query results Integrator for doing such operation.

11.1.6 Query Results Integration (EXITECH)

This module interact via SOAP messages with all the web services capable of giving information on multimedia contents in the AXMEDIS architecture. By now it is limited to the interaction with AXMEDIS,

Crawler component and AXEPTool but the fact the it interrogates web services allow to extend its field of action to different entities that can arise during or after the project.
This module collect the results of the query from all the web services capable of returning information and give back a cumulative response to the Query support web service interface that in turn communicates to the GUI that can visualize the results.
This is also a QueryResultListener, since its architecture is compliant with that defined for the asyncronous return of results from a result source.
This module exports a web service interface for receiving the notification of the tool that will provide query results to the Query Support.
Depending on the operation mode (synchronous or asynchronous) the Query Results Intergration module will notify results to the Query Support WebService Interface that in turn will reorganize results.

11.1.7 Interface with ParDB (EXITECH, UPC)

The AXDB Query Support web service must return back results that are logically the AND between the condition on metadata (directly extracted from AXDB) and the condition on licenses and PARs that have to be obtained from the web service provided by PUPF.
Since it is not reasonable to gather all the result from the AXDB and all the results from ParDb fro doing an AND among them, it is necessary that ParDb operates the PAR and license verification only on the AXOID that satisfy the metadata condition.
To this end the AXDB Query Support webservice once obtained from AXDB all the AXOID that satisfy the query generates a new query to be issued to ParDb by changing the <PARquery> tag.
For example if the following query is issued to the Query Support, where in boldface teh original section for PAR is reported:

```
<query xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://localhost/QUERY-v1-6.xsd">
        <source>
                <location>AXDB</location>
        </source>
        <PARquery>
                <querycondition>
                        <nesting>
                                <test>
                                        <field>distributable</field>
                                        <operator>EQ</operator>
                                        <value>italy</value>
                                </test>
                        </nesting>
                </querycondition>
        </PARquery>
        <AXinfoQuery>
                <querycondition>
                        <nesting>
                                <test>
                                        <field>DCMI:type</field>
                                        <operator>STARTWITH</operator>
                                        <value>commedia</value>
                                </test>
                                <or/>
                                <test>
                                        <field>DCMI:type</field>
                                        <operator>STARTWITH</operator>
                                        <value>commedia</value>
                </test>
            </nesting>
        </querycondition>
    </AXinfoQuery>
</query>
```

And supposing that the results on metadata are the following AXOIDs:

- 02fa59bd-d089-4640-9c31-b78563a2e2d7
- 08180e7b-5ee1-44ef-b943-cb0424208012
- 0944c363-1bbd-4c50-8cd8-52f852a94399

The AXDB Query Support will forward to the ParDb the follwoing query that express the concept that the old Par bounds have to be satisfied and that that check ha sto be extend to the defined set of AXOID:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<query xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Y:\AXMEDIS\Deliverables\Specification\AXMEDIS-DE3-1-2-2\AX-
database-and-query-support-PART9\QUERY-v1-6.xsd">
    <source>
        <location>AXDB</location>
    </source>
    <PARquery>
        <querycondition>
            <nesting>
                <nesting>
                    <test>
                        <field>distributable</field>
                        <operator>EQ</operator>
                        <value>italy</value>
                    </test>
                </nesting>
                <and/>
                <nesting>
                    <test>
                        <field>AXOID</field>
                        <operator>EQ</operator>
                        <value>02fa59bd-d089-4640-9c31-b78563a2e2d7</value>
                    </test>
                    <or/>
                    <test>
                        <field>AXOID</field>
                        <operator>EQ</operator>
                        <value>08180e7b-5ee1-44ef-b943-cb0424208012</value>
                    </test>
                    <or/>
                    <test>
                        <field>AXOID</field>
                        <operator>EQ</operator>
                        <value>0944c363-1bbd-4c50-8cd8-52f852a94399</value>
                    </test>

                </nesting>
            </nesting>
```

```
                    </querycondition>
          </PARquery>
</query>
```

The new PAR section is in boldface and the metadata section has disappeared.

11.2 Module Design in terms of Classes

This module is quite complex since it has a core part and a Webservice part, the core part is divided in 5 packages included inside the it.exitech.axmedis.querysupport package. Four on five of these packages use JAXB for XML parsing and validation and therefore include an xml packages; each xml package in turn contains a set of other packages that are automatically generated from JAXB and therefore it is not useful to give evidence of their structure since it can be recovered in any JAXB related document.

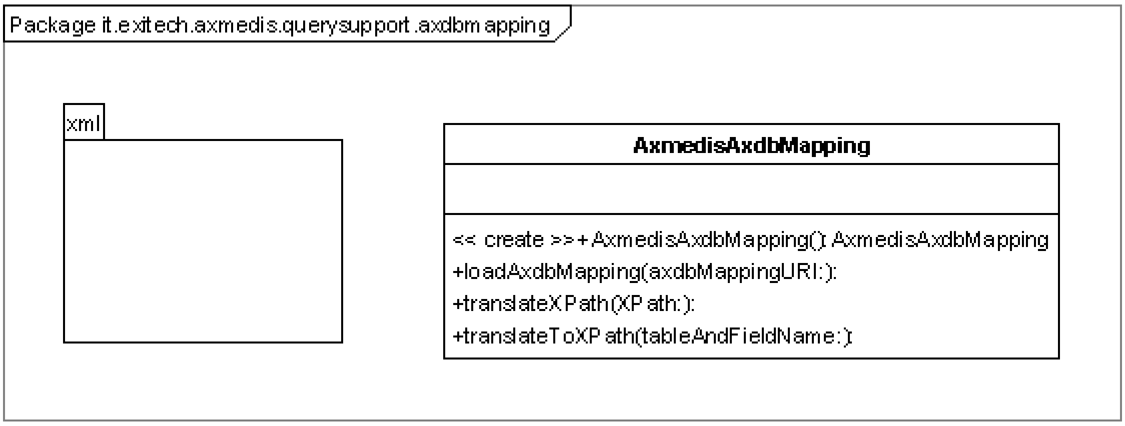
The webservice part is comprised of the general webservice and of the web service that is the adaptor toward the AXDB and therefore towards the LicenseDB.

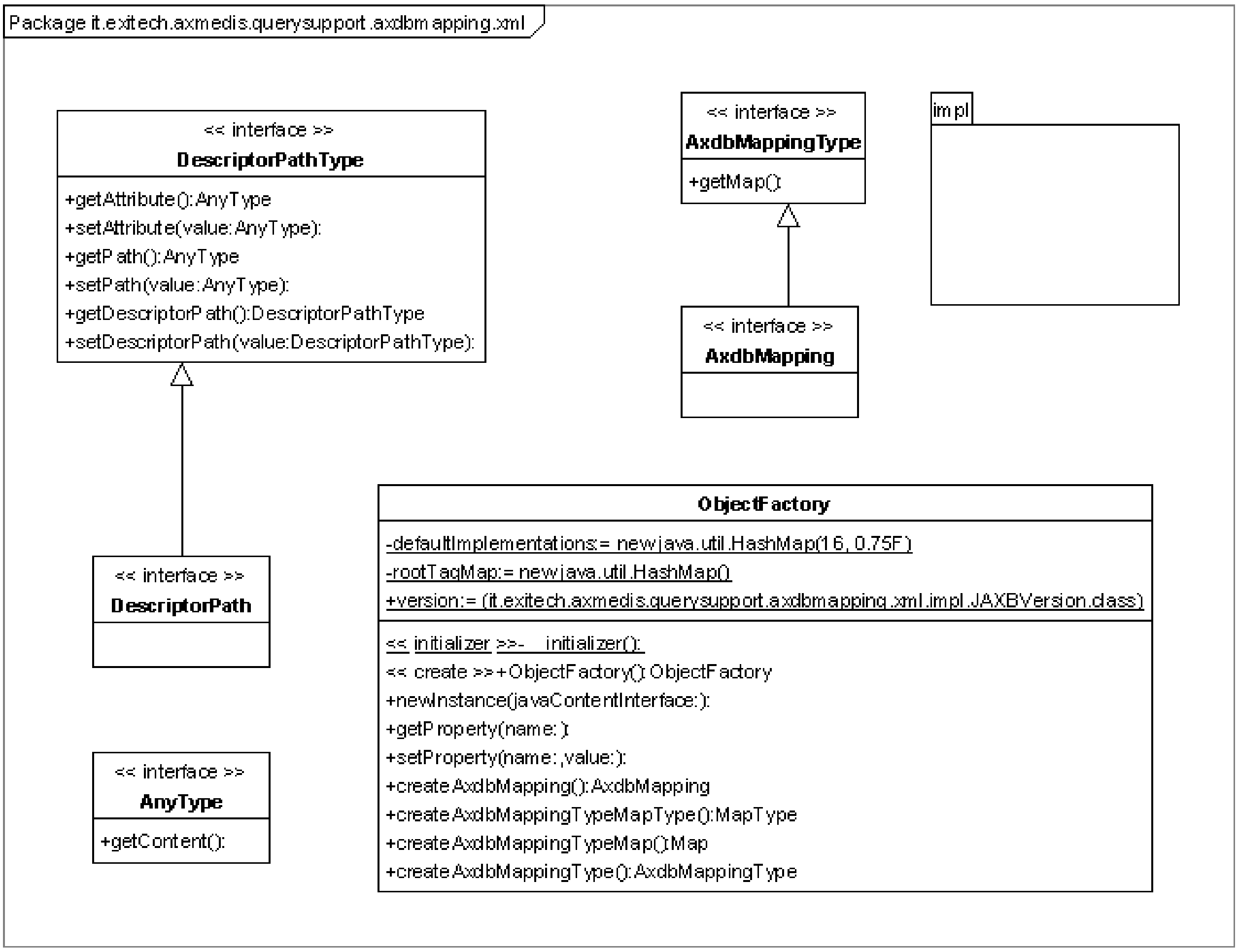
The AXDB adaptor is developed around two main packages that are the server package and the licence package, the first is the package that implement the server part of the web service, while the latter is related to the interfacing of the license database that implements that same WS interface.

The general webservice is developed around 3 sub-packages of the package it.exitech.axmedis.querysupport.mainquerysupport that are:

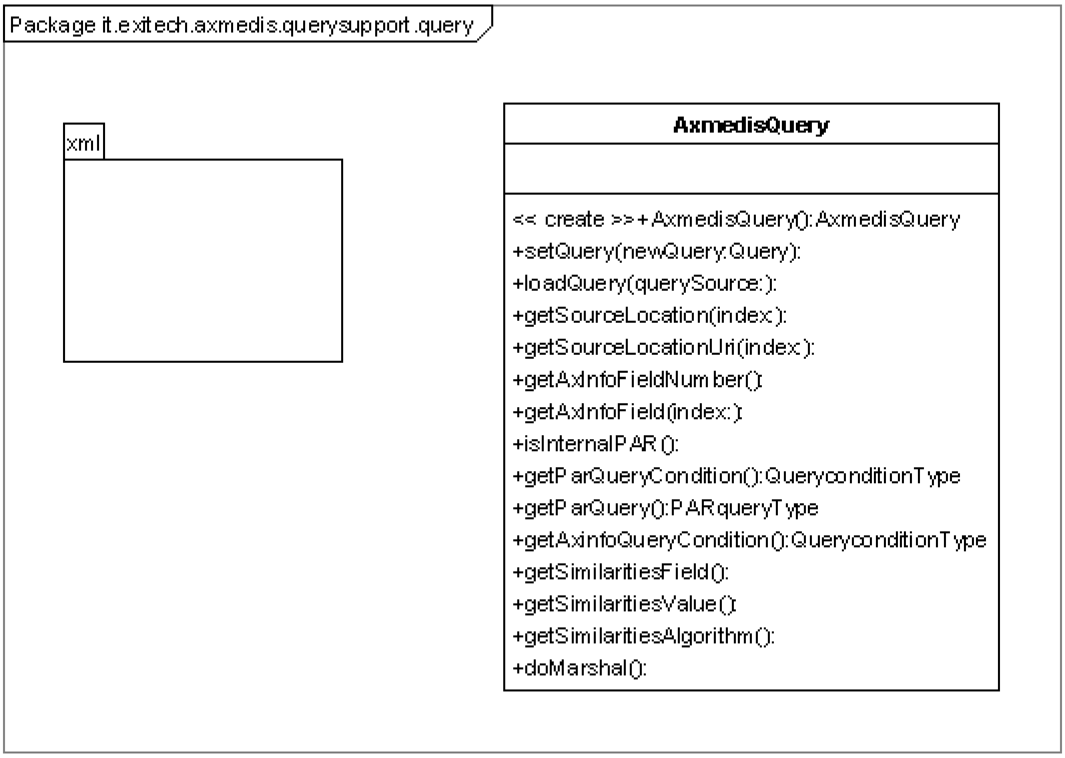
- the listenertable package, that has in charge the operation of collecting async responses and give them back to the client
- the listenerws package, that is a prototype of the listener ws, and
- the server package, that implements the server functionalities of the web service

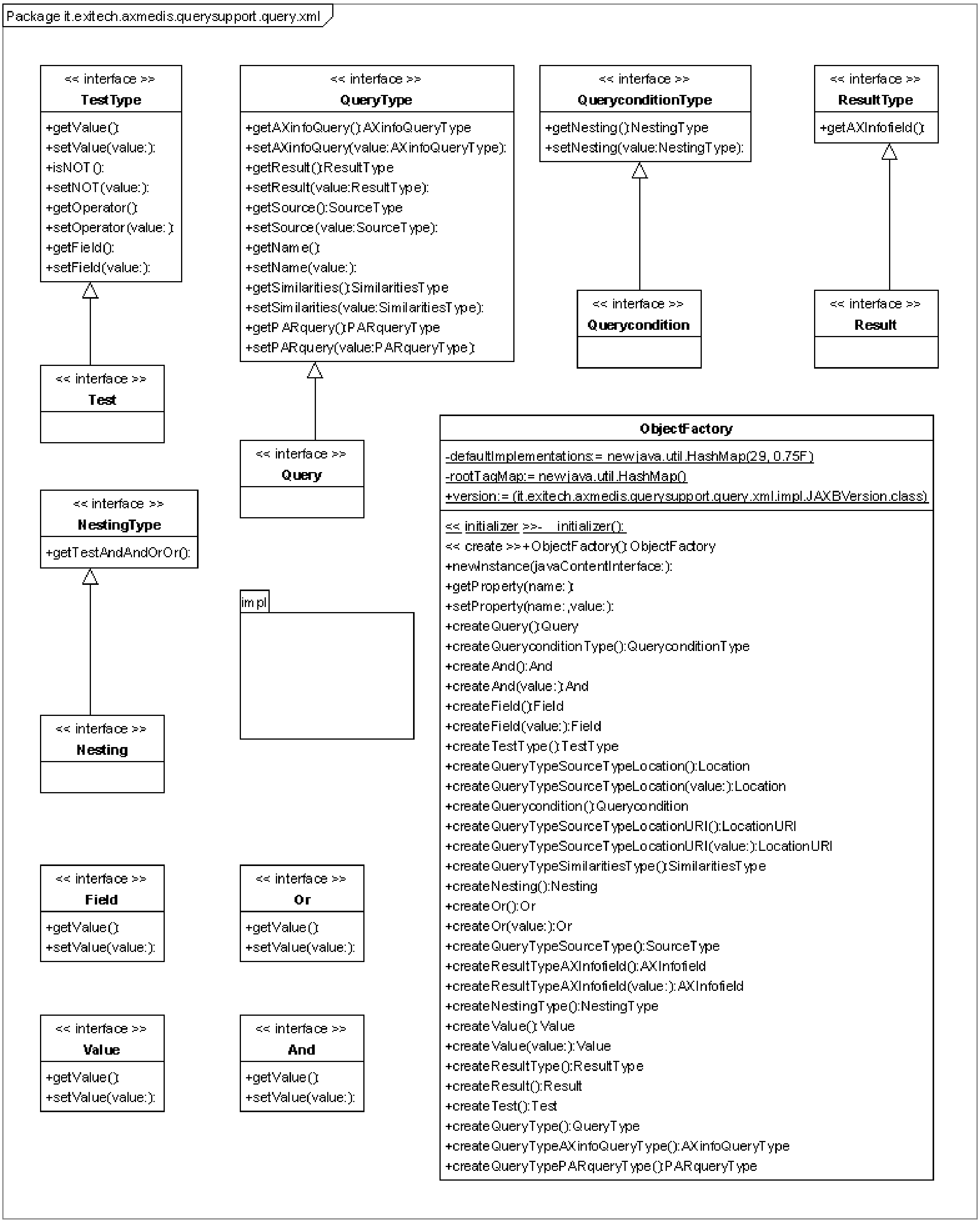
11.2.1 Package axdbmapping


Package it.exitech.axmedis.querysupport.axdbmapping
xml
AxmedisAxdbMapping
<< create >>+AxmedisAxdbMapping():AxmedisAxdbMapping
+loadAxdbMapping(axdbMappingURI:):
+translateXPath(XPath:):
+translateToXPath(tableAndFieldName:):

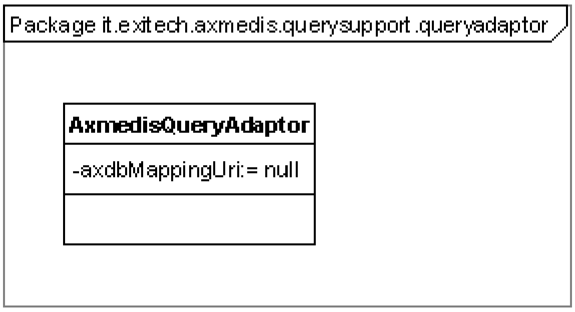

Package it.exitech.axmedis.querysupport.axdbmapping.xml
<< interface >>
DescriptorPathType
+getAttribute():AnyType
+setAttribute(value:AnyType):
+getPath():AnyType
+setPath(value:AnyType):
+getDescriptorPath():DescriptorPathType
+setDescriptorPath(value:DescriptorPathType):
<< interface >>
AxdbMappingType
+getMap():
impl
<< interface >>
AxdbMapping
<< interface >>
DescriptorPath
ObjectFactory
-defaultImplementations:= new java.util.HashMap(16, 0.75F)
-rootTagMap:= new java.util.HashMap()
+version:= (it.exitech.axmedis.querysupport.axdbmapping.xml.impl.JAXBVersion.class)
<< initializer >>- initializer():
<< create >>+ObjectFactory():ObjectFactory
+newInstance(javaContentInterface:):
+getProperty(name:):
+setProperty(name:,value:):
+createAxdbMapping():AxdbMapping
+createAxdbMappingTypeMapType():MapType
+createAxdbMappingTypeMap():Map
+createAxdbMappingType():AxdbMappingType
<< interface >>
AnyType
+getContent():

11.2.2 Package query

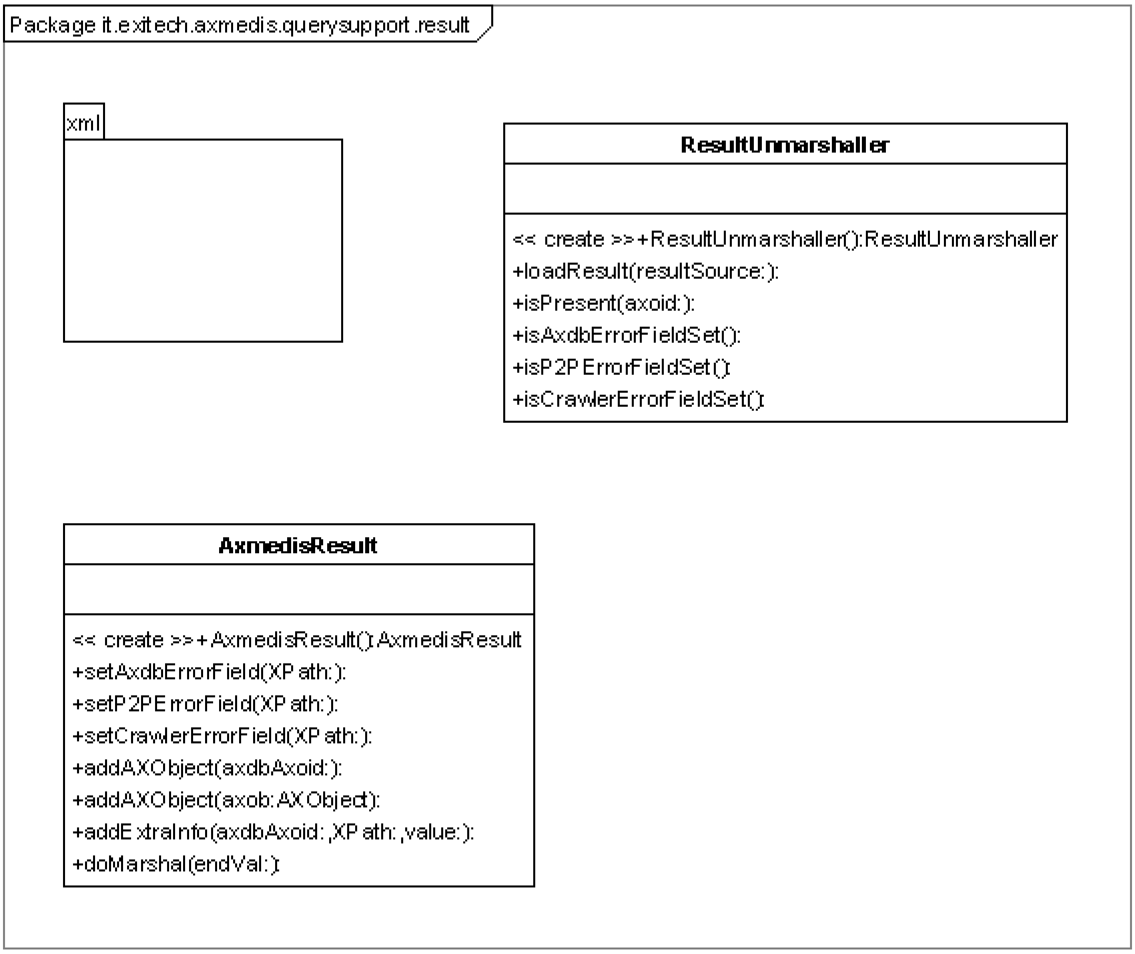

Package it.exitech.axmedis.querysupport.query
xml
AxmedisQuery
<< create >>+AxmedisQuery():AxmedisQuery
+setQuery(newQuery:Query):
+loadQuery(querySource:):
+getSourceLocation(index:):
+getSourceLocationUri(index:):
+getAxInfoFieldNumber():
+getAxInfoField(index:):
+isInternalPAR():
+getParQueryCondition():QueryconditionType
+getParQuery():PARqueryType
+getAxinfoQueryCondition():QueryconditionType
+getSimilaritiesField():
+getSimilaritiesValue():
+getSimilaritiesAlgorithm():
+doMarshal():

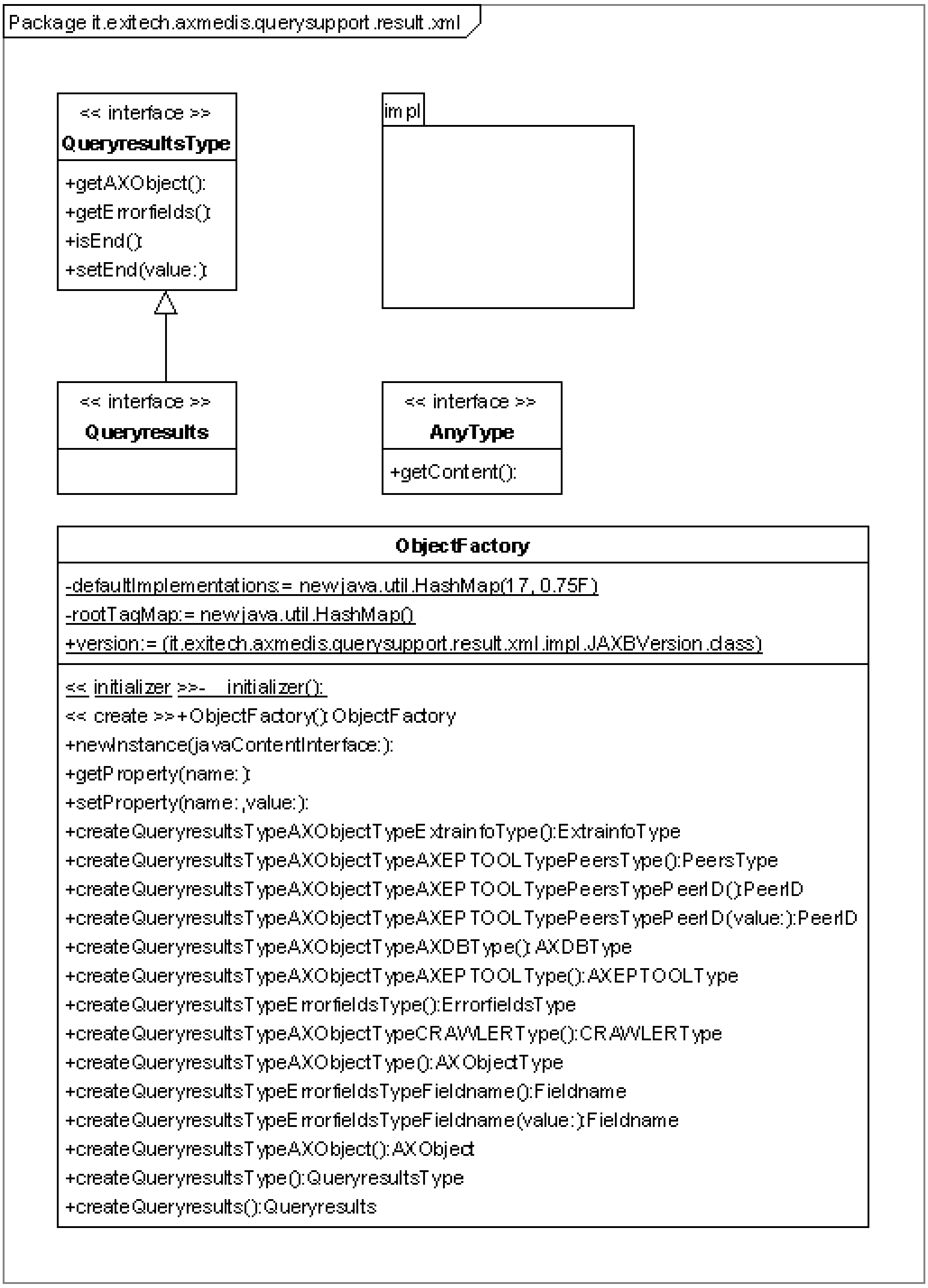

Package it.exitech.axmedis.querysupport.query.xml
<< interface >>
TestType
+getValue():
+setValue(value:):
+isNOT():
+setNOT(value:):
+getOperator():
+setOperator(value:):
+getField():
+setField(value:):
<< interface >>
QueryType
+getAXinfoQuery():AXinfoQueryType
+setAXinfoQuery(value:AXinfoQueryType):
+getResult():ResultType
+setResult(value:ResultType):
+getSource():SourceType
+setSource(value:SourceType):
+getName():
+setName(value:):
+getSimilarities():SimilaritiesType
+setSimilarities(value:SimilaritiesType):
+getPARquery():PARqueryType
+setPARquery(value:PARqueryType):
<< interface >>
QueryconditionType
+getNesting():NestingType
+setNesting(value:NestingType):
<< interface >>
ResultType
+getAXInfofield():
<< interface >>
Querycondition
<< interface >>
Result
<< interface >>
Test
<< interface >>
Query
ObjectFactory
-defaultImplementations:= new java.util.HashMap(29, 0.75F)
-rootTagMap:= new java.util.HashMap()
+version:= (it.exitech.axmedis.querysupport.query.xml.impl.JAXBVersion.class)
<< initializer >>- initializer():
<< create >>+ObjectFactory():ObjectFactory
+newInstance(javaContentInterface:):
+getProperty(name:):
+setProperty(name:,value:):
+createQuery():Query
+createQueryconditionType():QueryconditionType
+createAnd():And
+createAnd(value:):And
+createField():Field
+createField(value:):Field
+createTestType():TestType
+createQueryTypeSourceTypeLocation():Location
+createQueryTypeSourceTypeLocation(value:):Location
+createQuerycondition():Querycondition
+createQueryTypeSourceTypeLocationURI():LocationURI
+createQueryTypeSourceTypeLocationURI(value:):LocationURI
+createQueryTypeSimilaritiesType():SimilaritiesType
+createNesting():Nesting
+createOr():Or
+createOr(value:):Or
+createQueryTypeSourceType():SourceType
+createResultTypeAXInfofield():AXInfofield
+createResultTypeAXInfofield(value:):AXInfofield
+createNestingType():NestingType
+createValue():Value
+createValue(value:):Value
+createResultType():ResultType
+createResult():Result
+createTest():Test
+createQueryType():QueryType
+createQueryTypeAXinfoQueryType():AXinfoQueryType
+createQueryTypePARqueryType():PARqueryType
<< interface >>
NestingType
+getTestAndAndOrOr():
impl
<< interface >>
Nesting
<< interface >>
Field
+getValue():
+setValue(value:):
<< interface >>
Or
+getValue():
+setValue(value:):
<< interface >>
Value
+getValue():
+setValue(value:):
<< interface >>
And
+getValue():
+setValue(value:):

11.2.3 Package queryadaptor

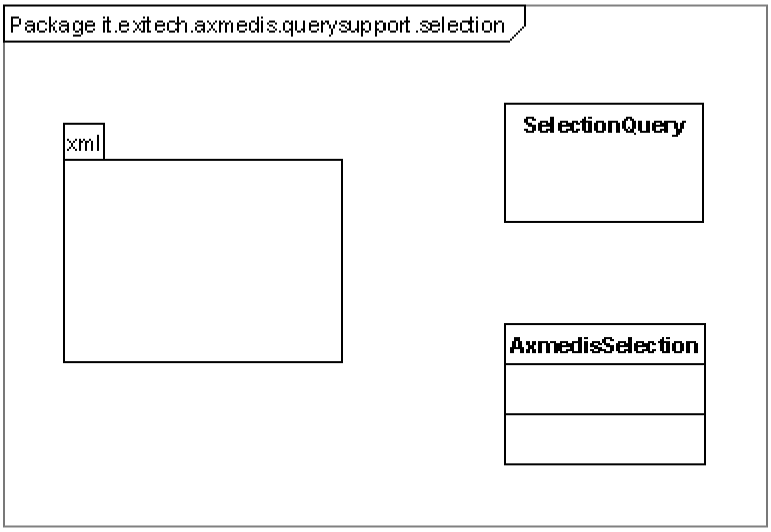


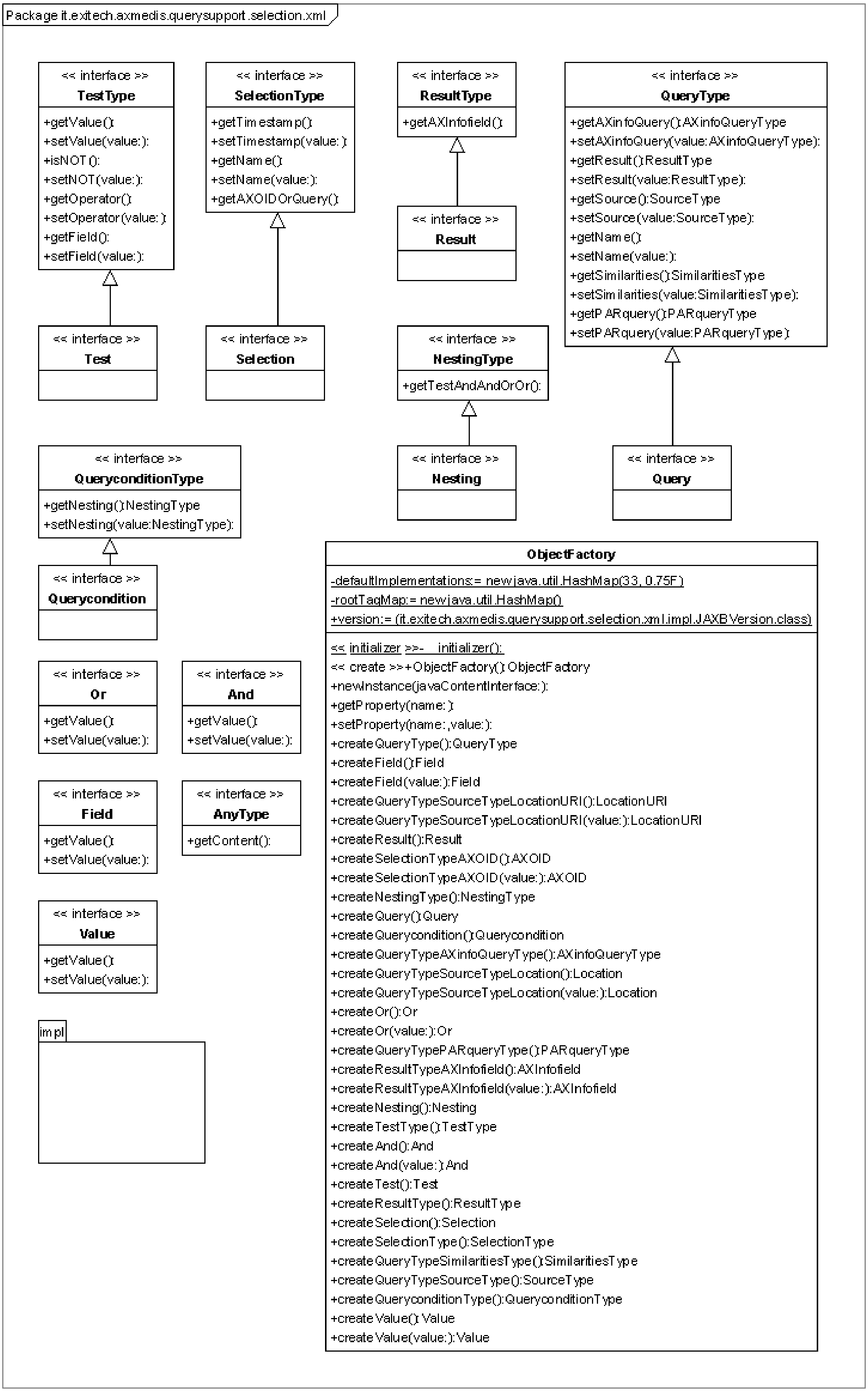
11.2.4 Package result


Package it.exitech.axmedis.querysupport.result
xml
ResultUnmarshaller
<< create >>+ResultUnmarshaller():ResultUnmarshaller
+loadResult(resultSource:):
+isPresent(axoid:):
+isAxdbErrorFieldSet():
+isP2PErrorFieldSet():
+isCrawlerErrorFieldSet():
AxmedisResult
<< create >>+AxmedisResult():AxmedisResult
+setAxdbErrorField(XPath:):
+setP2PErrorField(XPath:):
+setCrawlerErrorField(XPath:):
+addAXObject(axdbAxoid:):
+addAXObject(axob:AXObject):
+addExtraInfo(axdbAxoid:,XPath:,value:):
+doMarshal(endVal:):

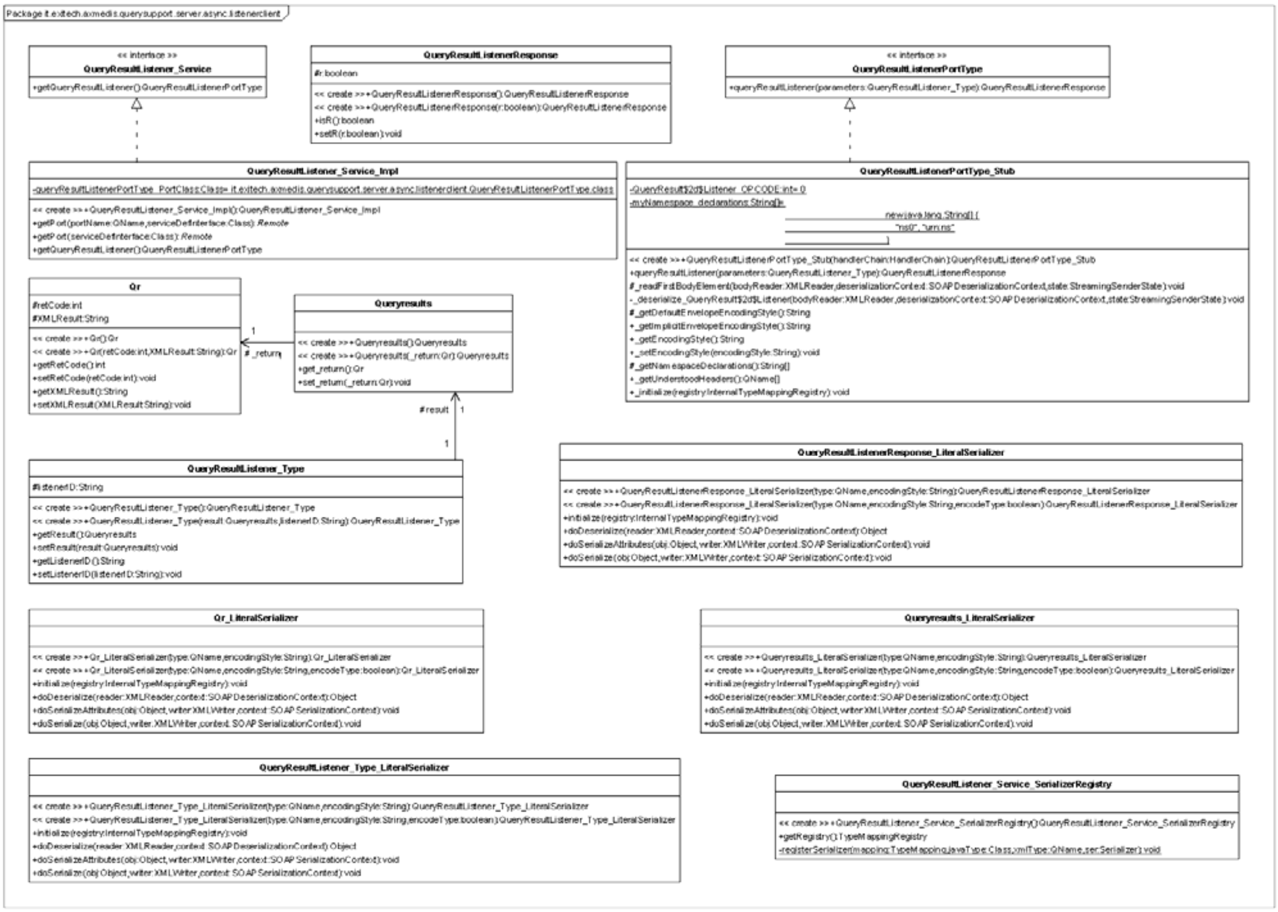

Package it.exitech.axmedis.querysupport.result.xml
<< interface >>
QueryresultsType
+getAXObject():
+getErrorfields():
+isEnd():
+setEnd(value:):
impl
<< interface >>
Queryresults
<< interface >>
AnyType
+getContent():
ObjectFactory
-defaultImplementations:= new java.util.HashMap(17, 0.75F)
-rootTagMap:= new java.util.HashMap()
+version:= (it.exitech.axmedis.querysupport.result.xml.impl.JAXBVersion.class)
<< initializer >>- initializer():
<< create >>+ObjectFactory(): ObjectFactory
+newInstance(javaContentInterface:):
+getProperty(name:):
+setProperty(name:,value:):
+createQueryresultsTypeAXObjectTypeExtrainfoType():ExtrainfoType
+createQueryresultsTypeAXObjectTypeAXEPTOOLTypePeersType():PeersType
+createQueryresultsTypeAXObjectTypeAXEPTOOLTypePeersTypePeerID():PeerID
+createQueryresultsTypeAXObjectTypeAXEPTOOLTypePeersTypePeerID(value:):PeerID
+createQueryresultsTypeAXObjectTypeAXDBType(): AXDBType
+createQueryresultsTypeAXObjectTypeAXEPTOOLType():AXEPTOOLType
+createQueryresultsTypeErrorfieldsType():ErrorfieldsType
+createQueryresultsTypeAXObjectTypeCRAWLERType():CRAWLERType
+createQueryresultsTypeAXObjectType():AXObjectType
+createQueryresultsTypeErrorfieldsTypeFieldname():Fieldname
+createQueryresultsTypeErrorfieldsTypeFieldname(value:):Fieldname
+createQueryresultsTypeAXObject():AXObject
+createQueryresultsType():QueryresultsType
+createQueryresults():Queryresults

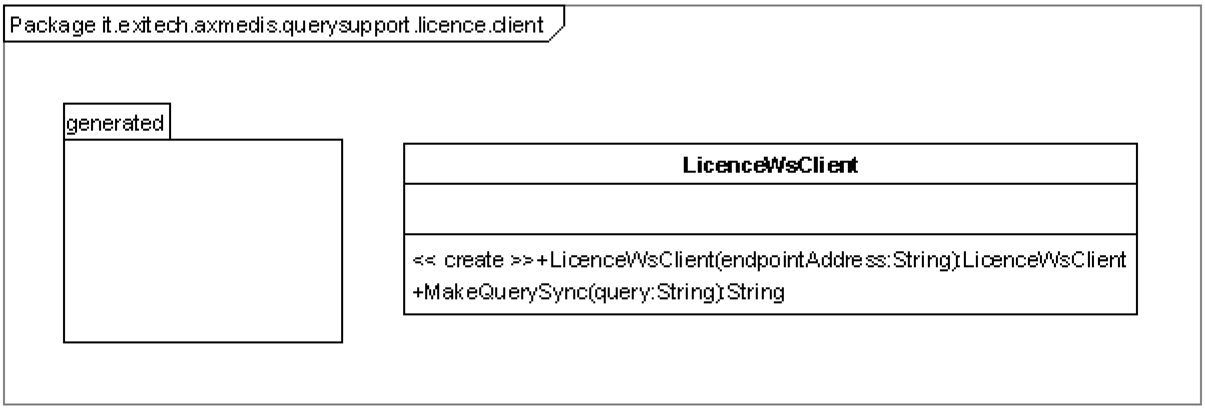
11.2.5 Package selection

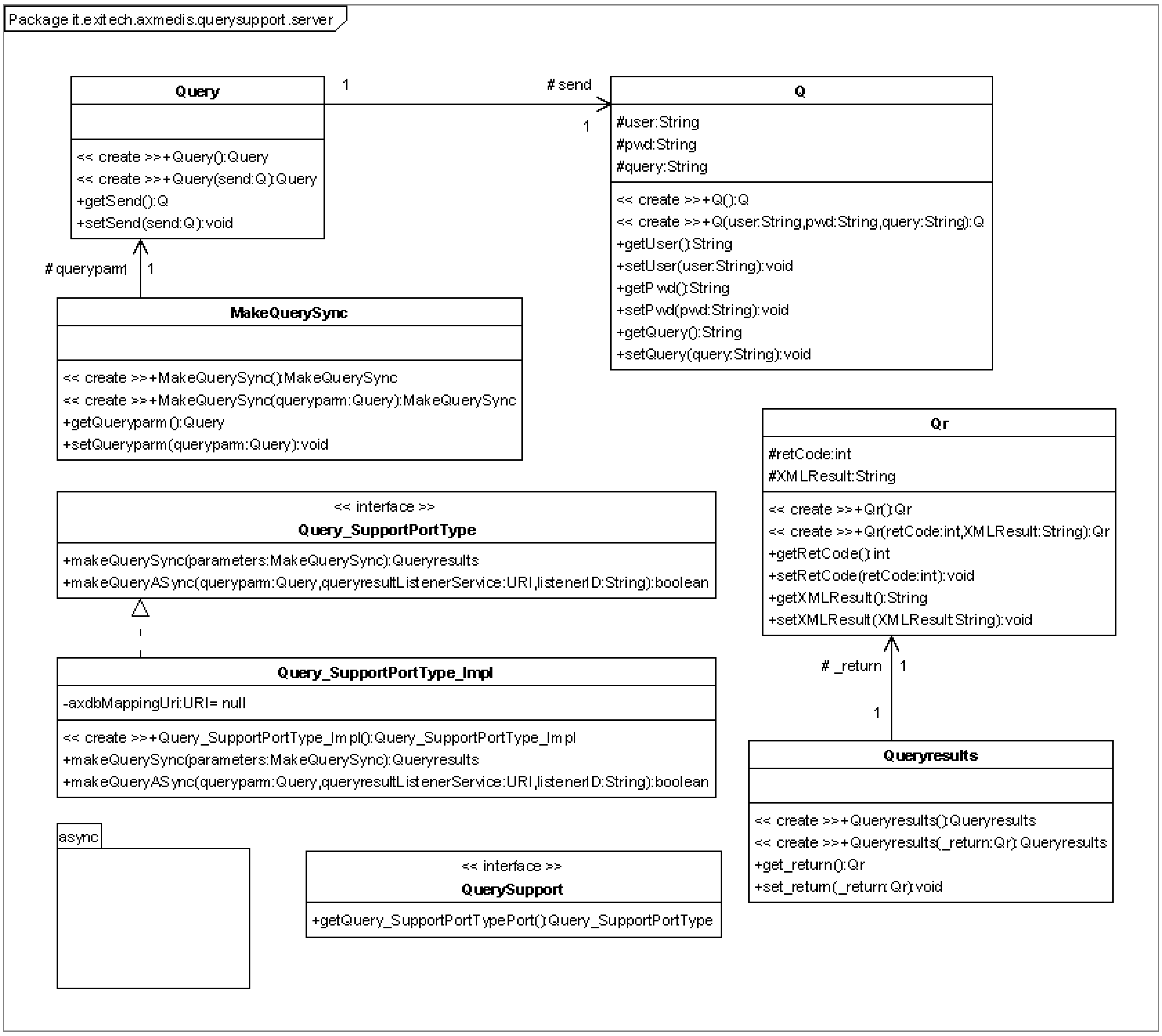


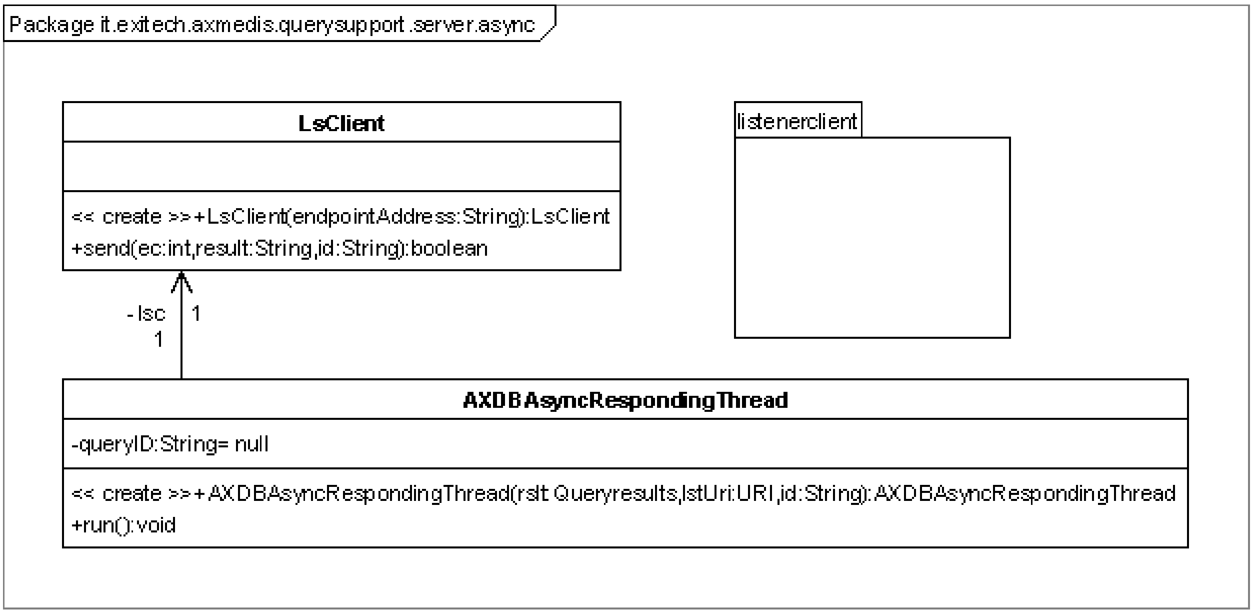

Package it.exitech.axmedis.querysupport.selection.xml
<< interface >>
TestType
+getValue():
+setValue(value:):
+isNOT():
+setNOT(value:):
+getOperator():
+setOperator(value:):
+getField():
+setField(value:):
<< interface >>
SelectionType
+getTimestamp():
+setTimestamp(value:):
+getName():
+setName(value:):
+getAXOIDOrQuery():
<< interface >>
ResultType
+getAXInfofield():
<< interface >>
QueryType
+getAXinfoQuery():AXinfoQueryType
+setAXinfoQuery(value:AXinfoQueryType):
+getResult():ResultType
+setResult(value:ResultType):
+getSource():SourceType
+setSource(value:SourceType):
+getName():
+setName(value:):
+getSimilarities():SimilaritiesType
+setSimilarities(value:SimilaritiesType):
+getPARquery():PARqueryType
+setPARquery(value:PARqueryType):
<< interface >>
Result
<< interface >>
Test
<< interface >>
Selection
<< interface >>
NestingType
+getTestAndAndOrOr():
<< interface >>
QueryconditionType
+getNesting():NestingType
+setNesting(value:NestingType):
<< interface >>
Nesting
<< interface >>
Query
ObjectFactory
-defaultImplementations:= new java.util.HashMap(33, 0.75F)
-rootTagMap:= new java.util.HashMap()
+version:= (it.exitech.axmedis.querysupport.selection.xml.impl.JAXBVersion.class)
<< initializer >>- initializer():
<< create >>+ObjectFactory():ObjectFactory
+newInstance(javaContentInterface:):
+getProperty(name:):
+setProperty(name:,value:):
+createQueryType():QueryType
+createField():Field
+createField(value:):Field
+createQueryTypeSourceTypeLocationURI():LocationURI
+createQueryTypeSourceTypeLocationURI(value:):LocationURI
+createResult():Result
+createSelectionTypeAXOID():AXOID
+createSelectionTypeAXOID(value:):AXOID
+createNestingType():NestingType
+createQuery():Query
+createQuerycondition():Querycondition
+createQueryTypeAXinfoQueryType():AXinfoQueryType
+createQueryTypeSourceTypeLocation():Location
+createQueryTypeSourceTypeLocation(value:):Location
+createOr():Or
+createOr(value:):Or
+createQueryTypePARqueryType():PARqueryType
+createResultTypeAXInfofield():AXInfofield
+createResultTypeAXInfofield(value:):AXInfofield
+createNesting():Nesting
+createTestType():TestType
+createAnd():And
+createAnd(value:):And
+createTest():Test
+createResultType():ResultType
+createSelection():Selection
+createSelectionType():SelectionType
+createQueryTypeSimilaritiesType():SimilaritiesType
+createQueryTypeSourceType():SourceType
+createQueryconditionType():QueryconditionType
+createValue():Value
+createValue(value:):Value
<< interface >>
Querycondition
<< interface >>
Or
+getValue():
+setValue(value:):
<< interface >>
And
+getValue():
+setValue(value:):
<< interface >>
Field
+getValue():
+setValue(value:):
<< interface >>
AnyType
+getContent():
<< interface >>
Value
+getValue():
+setValue(value:):
impl

11.2.6 Package server

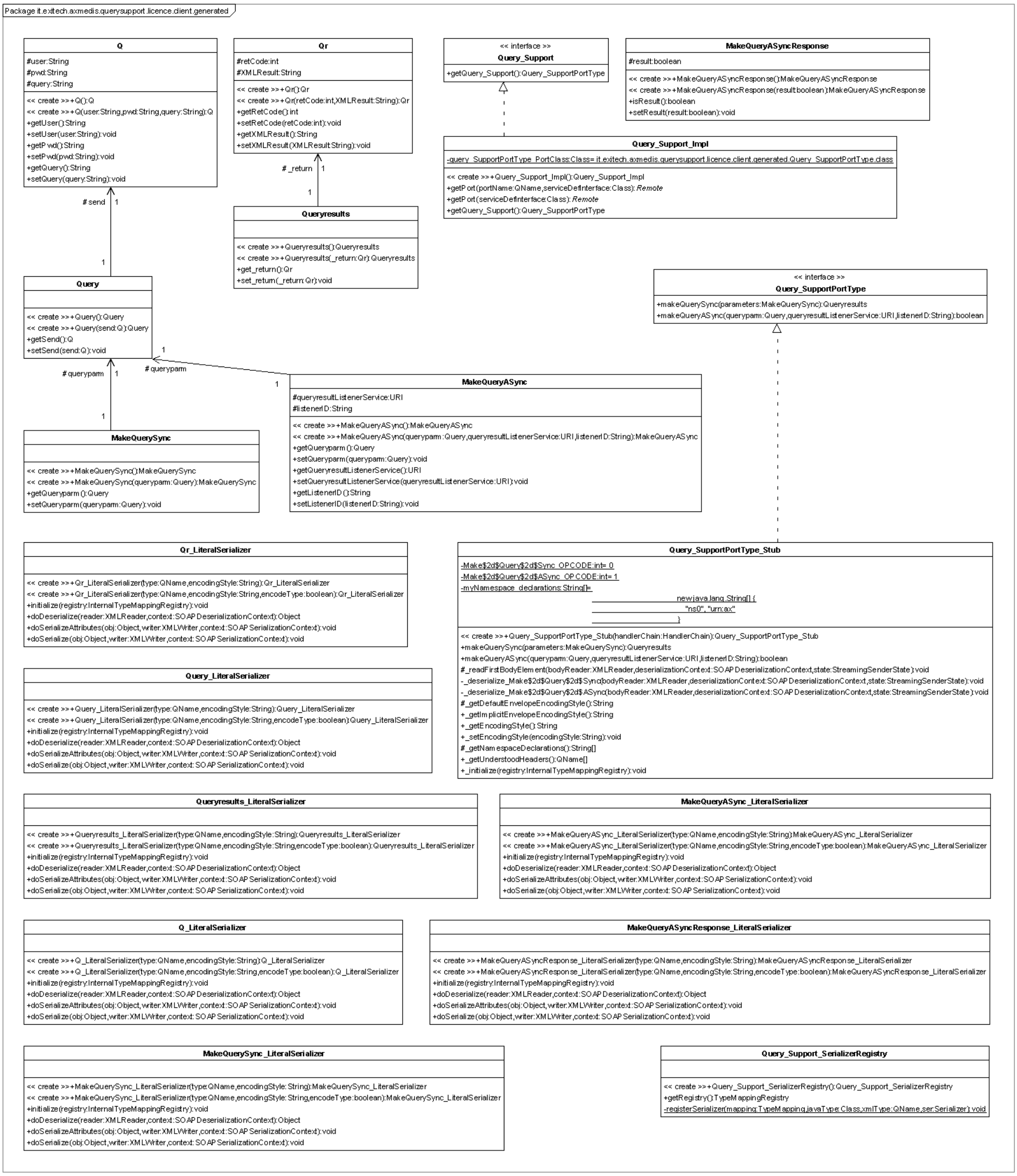

Package it.exitech.axmedis.querysupport.server.async.listenerclient
<< interface >>
QueryResultListener_Service
+getQueryResultListener():QueryResultListenerPortType
QueryResultListenerResponse
#r:boolean
<< create >>+QueryResultListenerResponse():QueryResultListenerResponse
<< create >>+QueryResultListenerResponse(r:boolean):QueryResultListenerResponse
+isR():boolean
+setR(r:boolean):void
<< interface >>
QueryResultListenerPortType
+queryResultListener(parameters:QueryResultListener_Type):QueryResultListenerResponse
QueryResultListener_Service_Impl
-queryResultListenerPortType_PortClass:Class= it.exitech.axmedis.querysupport.server.async.listenerclient.QueryResultListenerPortType.class
<< create >>+QueryResultListener_Service_Impl():QueryResultListener_Service_Impl
+getPort(portName:QName,serviceDefInterface:Class):Remote
+getPort(serviceDefInterface:Class):Remote
+getQueryResultListener():QueryResultListenerPortType
QueryResultListenerPortType_Stub
-QueryResult$2d$Listener_OPCODE:int= 0
-myNamespace_declarations:String[]=
new java.lang.String[] {
"ns0", "urn:ns"
}
<< create >>+QueryResultListenerPortType_Stub(handlerChain:HandlerChain):QueryResultListenerPortType_Stub
+queryResultListener(parameters:QueryResultListener_Type):QueryResultListenerResponse
#_readFirstBodyElement(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
-_deserialize_QueryResult$2d$Listener(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
#_getDefaultEnvelopeEncodingStyle():String
+_getImplicitEnvelopeEncodingStyle():String
+_getEncodingStyle():String
+_setEncodingStyle(encodingStyle:String):void
#_getNamespaceDeclarations():String[]
+_getUnderstoodHeaders():QName[]
+_initialize(registry:InternalTypeMappingRegistry):void
Qr
#retCode:int
#XMLResult:String
<< create >>+Qr():Qr
<< create >>+Qr(retCode:int,XMLResult:String):Qr
+getRetCode():int
+setRetCode(retCode:int):void
+getXMLResult():String
+setXMLResult(XMLResult:String):void
1
#_return
Queryresults
<< create >>+Queryresults():Queryresults
<< create >>+Queryresults(_return:Qr):Queryresults
+get_return():Qr
+set_return(_return:Qr):void
#result
1
1
QueryResultListener_Type
#listenerID:String
<< create >>+QueryResultListener_Type():QueryResultListener_Type
<< create >>+QueryResultListener_Type(result:Queryresults,listenerID:String):QueryResultListener_Type
+getResult():Queryresults
+setResult(result:Queryresults):void
+getListenerID():String
+setListenerID(listenerID:String):void
QueryResultListenerResponse_LiteralSerializer
<< create >>+QueryResultListenerResponse_LiteralSerializer(type:QName,encodingStyle:String):QueryResultListenerResponse_LiteralSerializer
<< create >>+QueryResultListenerResponse_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):QueryResultListenerResponse_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Qr_LiteralSerializer
<< create >>+Qr_LiteralSerializer(type:QName,encodingStyle:String):Qr_LiteralSerializer
<< create >>+Qr_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Qr_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Queryresults_LiteralSerializer
<< create >>+Queryresults_LiteralSerializer(type:QName,encodingStyle:String):Queryresults_LiteralSerializer
<< create >>+Queryresults_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Queryresults_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
QueryResultListener_Type_LiteralSerializer
<< create >>+QueryResultListener_Type_LiteralSerializer(type:QName,encodingStyle:String):QueryResultListener_Type_LiteralSerializer
<< create >>+QueryResultListener_Type_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):QueryResultListener_Type_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
QueryResultListener_Service_SerializerRegistry
<< create >>+QueryResultListener_Service_SerializerRegistry():QueryResultListener_Service_SerializerRegistry
+getRegistry():TypeMappingRegistry
-registerSerializer(mapping:TypeMapping,javaType:Class,xmlType:QName,ser:Serializer):void


Package it.exitech.axmedis.querysupport.licence.client
generated
LicenceWsClient
<< create >>+LicenceWsClient(endpointAddress:String):LicenceWsClient
+MakeQuerySync(query:String):String

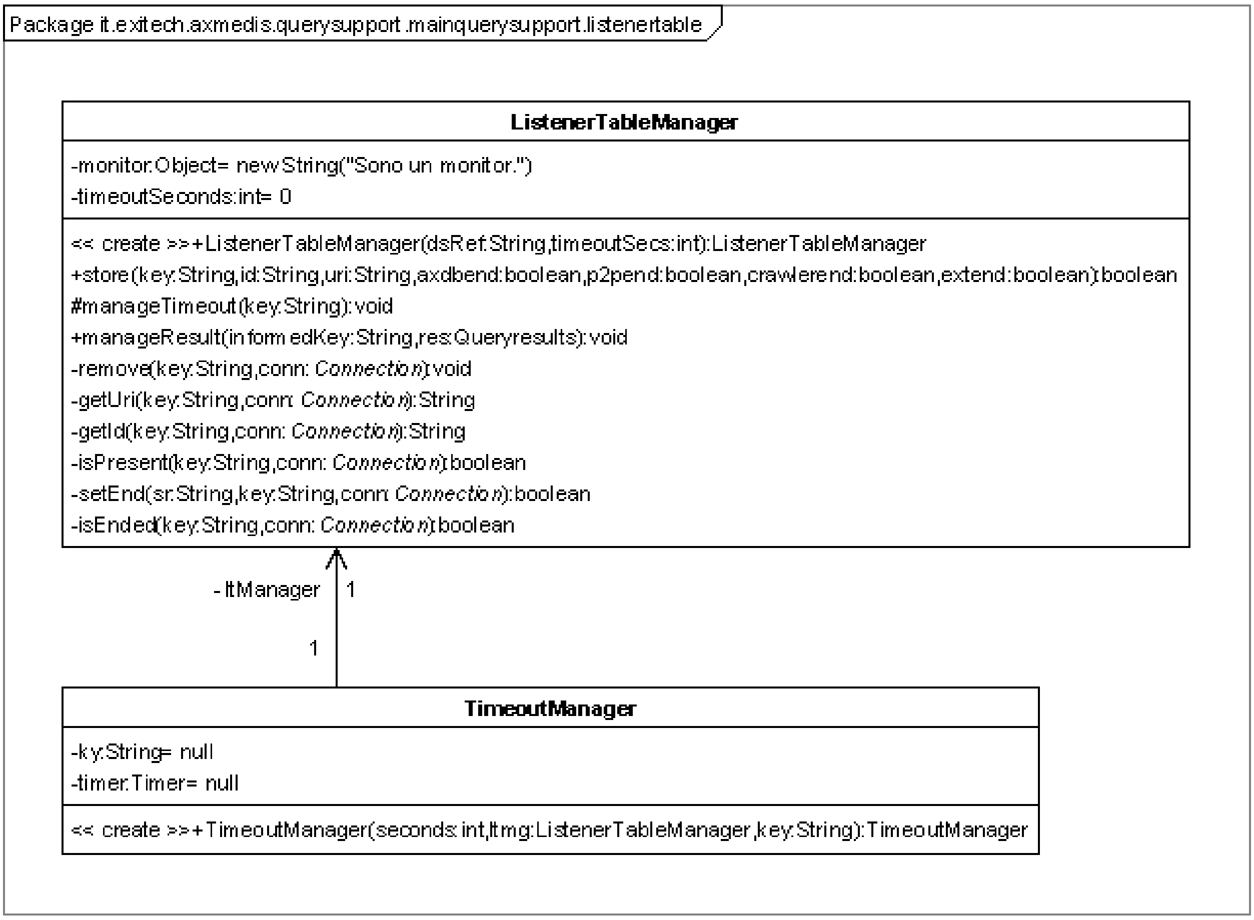

Package it.exitech.axmedis.querysupport.server
Query
<< create >>+Query():Query
<< create >>+Query(send:Q):Query
+getSend():Q
+setSend(send:Q):void
1
#send
1
Q
#user:String
#pwd:String
#query:String
<< create >>+Q():Q
<< create >>+Q(user:String,pwd:String,query:String):Q
+getUser():String
+setUser(user:String):void
+getPwd():String
+setPwd(pwd:String):void
+getQuery():String
+setQuery(query:String):void
#queryparm
1
MakeQuerySync
<< create >>+MakeQuerySync():MakeQuerySync
<< create >>+MakeQuerySync(queryparm:Query):MakeQuerySync
+getQueryparm():Query
+setQueryparm(queryparm:Query):void
Qr
#retCode:int
#XMLResult:String
<< create >>+Qr():Qr
<< create >>+Qr(retCode:int,XMLResult:String):Qr
+getRetCode():int
+setRetCode(retCode:int):void
+getXMLResult():String
+setXMLResult(XMLResult:String):void
<< interface >>
Query_SupportPortType
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):boolean
Query_SupportPortType_Impl
-axdbMappingUri:URI= null
<< create >>+Query_SupportPortType_Impl():Query_SupportPortType_Impl
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):boolean
# _return
1
1
Queryresults
<< create >>+Queryresults():Queryresults
<< create >>+Queryresults(_return:Qr):Queryresults
+get_return():Qr
+set_return(_return:Qr):void
async
<< interface >>
QuerySupport
+getQuery_SupportPortTypePort():Query_SupportPortType


Package it.exitech.axmedis.querysupport.server.async
LsClient
<< create >>+LsClient(endpointAddress:String):LsClient
+send(ec:int,result:String,id:String):boolean
listenerclient
- lsc
1
1
AXDBAsyncRespondingThread
-queryID:String= null
<< create >>+AXDBAsyncRespondingThread(rslt:Queryresults,lstUri:URI,id:String):AXDBAsyncRespondingThread
+run():void

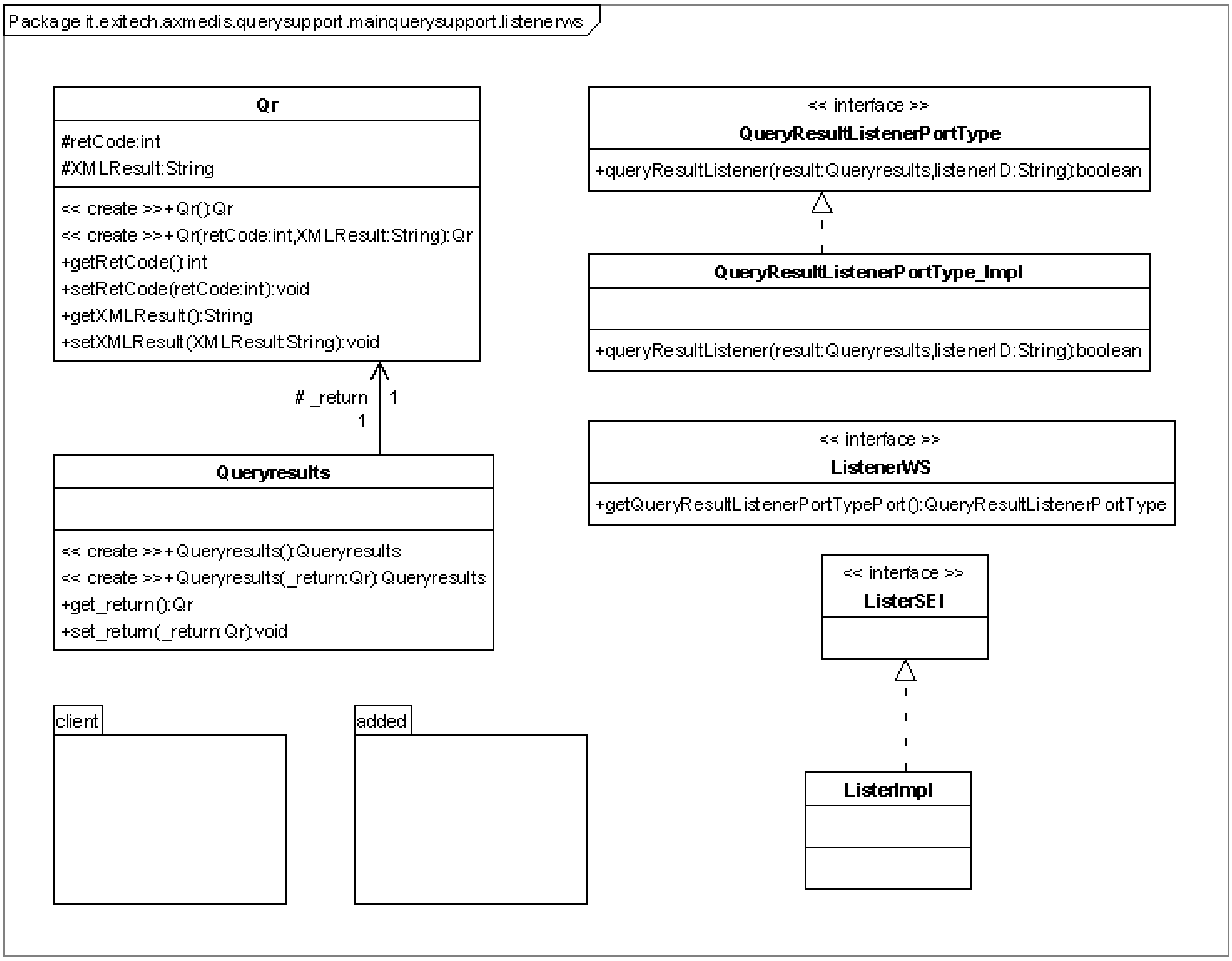
11.2.7 Package licence


Package it.exitech.axmedis.querysupport.licence.client.generated
Q
#user:String
#pwd:String
#query:String
<< create >>+Q():Q
<< create >>+Q(user:String,pwd:String,query:String):Q
+getUser():String
+setUser(user:String):void
+getPwd():String
+setPwd(pwd:String):void
+getQuery():String
+setQuery(query:String):void
Qr
#retCode:int
#XMLResult:String
<< create >>+Qr():Qr
<< create >>+Qr(retCode:int,XMLResult:String):Qr
+getRetCode():int
+setRetCode(retCode:int):void
+getXMLResult():String
+setXMLResult(XMLResult:String):void
# _return
1
1
Queryresults
<< create >>+Queryresults():Queryresults
<< create >>+Queryresults(_return:Qr):Queryresults
+get_return():Qr
+set_return(_return:Qr):void
<< interface >>
Query_Support
+getQuery_Support():Query_SupportPortType
MakeQueryASyncResponse
#result:boolean
<< create >>+MakeQueryASyncResponse():MakeQueryASyncResponse
<< create >>+MakeQueryASyncResponse(result:boolean):MakeQueryASyncResponse
+isResult():boolean
+setResult(result:boolean):void
Query_Support_Impl
-query_SupportPortType_PortClass:Class= it.exitech.axmedis.querysupport.licence.client.generated.Query_SupportPortType.class
<< create >>+Query_Support_Impl():Query_Support_Impl
+getPort(portName:QName,serviceDefInterface:Class):Remote
+getPort(serviceDefInterface:Class):Remote
+getQuery_Support():Query_SupportPortType
# send
1
1
Query
<< create >>+Query():Query
<< create >>+Query(send:Q):Query
+getSend():Q
+setSend(send:Q):void
<< interface >>
Query_SupportPortType
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):boolean
1
# querypam
1
# querypam
1
1
MakeQuerySync
<< create >>+MakeQuerySync():MakeQuerySync
<< create >>+MakeQuerySync(querypam:Query):MakeQuerySync
+getQueryparm():Query
+setQueryparm(querypam:Query):void
MakeQueryASync
#queryresultListenerService:URI
#listenerID:String
<< create >>+MakeQueryASync():MakeQueryASync
<< create >>+MakeQueryASync(querypam:Query,queryresultListenerService:URI,listenerID:String):MakeQueryASync
+getQueryparm():Query
+setQueryparm(querypam:Query):void
+getQueryresultListenerService():URI
+setQueryresultListenerService(queryresultListenerService:URI):void
+getListenerID():String
+setListenerID(listenerID:String):void
Qr_LiteralSerializer
<< create >>+Qr_LiteralSerializer(type:QName,encodingStyle:String):Qr_LiteralSerializer
<< create >>+Qr_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Qr_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Query_SupportPortType_Stub
-Make$2d$Query$2d$Sync_OPCODE:int= 0
-Make$2d$Query$2d$ASync_OPCODE:int= 1
-myNamespace_declarations:String[]=
newjava.lang.String[] {
"ns0", "urn:ax"
}
<< create >>+Query_SupportPortType_Stub(handlerChain:HandlerChain):Query_SupportPortType_Stub
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(querypam:Query,queryresultListenerService:URI,listenerID:String):boolean
#_readFirstBodyElement(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
-_deserialize_Make$2d$Query$2d$Sync(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
-_deserialize_Make$2d$Query$2d$ASync(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
#_getDefaultEnvelopeEncodingStyle():String
+_getImplicitEnvelopeEncodingStyle():String
+_getEncodingStyle():String
+_setEncodingStyle(encodingStyle:String):void
#_getNamespaceDeclarations():String[]
+_getUnderstoodHeaders():QName[]
+_initialize(registry:InternalTypeMappingRegistry):void
Query_LiteralSerializer
<< create >>+Query_LiteralSerializer(type:QName,encodingStyle:String):Query_LiteralSerializer
<< create >>+Query_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Query_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Queryresults_LiteralSerializer
<< create >>+Queryresults_LiteralSerializer(type:QName,encodingStyle:String):Queryresults_LiteralSerializer
<< create >>+Queryresults_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Queryresults_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
MakeQueryASync_LiteralSerializer
<< create >>+MakeQueryASync_LiteralSerializer(type:QName,encodingStyle:String):MakeQueryASync_LiteralSerializer
<< create >>+MakeQueryASync_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):MakeQueryASync_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Q_LiteralSerializer
<< create >>+Q_LiteralSerializer(type:QName,encodingStyle:String):Q_LiteralSerializer
<< create >>+Q_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Q_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
MakeQueryASyncResponse_LiteralSerializer
<< create >>+MakeQueryASyncResponse_LiteralSerializer(type:QName,encodingStyle:String):MakeQueryASyncResponse_LiteralSerializer
<< create >>+MakeQueryASyncResponse_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):MakeQueryASyncResponse_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
MakeQuerySync_LiteralSerializer
<< create >>+MakeQuerySync_LiteralSerializer(type:QName,encodingStyle:String):MakeQuerySync_LiteralSerializer
<< create >>+MakeQuerySync_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):MakeQuerySync_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Query_Support_SerializerRegistry
<< create >>+Query_Support_SerializerRegistry():Query_Support_SerializerRegistry
+getRegistry():TypeMappingRegistry
-registerSerializer(mapping:TypeMapping,javaType:Class,xmlType:QName,ser:Serializer):void

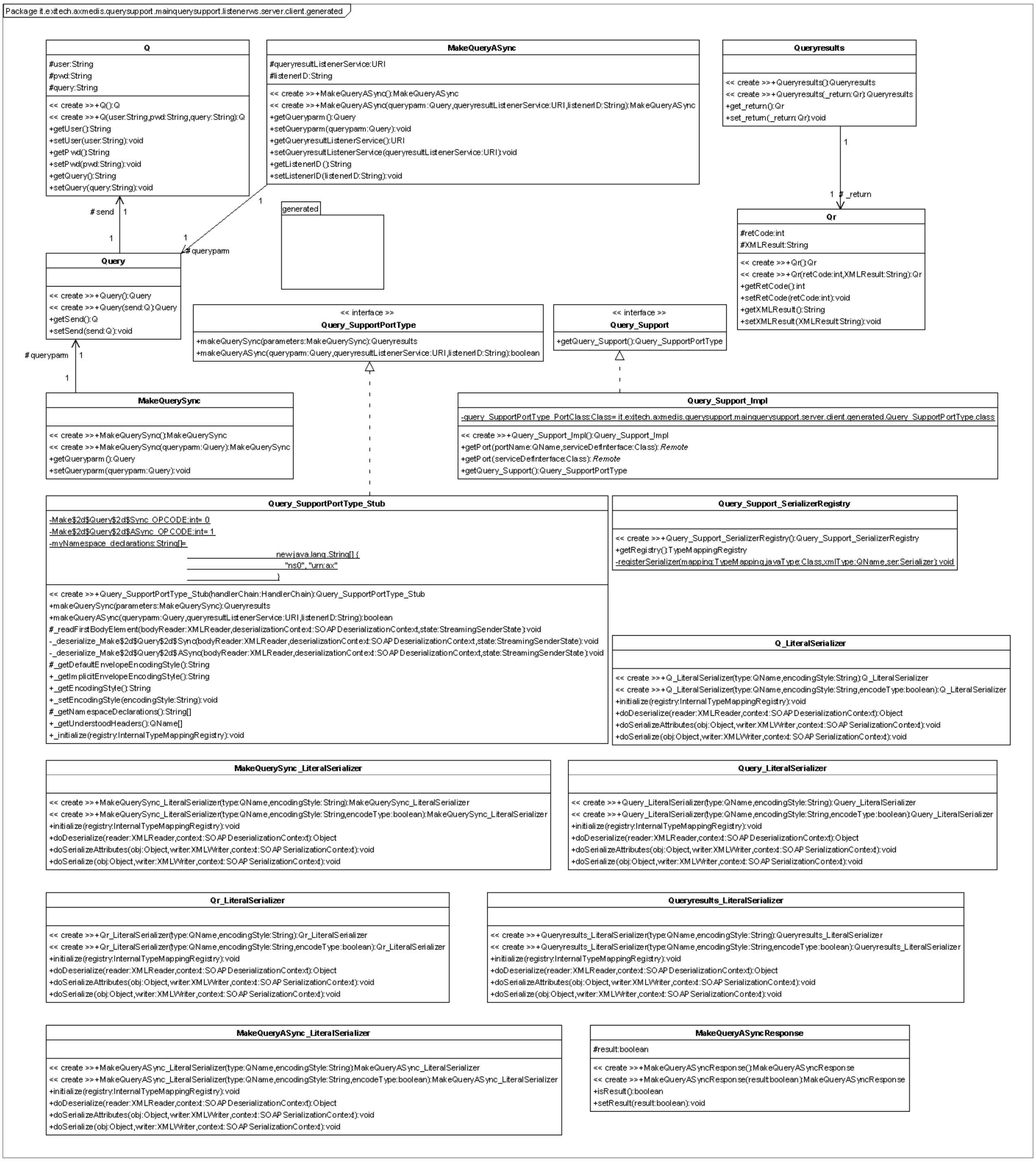
11.2.8 Package listenertable

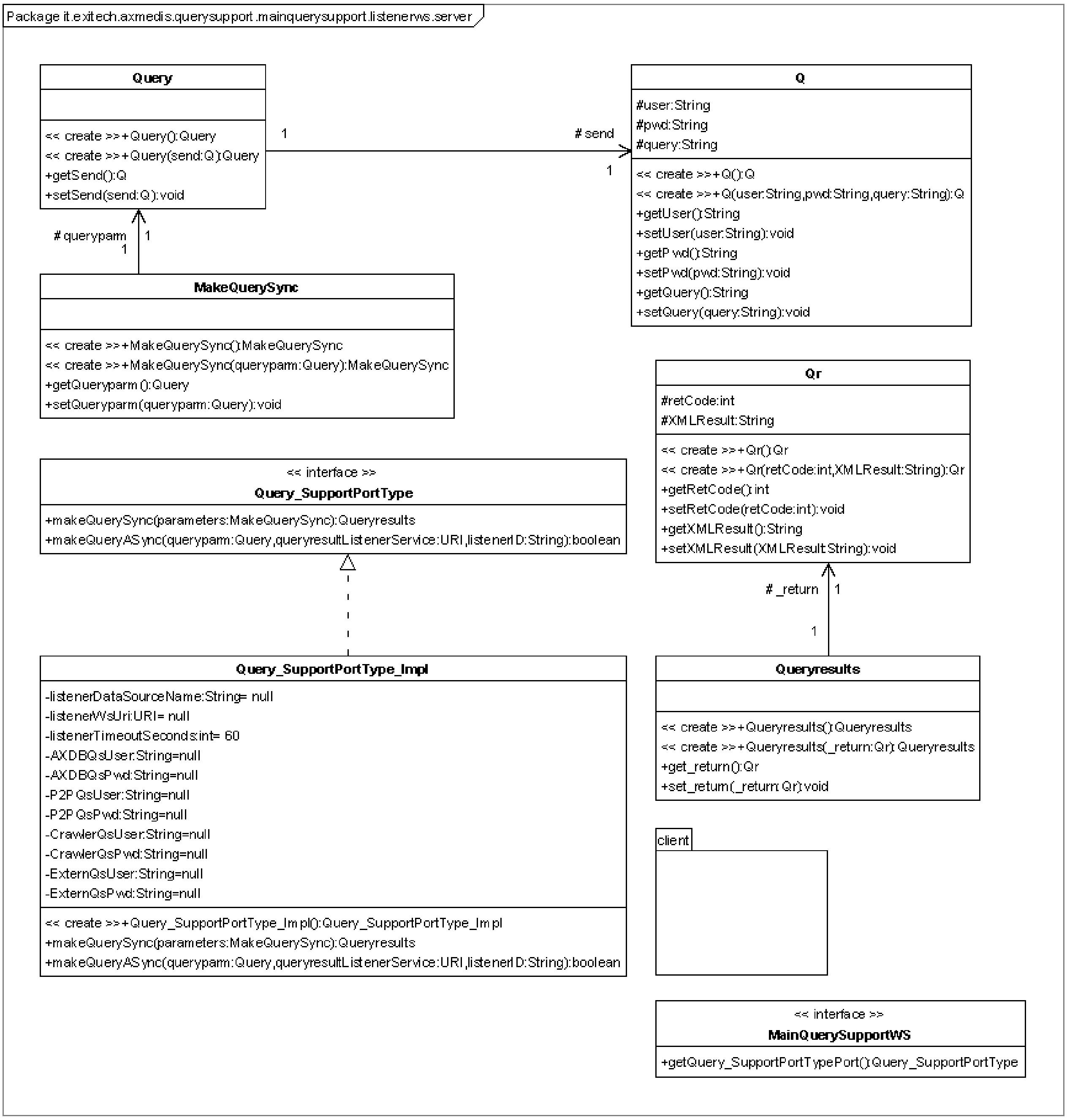

Package it.exitech.axmedis.querysupport.mainquerysupport.listenertable
ListenerTableManager
-monitor:Object= new String("Sono un monitor.")
-timeoutSeconds:int= 0
<< create >>+ListenerTableManager(dsRef:String,timeoutSecs:int):ListenerTableManager
+store(key:String,id:String,uri:String,axdbend:boolean,p2pend:boolean,crawlerend:boolean,extend:boolean):boolean
#manageTimeout(key:String):void
+manageResult(informedKey:String,res:Queryresults):void
-remove(key:String,conn: Connection):void
-getUri(key:String,conn: Connection):String
-getId(key:String,conn: Connection):String
-isPresent(key:String,conn: Connection):boolean
-setEnd(sr:String,key:String,conn: Connection):boolean
-isEnded(key:String,conn: Connection):boolean
- ltManager
1
1
TimeoutManager
-ky:String= null
-timer:Timer= null
<< create >>+TimeoutManager(seconds:int,ltmg:ListenerTableManager,key:String):TimeoutManager

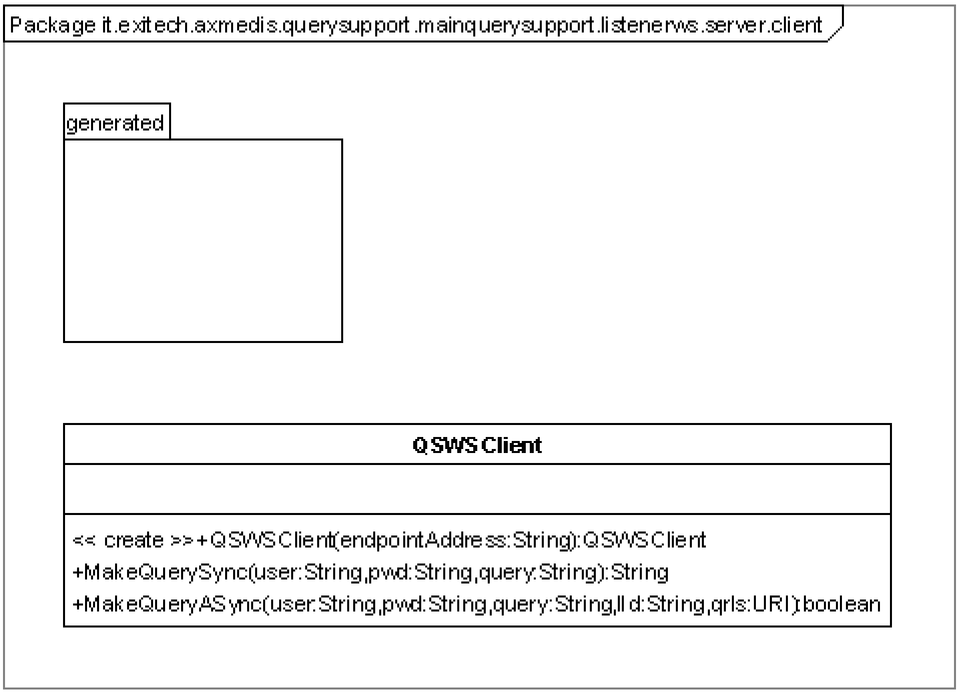
11.2.9 Package listenerws


Package it.exitech.axmedis.querysupport.mainquerysupport.listenerws
Qr
#retCode:int
#XMLResult:String
<< create >>+Qr():Qr
<< create >>+Qr(retCode:int,XMLResult:String):Qr
+getRetCode():int
+setRetCode(retCode:int):void
+getXMLResult():String
+setXMLResult(XMLResult:String):void
# _return
1
1
Queryresults
<< create >>+Queryresults():Queryresults
<< create >>+Queryresults(_return:Qr):Queryresults
+get_return():Qr
+set_return(_return:Qr):void
<< interface >>
QueryResultListenerPortType
+queryResultListener(result:Queryresults,listenerID:String):boolean
QueryResultListenerPortType_Impl
+queryResultListener(result:Queryresults,listenerID:String):boolean
<< interface >>
ListenerWS
+getQueryResultListenerPortTypePort():QueryResultListenerPortType
<< interface >>
ListerSEI
ListerImpl
client
added

11.2.10 Package server


Package it.exitech.axmedis.querysupport.mainquerysupport.listenerws.server.client.generated
Q
#user:String
#pwd:String
#query:String
<< create >>+Q():Q
<< create >>+Q(user:String,pwd:String,query:String):Q
+getUser():String
+setUser(user:String):void
+getPwd():String
+setPwd(pwd:String):void
+getQuery():String
+setQuery(query:String):void
MakeQueryASync
#queryresultListenerService:URI
#listenerID:String
<< create >>+MakeQueryASync():MakeQueryASync
<< create >>+MakeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):MakeQueryASync
+getQueryparm():Query
+setQueryparm(queryparm:Query):void
+getQueryresultListenerService():URI
+setQueryresultListenerService(queryresultListenerService:URI):void
+getListenerID():String
+setListenerID(listenerID:String):void
Queryresults
<< create >>+Queryresults():Queryresults
<< create >>+Queryresults(_return:Qr):Queryresults
+get_return():Qr
+set_return(_return:Qr):void
1
1
# _return
# send
1
1
1
1
# queryparm
generated
Qr
#retCode:int
#XMLResult:String
<< create >>+Qr():Qr
<< create >>+Qr(retCode:int,XMLResult:String):Qr
+getRetCode():int
+setRetCode(retCode:int):void
+getXMLResult():String
+setXMLResult(XMLResult:String):void
Query
<< create >>+Query():Query
<< create >>+Query(send:Q):Query
+getSend():Q
+setSend(send:Q):void
<< interface >>
Query_SupportPortType
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):boolean
<< interface >>
Query_Support
+getQuery_Support():Query_SupportPortType
# queryparm
1
1
MakeQuerySync
<< create >>+MakeQuerySync():MakeQuerySync
<< create >>+MakeQuerySync(queryparm:Query):MakeQuerySync
+getQueryparm():Query
+setQueryparm(queryparm:Query):void
Query_Support_Impl
-query_SupportPortType_PortClass:Class= it.exitech.axmedis.querysupport.mainquerysupport.server.client.generated.Query_SupportPortType.class
<< create >>+Query_Support_Impl():Query_Support_Impl
+getPort(portName:QName,serviceDefInterface:Class): Remote
+getPort(serviceDefInterface:Class): Remote
+getQuery_Support():Query_SupportPortType
Query_SupportPortType_Stub
-Make$2d$Query$2d$Sync_OPCODE:int= 0
-Make$2d$Query$2d$ASync_OPCODE:int= 1
-myNamespace_declarations:String[]=
new java.lang.String[] {
"ns0", "urn:ax"
}
<< create >>+Query_SupportPortType_Stub(handlerChain:HandlerChain):Query_SupportPortType_Stub
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):boolean
#_readFirstBodyElement(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
-_deserialize_Make$2d$Query$2d$Sync(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
-_deserialize_Make$2d$Query$2d$ASync(bodyReader:XMLReader,deserializationContext:SOAPDeserializationContext,state:StreamingSenderState):void
#_getDefaultEnvelopeEncodingStyle():String
+_getImplicitEnvelopeEncodingStyle():String
+_getEncodingStyle():String
+_setEncodingStyle(encodingStyle:String):void
#_getNamespaceDeclarations():String[]
+_getUnderstoodHeaders():QName[]
+_initialize(registry:InternalTypeMappingRegistry):void
Query_Support_SerializerRegistry
<< create >>+Query_Support_SerializerRegistry():Query_Support_SerializerRegistry
+getRegistry():TypeMappingRegistry
-registerSerializer(mapping:TypeMapping,javaType:Class,xmlType:QName,ser:Serializer):void
Q_LiteralSerializer
<< create >>+Q_LiteralSerializer(type:QName,encodingStyle:String):Q_LiteralSerializer
<< create >>+Q_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Q_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
MakeQuerySync_LiteralSerializer
<< create >>+MakeQuerySync_LiteralSerializer(type:QName,encodingStyle:String):MakeQuerySync_LiteralSerializer
<< create >>+MakeQuerySync_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):MakeQuerySync_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Query_LiteralSerializer
<< create >>+Query_LiteralSerializer(type:QName,encodingStyle:String):Query_LiteralSerializer
<< create >>+Query_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Query_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Qr_LiteralSerializer
<< create >>+Qr_LiteralSerializer(type:QName,encodingStyle:String):Qr_LiteralSerializer
<< create >>+Qr_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Qr_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
Queryresults_LiteralSerializer
<< create >>+Queryresults_LiteralSerializer(type:QName,encodingStyle:String):Queryresults_LiteralSerializer
<< create >>+Queryresults_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):Queryresults_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
MakeQueryASync_LiteralSerializer
<< create >>+MakeQueryASync_LiteralSerializer(type:QName,encodingStyle:String):MakeQueryASync_LiteralSerializer
<< create >>+MakeQueryASync_LiteralSerializer(type:QName,encodingStyle:String,encodeType:boolean):MakeQueryASync_LiteralSerializer
+initialize(registry:InternalTypeMappingRegistry):void
+doDeserialize(reader:XMLReader,context:SOAPDeserializationContext):Object
+doSerializeAttributes(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
+doSerialize(obj:Object,writer:XMLWriter,context:SOAPSerializationContext):void
MakeQueryASyncResponse
#result:boolean
<< create >>+MakeQueryASyncResponse():MakeQueryASyncResponse
<< create >>+MakeQueryASyncResponse(result:boolean):MakeQueryASyncResponse
+isResult():boolean
+setResult(result:boolean):void


Package it.exitech.axmedis.querysupport.mainquerysupport.listenerws.server
Query
<< create >>+Query():Query
<< create >>+Query(send:Q):Query
+getSend():Q
+setSend(send:Q):void
1
# send
1
Q
#user:String
#pwd:String
#query:String
<< create >>+Q():Q
<< create >>+Q(user:String,pwd:String,query:String):Q
+getUser():String
+setUser(user:String):void
+getPwd():String
+setPwd(pwd:String):void
+getQuery():String
+setQuery(query:String):void
# queryparm
1
1
MakeQuerySync
<< create >>+MakeQuerySync():MakeQuerySync
<< create >>+MakeQuerySync(queryparm:Query):MakeQuerySync
+getQueryparm():Query
+setQueryparm(queryparm:Query):void
Qr
#retCode:int
#XMLResult:String
<< create >>+Qr():Qr
<< create >>+Qr(retCode:int,XMLResult:String):Qr
+getRetCode():int
+setRetCode(retCode:int):void
+getXMLResult():String
+setXMLResult(XMLResult:String):void
<< interface >>
Query_SupportPortType
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):boolean
# _return
1
1
Query_SupportPortType_Impl
-listenerDataSourceName:String= null
-listenerWsUri:URI= null
-listenerTimeoutSeconds:int= 60
-AXDBQsUser:String=null
-AXDBQsPwd:String=null
-P2PQsUser:String=null
-P2PQsPwd:String=null
-CrawlerQsUser:String=null
-CrawlerQsPwd:String=null
-ExternQsUser:String=null
-ExternQsPwd:String=null
<< create >>+Query_SupportPortType_Impl():Query_SupportPortType_Impl
+makeQuerySync(parameters:MakeQuerySync):Queryresults
+makeQueryASync(queryparm:Query,queryresultListenerService:URI,listenerID:String):boolean
Queryresults
<< create >>+Queryresults():Queryresults
<< create >>+Queryresults(_return:Qr):Queryresults
+get_return():Qr
+set_return(_return:Qr):void
client
<< interface >>
MainQuerySupportWS
+getQuery_SupportPortTypePort():Query_SupportPortType


Package it.exitech.axmedis.querysupport.mainquerysupport.listenerws.server.client
generated
QSWS Client
<< create >>+QSWSClient(endpointAddress:String):QSWSClient
+MakeQuerySync(user:String,pwd:String,query:String):String
+MakeQueryASync(user:String,pwd:String,query:String,lId:String,qrls:URI):boolean

11.3 User interface description

User interface is not present since it is a webservice module.

11.4 Technical and Installation information

In this section it is reported a draft configuration manual for installing the service as it is provided by EXITECH in the SVN repository of the project. Integration to this manual have to be done by DSI for Crawling and for P2P.

11.4.1 Prerequisites

In this manual is it assumed that you install all in 1 WINDOWS server and that in this server you have:

- Mysql 4.1.10 or higher but not mysql 5.x installed and working
- Tomcat 5.5.x installed and working and respond on port 8080
- IIS (or Apache) installed and working and respond on port 80
-

It is assumed also that:

- your Tomcat is installed in $TOMCAT (say C:\Programmi\Apache Software Foundation\Tomcat 5.5, or something like that);
- the document root of your web server is located in $HTDOCS (say C:\inetpub\wwwroot, or something similar);
- your local copy of the AXMEDIS SVN repository is in $SVNROOT;
- you have got from the SVN repository the following files:
    - $SVNROOT\WebServices\AxdbQSWs\bin\Tomcat55\AXDBQsWS.war
    - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\LicenseWs.war
    - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\metadata-mapper.xml
    - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\Tomcat-shared-libs.zip
    - $SVNROOT\WebServices\QuerySupportWS\bin\Tomcat55\MainQuerySupportWS.war
    - $SVNROOT\WebServices\QuerySupportWS\doc\test\CrawlerWS.war
    - $SVNROOT\WebServices\QuerySupportWS\doc\test\ExternalQuerySupportWS.war

    - $SVNROOT\WebServices\QuerySupportWS\doc\test\P2PWS.war
    - $SVNROOT\WebServices\QuerySupportWS\doc\test\QSClient
- you have created under $HTDOCS the directories:
    - axmedis

If you made different assumption follow the guidelines and substitute the proposed path and names with the ones you chose.

11.4.2 Installation of AXMEDIS Database (AXDB)

See Section 5.4.2

11.4.3 Query Support Web Services

11.4.3.1 Installation of the WebServices

You have to copy **metadata-mapper.xml**, in your $HTDOCS\axmedis.
Before deploying the web services you need to prepare Tomcat in order to allow it to use JWSDP. If not already done for other webservices, unpack the Tomcat-shared-libs.zip archive in $TOMCAT\shared\libs.
Restart Tomcat.
Now we will start to install the webservices that do not require special configuration. These web services are:

- CrawlerWS.war (that is a fake webservice for Crawler, if you have the real one, please follow its installation instruction)
- P2PWS.war (that is a fake webservice for AXEPTOOL of the P2P network, you can avoid to install it if you point directly to the P2P service provided by the axmedis consortium)
- ExternalQuerySupportWS.war (that is a fake webservice for kiosk that simulate the kiosk factory web service, if you have the real one please follow its installation instruction)

Connect to your Tomcat at http://localhost:8080 and click on the "Tomcat Manager" link.
Use the "WAR file to deploy" section to select the cited .war and to deploy them.
You can verify if the services are working by checking them at:
http://localhost:8080/ExternalQuerySupportWS/ExternalQuerySupportWS
something like that should appear:

Web Services

Port Name	Status	Information	
ExternalQuery SupportWSd	ACTIVE	Address:	http://192.168.1.81:8080/ ExternalQuerySupportWS/ExternalQuerySupportWS
		WSDL:	http://192.168.1.81:8080/ExternalQuerySupptWS?WSDL
		Port QName:	{http://www.axmedis.org/ query_support.wsdl}Query_SupportPortTypePort
		Remote interface:	it.exitech.axmedis.querysupport. externalquerysupportws.server.Query_SupportPortType
		Implementation class:	it.exitech.axmedis.querysupport. externalquerysupportws.server.Query_SupportPortType_Impl
		Model:	http://192.168.1.81:8080/ ExternalQuerySupportWS/ExternalQuerySupportWS?model

(please note that return have been added to make them readable)

something like that should appear:

Web Services

Port	Status	Information

Name		
P2PWSd	ACTIVE	Address: http://192.168.1.81:8080/P2PWS/P2PWS WSDL: http://192.168.1.81:8080/P2PWS/P2PWS?WSDL Port QName: {http://www.axmedis.org/query_support.wsdl}Query_SupportPortTypePort Remote interface: it.exitech.axmedis.querysupport.p2p.server.Query_SupportPortType Implementation class: it.exitech.axmedis.querysupport.p2p.server.Query_SupportPortType_Impl Model: http://192.168.1.81:8080/P2PWS/P2PWS?model

http://localhost:8080/CrawlerWS/CrawlerWS
something like that should appear:

Web Services

Port Name	Status	Information
CrawlerD	ACTIVE	Address: http://192.168.1.81:8080/CrawlerWS/CrawlerWS WSDL: http://192.168.1.81:8080/CrawlerWS/CrawlerWS?WSDL Port QName: {http://www.axmedis.org/query_support.wsdl}Query_SupportPortTypePort Remote interface: it.exitech.axmedis.querysupport.crawler.server.Query_SupportPortType Implementation class: it.exitech.axmedis.querysupport.crawler.server.Query_SupportPortType_Impl Model: http://192.168.1.81:8080/CrawlerWS/CrawlerWS?model

11.4.3.2 UPC PAR Query Support [UPC]

Now you are ready to install PARQuerySupport that is the Webservice for UPC PAR database.

If you have followed all the prerequisites you have simply to deploy the **PARQuerySupport.war**, otherwise modify the Axlicdb.properties file in **PARQuerySupport.war/WEB-INF/classes** directory by setting the correct values for AxlicdbUrl, AxlicdbUser, AxlicdbPwd, that are the URL of the database, the user allowed to access to the AXDB and its password.

Verify the web service at the address http://localhost:8080/PARQuerySupport/PARQuerySupport, something like that should appear:

Web Services

Port Name	Status	Information
PARQuery SupportWS	ACTIVE	Address: http://192.168.45.173:8080/PARQuerySupport/PARQuerySupport WSDL: http://192.168.45.173:8080/PARQuerySupport/PARQuerySupport?WSDL Port QName: {http://www.axmedis.org/query_support.wsdl}Query_SupportPortTypePort Remote interface: es.fupf.dmag.axmedis.QuerySupport.Query_SupportPortType Implementation class: es.fupf.dmag.axmedis.QuerySupport.PARQuerySupport_Impl

		Model:	http://192.168.45.173:8080/PARQuerySupport/PARQuerySupport?model

Now we are ready to install the AXMEDIS database Query support web service that is the service the will resolve the queries on AXDB.
Edit the **axdb.properties** inside **AXDBQsWS.war\WEB-INF\classes** and change if needed axdbUrl, axdbUser and axdbPwd. After that change axdbMapping in order to point to http://localhost/axmedis/metadata-mapper.xml or to the path where you copied metadata-mapped.xml.
Update the archive and open **licenceWS.properties** inside **AXDBQsWS.war\WEB-INF\classes** and change (if needed) the path where the LicenseWs is. If you have followed all the guidelines it should be http://localhost:8080/LicenseWs/license

Deploy as usual the modified **AXDBQsWS.war**. By connecting to http://localhost:8080/AXDBQsWS/QuerySupport something like the following screen should appear:

Web Services

Port Name	Status	Information	
AXDBQuerySupportWs	ACTIVE	Address:	http://192.168.1.81:8080/AXDBQsWS/QuerySupport
		WSDL:	http://192.168.1.81:8080/AXDBQsWS/QuerySuWSDL
		Port QName:	{http://www.axmedis.org/query_support.wsdl} Query_SupportPortTypePort
		Remote interface:	it.exitech.axmedis.querysupport.server. Query_SupportPortType
		Implementation class:	it.exitech.axmedis.querysupport.server. Query_SupportPortType_Impl
		Model:	http://192.168.1.81:808DBQsWS/QuerySupport?model

Now queries on AXDB can be issued, but in order to have the full service, it is necessary to deploy also the **MainQuerySupportWS.war** that is the dispatcher to the different web services (crawler, db, P2P, etc).
Edit the MainQuerySupportWS.properties file inside **MainQuerySupportWS.war/WEB-INF/classes** and verify the endopoint to the previously installed services.
In the same file remember to change also the following parameters in order to allow connection to Qs Webservice protected by username and password:

```
AXDBQsUser=test
AXDBQsPwd=test

P2PQsUser=test
P2PQsPwd=test

CrawlerQsUser=user
CrawlerQsPwd=axmedis

ExternQsUser=test
ExternQsPwd=test
```

Last but not least deploy **MainQuerySupportWS.war** as usual.
By pointing to http://localhost:8080/MainQuerySupportWs/mqs something like that should appear:

Web Services

Port Name	Status	Information
MainQuerySupportWS	ACTIVE	Address: http://192.168.1.81:8080/MainQuerySupportWs/mqs WSDL: http://192.168.1.81:8080/MainQuerySupportWs/mqs?WSDL Port QName: {http://www.axmedis.org/query_support.wsdl}Query_SupportPortTypePort Remote interface: it.exitech.axmedis.querysupport.mainquerysupport.server.Query_SupportPortType Implementation class: it.exitech.axmedis.querysupport.mainquerysupport.server.Query_SupportPortType_Impl Model: http://192.168.1.81:8080/MainQuerySupportWs/mqs?model
ListenerWS	ACTIVE	Address: http://192.168.1.81:8080/MainQuerySupportWs/ListenerWS WSDL: http://192.168.1.81:8080/MainQuerySupportWs/ListenerWS?WSDL Port QName: {http://www.someone.org/querylistener.wsdl}QueryResultListenerPortTypePort Remote interface: it.exitech.axmedis.querysupport.mainquerysupport.listenerws.QueryResultListenerPortType Implementation class: it.exitech.axmedis.querysupport.mainquerysupport.listenerws.QueryResultListenerPortType_Impl Model: http://192.168.1.81:8080/MainQuerySupportWs/ListenerWS?model

(please note that return have been added to make them readable)

11.4.3.3 Test of the Query support

In order to test your installation with the pre-filled database, you need to issue with a client the following query, after putting inside the database the object of the collection provided by Tiscali:

```
<?xml version="1.0" encoding="UTF-8"?>
<query xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://localhost/axQUERY-v1-6.xsd">
<source>
<location>AXDB</location>
</source>
<AXinfoQuery>
<querycondition>
<nesting>
<test>
<field>DCMI:type</field>
<operator>STARTWITH</operator>
<value>commedia</value>
</test>
```

```
</nesting>
</querycondition>
</AXinfoQuery>
</query>
```

11.5 Draft User Manual

No user manual is reported since this is not usable directly from the end user. All info are reported in the previous section where the installation manual is presented.

11.6 Examples of usage

Examples of usage are reported in the installation manual in Section 11.4

11.7 Integration and compilation issues

The module is compatible with JDK 1.5, JWSDP 1.6 and Tomcat 5.5. In the repository the project for NetBeans 5.5 is reported.

11.8 Configuration Parameters

All the configuration parameters are reported in the section 11.4.3.1

12 User Selection Archive (EXITECH)

Module/Tool Profile		
User Selection Archive		
Responsible Name	Fioravanti	
Responsible Partner	EXITECH	
Status (proposed/approved)	Approved	
Implemented/not implemented	Implemented	
Status of the implementation	Completed, under maintenance	
Executable or Library/module (Support)	Web service	
Single Thread or Multithread	Multithread	
Language of Development	JAVA	
Platforms supported	All supported by JDK 1.5 and Tomcat 5.5	
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/WebServices/UserSelectionArchive/	
Reference to the AXFW location of the demonstrator executable tool for internal download	none	
Reference to the AXFW location of the demonstrator executable tool for public download	none	
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	http://81.73.104.125:8080/SelectionWS/sa	
Test cases (present/absent)	absent	
Test cases location	none	
Usage of the AXMEDIS configuration manager (yes/no)	no	
Usage of the AXMEDIS Error Manager (yes/no)	no	
Major Problems not solved	-- --	
Major pending requirements	-- --	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section

Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

12.1 General Description of the Module

This module takes care of storing the query and the selection of each user in order to have a personal archive for each user that want to store his/her queries and selection for future reuse. The query or the selection must have some characteristics in order to be easily identified by the user. It can be supposed to have the following data stored with each query or selection:

- User that has created the query or selection
- ID of the query or selection (Mandatory);
- Name of the query or selection assigned by the user (Mandatory): this is for allowing the user to assign a symbolic name to the query for a future fast recovery without browsing all the items stored in the personal repository;
- Timestamp of the query or selection assigned by the user (Mandatory): this is useful for expanded or actualized selections since people can have more that one snapshot of the same selection;
- List of groups of users that are entitled to operate on the selection (Optional);
- List of keyword (Optional): the user assign a list of keywords for recovering queries and selection by keywords instead of directly pointing to a selection;

The described parameters allow the maximum flexibility in browsing, recovering and selecting query and selections.

Query and Selection Archive for Each User has to provide some services for allowing the user to interact with selection stored in each personal profile.

Since Query can be regarded as a selection with only a query inside, all queries will be stored as selections.

A first definition of the services provided is:

- List all selection created by the user;
- List all selection for which the user is entitled (this is because a user can create a selection that can be managed by a group of users);
- Load a selection;
- Save a selection (several selection with the same name but different timestamps can exist in the database;
- Delete a selection;

This module offers its service with a web service interface that operates according to the methods described in the following, while the WSDL is reported in section 35.

Selection_Archive	
Method	***Description***
ListUserSelection	This method will return all the selections created by the user
ListEntitledSelection	This method will return all the selections for which the user is entitled
LoadSelection	This method get a selection from the AXDB
SaveSelection	This method save a selection to the AXDB
DeleteSelection	This method eliminate a selection for the AXDB
ActualizeSelection_sync	This methods return synchronously the actualized selection that corresponds to the selection given as input parameter.
ActualizeSelection_async	This methods return asynchronously the actualized selection that corresponds to the selection given as input parameter.

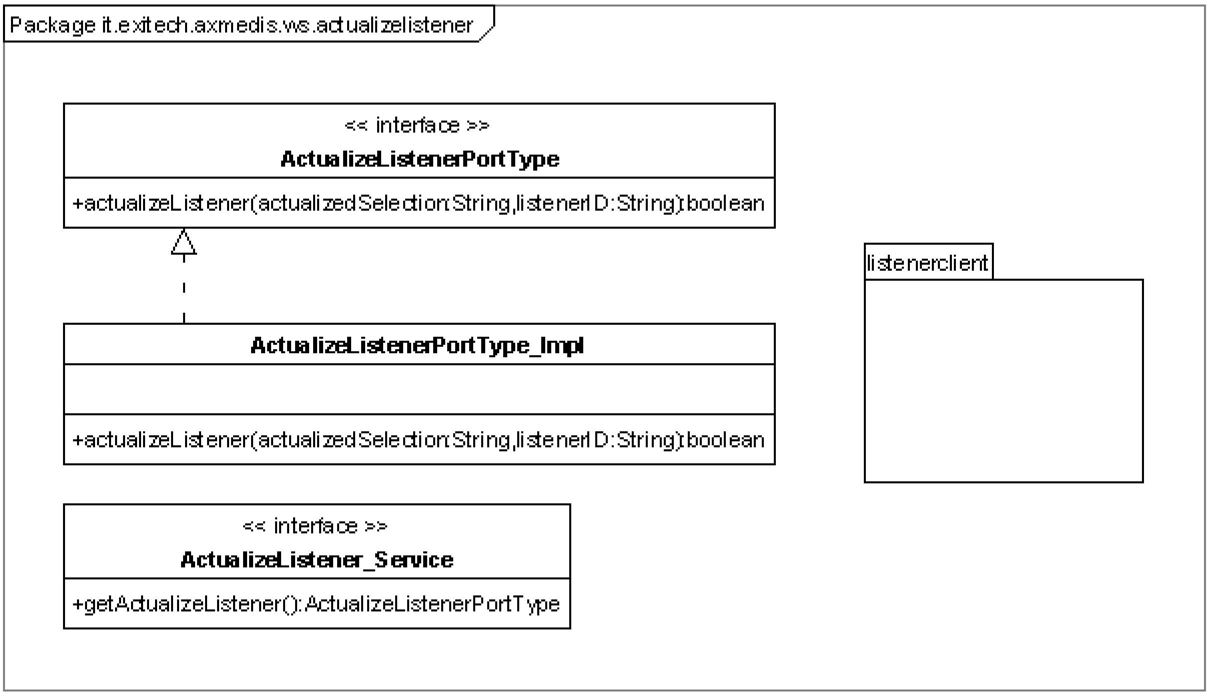
12.1.1 Actualize Listener

In this paragraph a model for the listener to be implemented is present. It will have only one method named Actualize_Listener to which all the results will be communicated as soon as they are ready. The specification in terms of WSDL is reported in Section 36

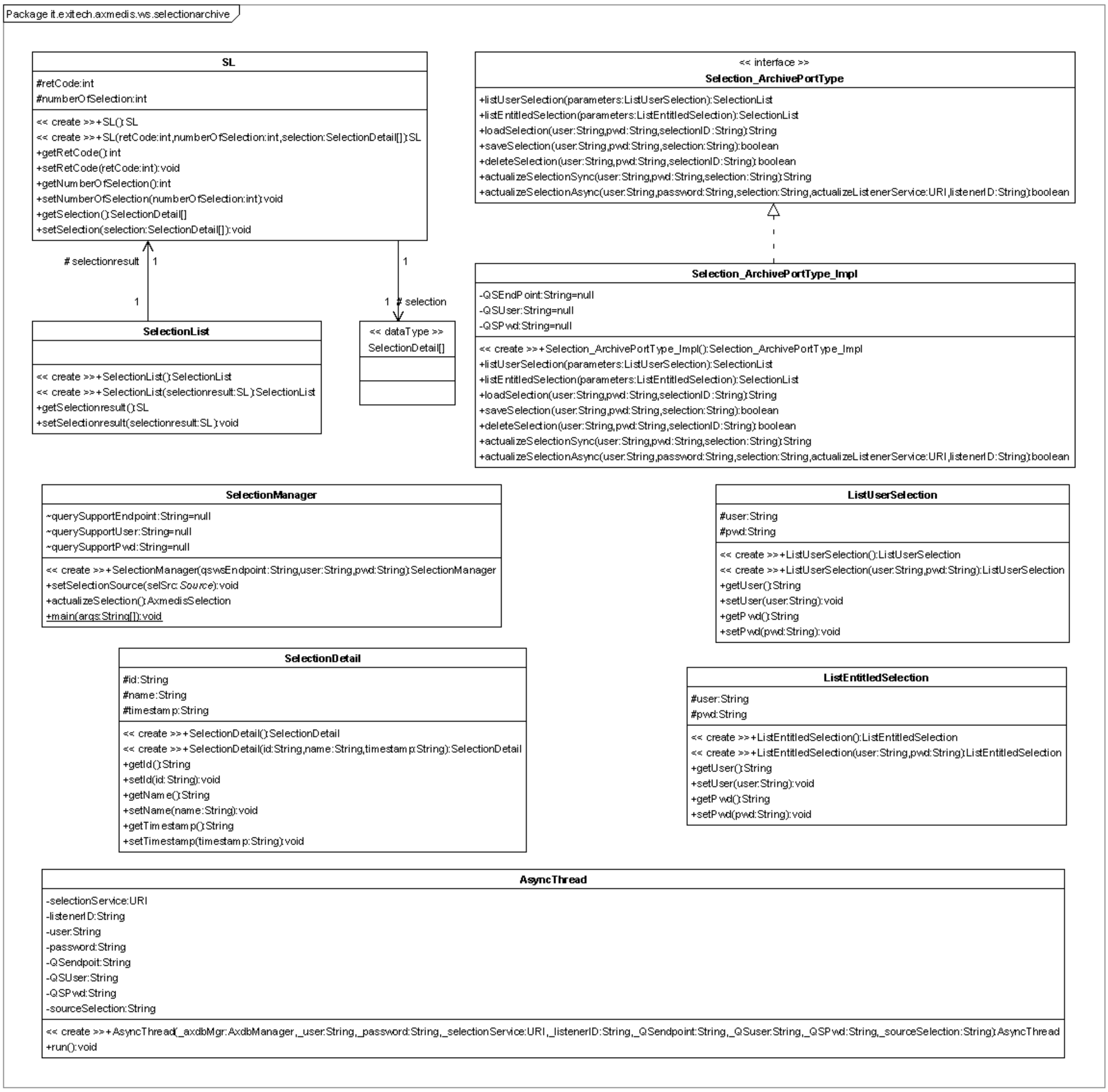
12.2 Module Design in terms of Classes

This webservice uses the client package of the mainquerysupport web service in order to actualize selections, while the code of the service is developed in 2 packages under the it.exitech.axmedis.ws package that are the actualizelistener package that is related to the implementation of a test listener server and the selectionarchive that is the real implementation of the web service.

12.2.1 Package actualizelistener


Package it.exitech.axmedis.ws.actualizelistener
<< interface >>
ActualizeListenerPortType
+actualizeListener(actualizedSelection:String,listenerID:String):boolean
ActualizeListenerPortType_Impl
+actualizeListener(actualizedSelection:String,listenerID:String):boolean
<< interface >>
ActualizeListener_Service
+getActualizeListener():ActualizeListenerPortType
listenerclient

12.2.2 Package selectionarchive


Package it.exitech.axmedis.ws.selectionarchive
SL
#retCode:int
#numberOfSelection:int
<< create >>+SL():SL
<< create >>+SL(retCode:int,numberOfSelection:int,selection:SelectionDetail[]):SL
+getRetCode():int
+setRetCode(retCode:int):void
+getNumberOfSelection():int
+setNumberOfSelection(numberOfSelection:int):void
+getSelection():SelectionDetail[]
+setSelection(selection:SelectionDetail[]):void
# selectionresult
1
1
1
1
# selection
SelectionList
<< create >>+SelectionList():SelectionList
<< create >>+SelectionList(selectionresult:SL):SelectionList
+getSelectionresult():SL
+setSelectionresult(selectionresult:SL):void
<< dataType >>
SelectionDetail[]
<< interface >>
Selection_ArchivePortType
+listUserSelection(parameters:ListUserSelection):SelectionList
+listEntitledSelection(parameters:ListEntitledSelection):SelectionList
+loadSelection(user:String,pwd:String,selectionID:String):String
+saveSelection(user:String,pwd:String,selection:String):boolean
+deleteSelection(user:String,pwd:String,selectionID:String):boolean
+actualizeSelectionSync(user:String,pwd:String,selection:String):String
+actualizeSelectionAsync(user:String,password:String,selection:String,actualizeListenerService:URI,listenerID:String):boolean
Selection_ArchivePortType_Impl
-QSEndPoint:String=null
-QSUser:String=null
-QSPwd:String=null
<< create >>+Selection_ArchivePortType_Impl():Selection_ArchivePortType_Impl
+listUserSelection(parameters:ListUserSelection):SelectionList
+listEntitledSelection(parameters:ListEntitledSelection):SelectionList
+loadSelection(user:String,pwd:String,selectionID:String):String
+saveSelection(user:String,pwd:String,selection:String):boolean
+deleteSelection(user:String,pwd:String,selectionID:String):boolean
+actualizeSelectionSync(user:String,pwd:String,selection:String):String
+actualizeSelectionAsync(user:String,password:String,selection:String,actualizeListenerService:URI,listenerID:String):boolean
SelectionManager
~querySupportEndpoint:String=null
~querySupportUser:String=null
~querySupportPwd:String=null
<< create >>+SelectionManager(qswsEndpoint:String,user:String,pwd:String):SelectionManager
+setSelectionSource(selSrc:Source):void
+actualizeSelection():AxmedisSelection
+main(args:String[]):void
ListUserSelection
#user:String
#pwd:String
<< create >>+ListUserSelection():ListUserSelection
<< create >>+ListUserSelection(user:String,pwd:String):ListUserSelection
+getUser():String
+setUser(user:String):void
+getPwd():String
+setPwd(pwd:String):void
SelectionDetail
#id:String
#name:String
#timestamp:String
<< create >>+SelectionDetail():SelectionDetail
<< create >>+SelectionDetail(id:String,name:String,timestamp:String):SelectionDetail
+getId():String
+setId(id:String):void
+getName():String
+setName(name:String):void
+getTimestamp():String
+setTimestamp(timestamp:String):void
ListEntitledSelection
#user:String
#pwd:String
<< create >>+ListEntitledSelection():ListEntitledSelection
<< create >>+ListEntitledSelection(user:String,pwd:String):ListEntitledSelection
+getUser():String
+setUser(user:String):void
+getPwd():String
+setPwd(pwd:String):void
AsyncThread
-selectionService:URI
-listenerID:String
-user:String
-password:String
-QSendpoit:String
-QSUser:String
-QSPwd:String
-sourceSelection:String
<< create >>+AsyncThread(_axdbMgr:AxdbManager,_user:String,_password:String,_selectionService:URI,_listenerID:String,_QSendpoint:String,_QSuser:String,_QSPwd:String,_sourceSelection:String):AsyncThread
+run():void

12.3 User interface description

No user interface has been provided for this back-office module

12.4 Technical and Installation information

In this section it is reported a draft configuration manual for installing the service as it is provided by EXITECH in the SVN repository of the project.

12.4.1 Prerequisites

In this manual is it assumed that you install all in one WINDOWS server and that in this server you have:

- Mysql 4.1.10 or higher but not mysql 5.x installed and working
- Tomcat 5.5.x installed and working and respond on port 8080

It is assumed also that:

- your Tomcat is installed in $TOMCAT (say C:\Programmi\Apache Software Foundation\Tomcat 5.5, or something like that);
- your local copy of the AXMEDIS SVN repository is in $SVNROOT;
- you have got from the SVN repository the following files:
  - $SVNROOT\WebServices\WebServices/UserSelectionArchive/bin/Tomcat5.5/SelectionWS.war
  - $SVNROOT\WebServices\AxdbQSWs\doc\configuration-deployment\Tomcat-shared-libs.zip

If you made different assumption follow the guidelines and substitute the proposed path and names with the ones you chose.

12.4.2 Installation of AXMEDIS Database (AXDB)

See Section 5.4.2

12.4.3 Selection Web Services

If not already done, before deploying the web services you need to prepare Tomcat in order to allow it to use JWSDP. Unpack the Tomcat-shared-libs.zip archive in $TOMCAT\shared\libs.
Restart Tomcat.
The web service has inside its web\WEB-INF\classes directory two configuration file.
The first is the axdb.properites that allows you to configure database URL, user and password according to your configuration. The parameters to be configured are:

- axdbUrl (for example jdbc:mysql://localhost/axdb-test?jdbcCompliantTruncation=false)
- axdbUser (for example axdbuser)
- axdbPwd (for example mkzamk)

The second file is named selection.properties and allows the selection Web Service to call an instance of the Query support with a valid user and password, the parameters are:

- QuerySupport, that is the URL of the QS web service (for example http://localhost:8080/MainQuerySupportWS/mqs)
- QSUser, that is the QS user to be adopted (for example test)
- QSPwd, that is the password of the user for QS web service (for example test)

After this preliminary configuration the WAR file can be deployed on Tomcat without any other issue to be addressed.
Once deployed by looking at http://yourserver:yourport/SelectionWS/sa you will obtain something like:
Web Services

Port Name	Status	Information
listenerservice	ACTIVE	Address: http://localhost:8080/SelectionWS/listener WSDL: http://localhost:8080/SelectionWS/listener?WSDL Port QName: {http://www.someone.org/actualizelistener.wsdl}ActualizeListener Remote interface: it.exitech.axmedis.ws.actualizelistener.ActualizeListenerPortType Implementation class: it.exitech.axmedis.ws.actualizelistener.ActualizeListenerPortType_Impl Model: http://localhost:8080/SelectionWS/listener?model
Selection Archived	ACTIVE	Address: http://localhost:8080/SelectionWS/sa WSDL: http://localhost:8080/SelectionWS/sa?WSDL Port QName: {http://www.axmedis.org/selection_archive.wsdl}Selection_Archive Remote interface: it.exitech.axmedis.ws.selectionarchive.Selection_ArchivePortType Implementation class: it.exitech.axmedis.ws.selectionarchive.Selection_ArchivePortType_Impl

		Model:	http://localhost:8080/SelectionWS/sa?model

(please note that return have been added to make them readable)

12.5 Draft User Manual

No user manual is present since it is a back-office module, please refer to the previous section.

12.6 Examples of usage

In order to test the actualization of the selection you can use the following example selection on the online web service using user test and password test:

```
<?xml version="1.0" encoding="UTF-8"?>
<selection                                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Y:\AXMEDIS\Deliverables\Specification\AXMEDIS-DE3-1-2-2\AX-
database-and-query-support-PART9\Selection-v1-7.xsd"            timestamp="2006-03-01T16:00:00"
name="testactualization">
	<AXOID>myaxoid1</AXOID>
	<query>
		<source>
			<location>AXDB</location>
		</source>
		<PARquery>
			<querycondition>
				<nesting>
					<test>
						<field>distributable</field>
						<operator>EQ</operator>
						<value>italy</value>
					</test>
				</nesting>
			</querycondition>
		</PARquery>
		<AXinfoQuery>
			<querycondition>
				<nesting>
					<test>
						<field>DCMI:type</field>
						<operator>STARTWITH</operator>
						<value>commedia</value>
					</test>
				</nesting>
			</querycondition>
		</AXinfoQuery>
	</query>
	<AXOID>myaxoid2</AXOID>
</selection>
```

12.7 Integration and compilation issues

No compilation issues are present apart from the requirement of using JDK 1.5 and JWSDP 1.6

12.8 Configuration Parameters

Config parameter	Possible values
axdbUrl	The URI of the database jdbc:mysql://localhost/axdb-test?jdbcCompliantTruncation=false
axdbUser	The user for database access
axdbPwd	The password for database user
QuerySupport	The URL of the QS web service such as http\://localhost\:8080/MainQuerySupportWS/mqs
QSUser	User to access QS
QSPwd	Password of the user to access QS

13 Query User Interface WEB BASED (EXITECH)

Module/Tool Profile	
Query User Interface Web Based	
Responsible Name	Fioravanti
Responsible Partner	EXITECH
Status (proposed/approved)	approved
Implemented/not implemented	Implemented
Status of the implementation	Completed, under maintenance
Executable or Library/module (Support)	
Single Thread or Multithread	Multithread
Language of Development	JAVA
Platforms supported	All supported by JDK 1.5
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/newrepos/Applications/QueryWebUserInterface/source/
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/newrepos/Applications/QueryWebUserInterface/bin/
Reference to the AXFW location of the demonstrator executable tool for public download	--
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	user test pwd test
Test cases (present/absent)	--
Test cases location	--
Usage of the AXMEDIS configuration manager (yes/no)	no
Usage of the AXMEDIS Error Manager (yes/no)	no
Major Problems not solved	none
Major pending requirements	none

Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)

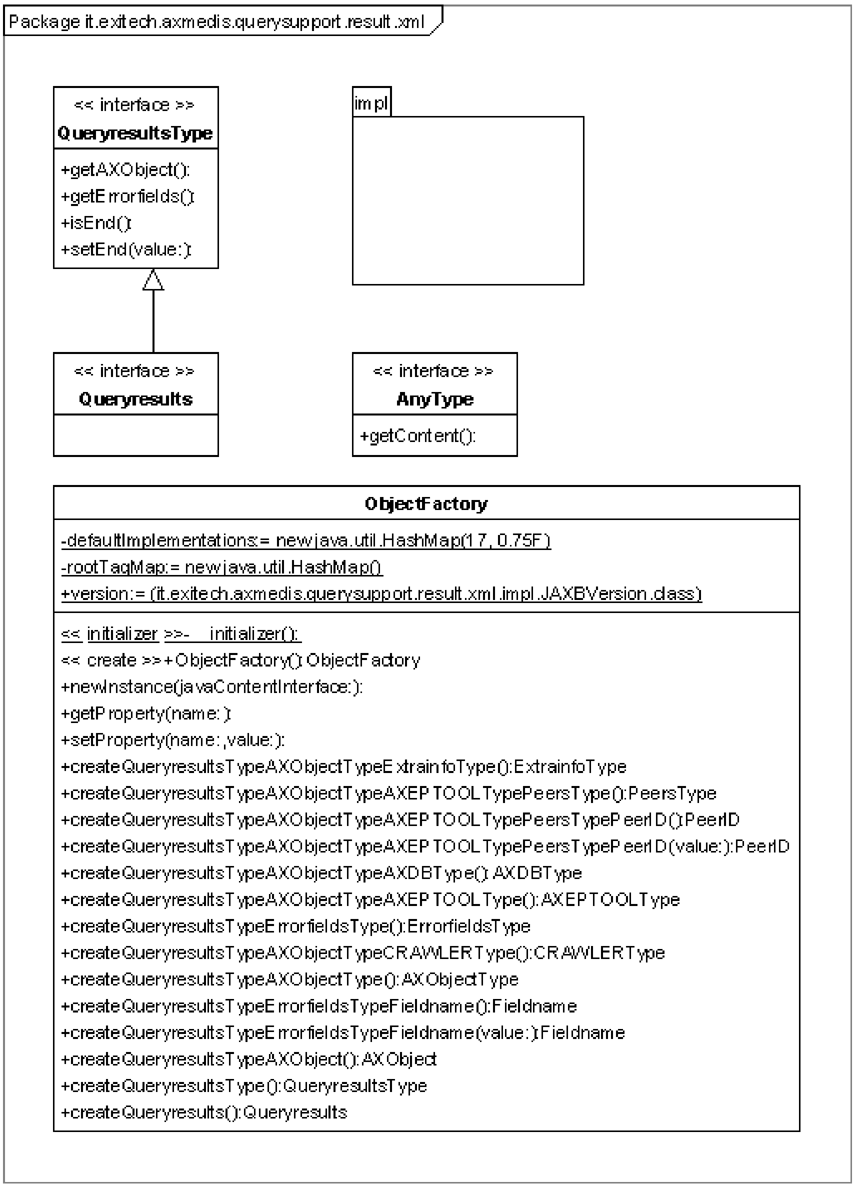
Formats Used	Shared with	format name or reference to a section
Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not

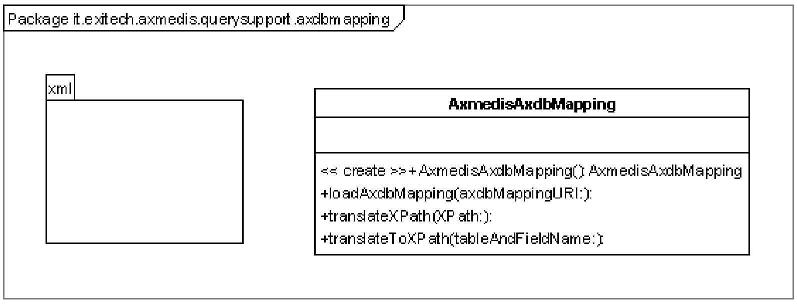
13.1 General Description of the Module (EXITECH)

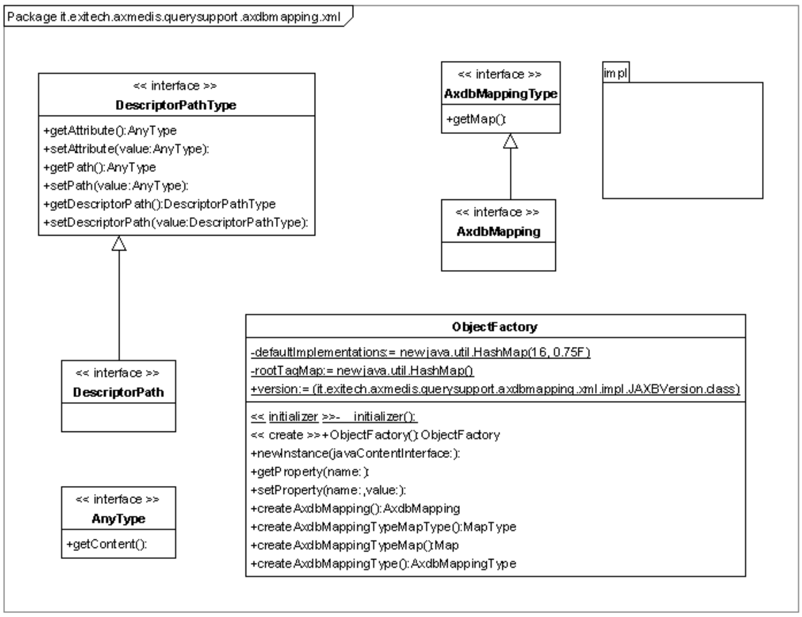
The Web Query user interface is a simplified version of the C++ Query User Interface defined in Section 14 suitable for Kiosks and web portals and mainly targeted to the real end user that wants to retrieve digital items giving some simple information. To this end the expressivity of the query is limited and not all the contraints and logical operators are supported. Also the part related to PAR is really simplified in order to address the most common inquiries. Please note that at the moment several problems exists in the PARQuerySupport and some queries seems impossible to be performed and therefore it is strongly suggested not to use this feature until PARQuerySupport will be stable.
It is important, in any case to have an administrative section that allows the kiosk or portal configuration person to allow a certain level of customization in order to fit the user interface to the targeted audience.
It is important that the interface reads the same configuration files that is used by the target query support for estabilishing the fields to be used and the correct mapping between the issued query and the expected parameters on the query support side.
This module has been used with different configuration parameters also for P2P Query Portal in order to allow code reuse and sharing of modules. In the installation and configuration section the parameters for the configuration will be detailed.

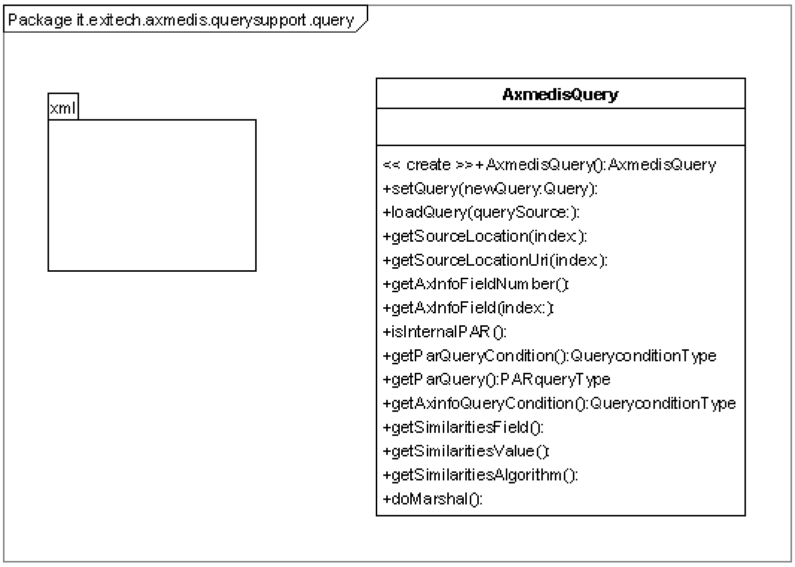
13.2 Module Design in terms of Classes (EXITECH)

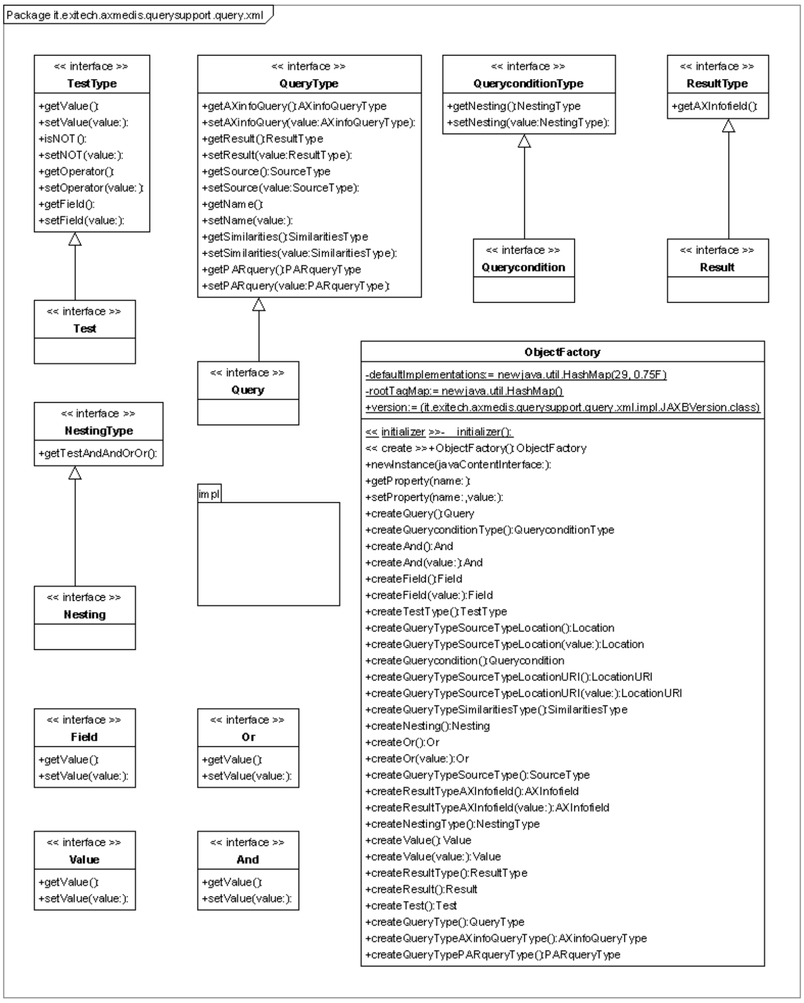
The general part related to the mapping file, to the query and to the results is common with query support, and is reported below, while some other special purposes classes are identified and shown in the last diagram of this section. Web pages and JSP are not reported here.

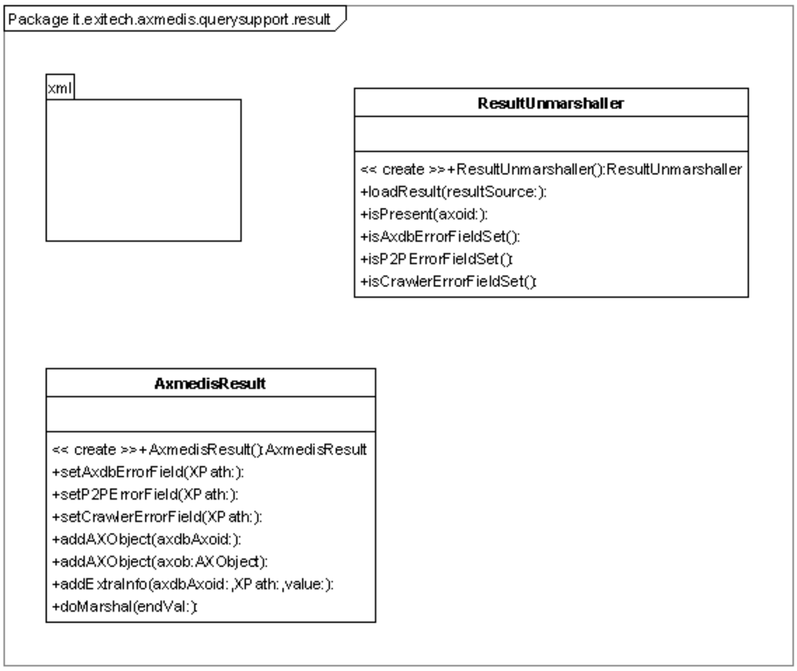

Package it.exitech.axmedis.querysupport.result.xml
<< interface >>
QueryresultsType
+getAXObject():
+getErrorfields():
+isEnd():
+setEnd(value:):
impl
<< interface >>
Queryresults
<< interface >>
AnyType
+getContent():
ObjectFactory
-defaultImplementations:= new java.util.HashMap(17, 0.75F)
-rootTagMap:= new java.util.HashMap()
+version:= (it.exitech.axmedis.querysupport.result.xml.impl.JAXBVersion.class)
<< initializer >>- initializer():
<< create >>+ObjectFactory():ObjectFactory
+newInstance(javaContentInterface:):
+getProperty(name:):
+setProperty(name:,value:):
+createQueryresultsTypeAXObjectTypeExtrainfoType():ExtrainfoType
+createQueryresultsTypeAXObjectTypeAXEPTOOLTypePeersType():PeersType
+createQueryresultsTypeAXObjectTypeAXEPTOOLTypePeersTypePeerID():PeerID
+createQueryresultsTypeAXObjectTypeAXEPTOOLTypePeersTypePeerID(value:):PeerID
+createQueryresultsTypeAXObjectTypeAXDBType():AXDBType
+createQueryresultsTypeAXObjectTypeAXEPTOOLType():AXEPTOOLType
+createQueryresultsTypeErrorfieldsType():ErrorfieldsType
+createQueryresultsTypeAXObjectTypeCRAWLERType():CRAWLERType
+createQueryresultsTypeAXObjectType():AXObjectType
+createQueryresultsTypeErrorfieldsTypeFieldname():Fieldname
+createQueryresultsTypeErrorfieldsTypeFieldname(value:):Fieldname
+createQueryresultsTypeAXObject():AXObject
+createQueryresultsType():QueryresultsType
+createQueryresults():Queryresults

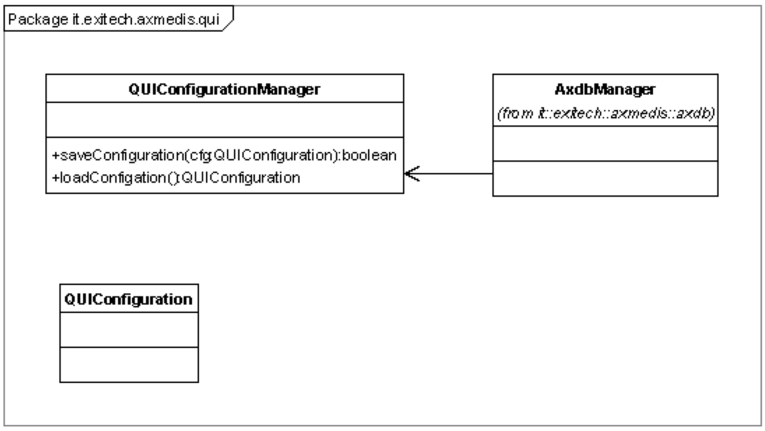

Package it.exitech.axmedis.querysupport.axdbmapping
xml
AxmedisAxdbMapping
<< create >>+AxmedisAxdbMapping():AxmedisAxdbMapping
+loadAxdbMapping(axdbMappingURI:):
+translateXPath(XPath:):
+translateToXPath(tableAndFieldName:):


Package it.exitech.axmedis.querysupport.axdbmapping.xml
<< interface >>
DescriptorPathType
+getAttribute():AnyType
+setAttribute(value:AnyType):
+getPath():AnyType
+setPath(value:AnyType):
+getDescriptorPath():DescriptorPathType
+setDescriptorPath(value:DescriptorPathType):
<< interface >>
AxdbMappingType
+getMap():
impl
<< interface >>
AxdbMapping
<< interface >>
DescriptorPath
ObjectFactory
-defaultImplementations:= new java.util.HashMap(16, 0.75F)
-rootTagMap:= new java.util.HashMap()
+version:= (it.exitech.axmedis.querysupport.axdbmapping.xml.impl.JAXBVersion.class)
<< initializer >>-__initializer():
<< create >>+ObjectFactory():ObjectFactory
+newInstance(javaContentInterface:):
+getProperty(name:):
+setProperty(name:,value:):
+createAxdbMapping():AxdbMapping
+createAxdbMappingTypeMapType():MapType
+createAxdbMappingTypeMap():Map
+createAxdbMappingType():AxdbMappingType
<< interface >>
AnyType
+getContent():


Package it.exitech.axmedis.querysupport.query
xml
AxmedisQuery
<< create >>+AxmedisQuery():AxmedisQuery
+setQuery(newQuery:Query):
+loadQuery(querySource:):
+getSourceLocation(index:):
+getSourceLocationUri(index:):
+getAxInfoFieldNumber():
+getAxInfoField(index:):
+isInternalPAR():
+getParQueryCondition():QueryconditionType
+getParQuery():PARqueryType
+getAxinfoQueryCondition():QueryconditionType
+getSimilaritiesField():
+getSimilaritiesValue():
+getSimilaritiesAlgorithm():
+doMarshal():


Package it.exitech.axmedis.querysupport.query.xml
<< interface >>
TestType
+getValue():
+setValue(value:):
+isNOT():
+setNOT(value:):
+getOperator():
+setOperator(value:):
+getField():
+setField(value:):
<< interface >>
QueryType
+getAXinfoQuery():AXinfoQueryType
+setAXinfoQuery(value:AXinfoQueryType):
+getResult():ResultType
+setResult(value:ResultType):
+getSource():SourceType
+setSource(value:SourceType):
+getName():
+setName(value:):
+getSimilarities():SimilaritiesType
+setSimilarities(value:SimilaritiesType):
+getPARquery():PARqueryType
+setPARquery(value:PARqueryType):
<< interface >>
QueryconditionType
+getNesting():NestingType
+setNesting(value:NestingType):
<< interface >>
ResultType
+getAXInfofield():
<< interface >>
Querycondition
<< interface >>
Result
<< interface >>
Test
<< interface >>
Query
ObjectFactory
-defaultImplementations:= new java.util.HashMap(29, 0.75F)
-rootTagMap:= new java.util.HashMap()
+version:= (it.exitech.axmedis.querysupport.query.xml.impl.JAXBVersion.class)
<< initializer >>- initializer():
<< create >>+ObjectFactory():ObjectFactory
+newInstance(javaContentInterface:):
+getProperty(name:):
+setProperty(name:,value:):
+createQuery():Query
+createQueryconditionType():QueryconditionType
+createAnd():And
+createAnd(value:):And
+createField():Field
+createField(value:):Field
+createTestType():TestType
+createQueryTypeSourceTypeLocation():Location
+createQueryTypeSourceTypeLocation(value:):Location
+createQuerycondition():Querycondition
+createQueryTypeSourceTypeLocationURI():LocationURI
+createQueryTypeSourceTypeLocationURI(value:):LocationURI
+createQueryTypeSimilaritiesType():SimilaritiesType
+createNesting():Nesting
+createOr():Or
+createOr(value:):Or
+createQueryTypeSourceType():SourceType
+createResultTypeAXInfofield():AXInfofield
+createResultTypeAXInfofield(value:):AXInfofield
+createNestingType():NestingType
+createValue():Value
+createValue(value:):Value
+createResultType():ResultType
+createResult():Result
+createTest():Test
+createQueryType():QueryType
+createQueryTypeAXinfoQueryType():AXinfoQueryType
+createQueryTypePARqueryType():PARqueryType
<< interface >>
NestingType
+getTestAndAndOrOr():
impl
<< interface >>
Nesting
<< interface >>
Field
+getValue():
+setValue(value:):
<< interface >>
Or
+getValue():
+setValue(value:):
<< interface >>
Value
+getValue():
+setValue(value:):
<< interface >>
And
+getValue():
+setValue(value:):


Package it.exitech.axmedis.querysupport.result
xml
ResultUnmarshaller
<< create >>+ResultUnmarshaller():ResultUnmarshaller
+loadResult(resultSource:):
+isPresent(axoid:):
+isAxdbErrorFieldSet():
+isP2PErrorFieldSet():
+isCrawlerErrorFieldSet():
AxmedisResult
<< create >>+AxmedisResult():AxmedisResult
+setAxdbErrorField(XPath:):
+setP2PErrorField(XPath:):
+setCrawlerErrorField(XPath:):
+addAXObject(axdbAxoid:):
+addAXObject(axob:AXObject):
+addExtraInfo(axdbAxoid:,XPath:,value:):
+doMarshal(endVal:):

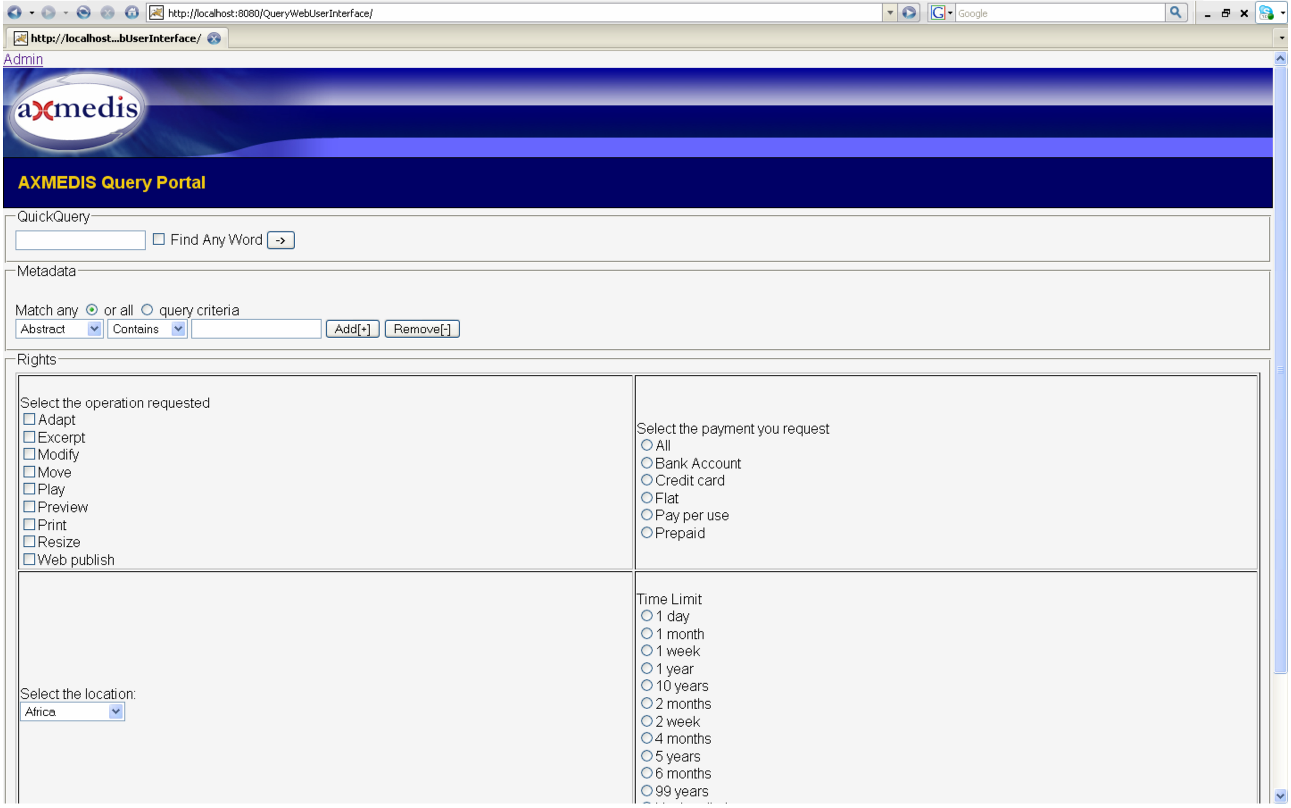

Package it.exitech.axmedis.qui
QUIConfigurationManager
+saveConfiguration(cfg:QUIConfiguration):boolean
+loadConfigation():QUIConfiguration
AxdbManager
(from it::exitech::axmedis::axdb)
QUIConfiguration

13.3 User interface description (EXITECH)

User interface for web queries has to be customizable and therefore has two faces, the user and the administrative one. When something is changed in the administrative side, the interface for users will be modified accordingly.

It is also necessary to remind that the Query User Interface can be used as Query interface for P2P.

In the following screenshot we have one of the possible user interfaces that can be generated by the admin interface proposed below:



http://localhost:8080/QueryWebUserInterface/
http://localhost...bUserInterface/
Admin
axmedis

AXMEDIS Query Portal

QuickQuery

□ Find Any Word ->

Metadata

Match any ● or all ○ query criteria

Abstract | Contains | Add[+] | Remove[-]

Rights

Select the operation requested
□ Adapt
□ Excerpt
□ Modify
□ Move
□ Play
□ Preview
□ Print
□ Resize
□ Web publish

Select the payment you request
○ All
○ Bank Account
○ Credit card
○ Flat
○ Pay per use
○ Prepaid

Select the location:
Africa

Time Limit
○ 1 day
○ 1 month
○ 1 week
○ 1 year
○ 10 years
○ 2 months
○ 2 week
○ 4 months
○ 5 years
○ 6 months
○ 99 years

This user interface is mainly divided in two sections. The upper part that is related to Metadata and that offer the possibility to match any or all the criteria, by dynamically adding or removing constraints, and by performing a quinck query based on the most commonly used metadata such as Title, Creator, Descrption, etc

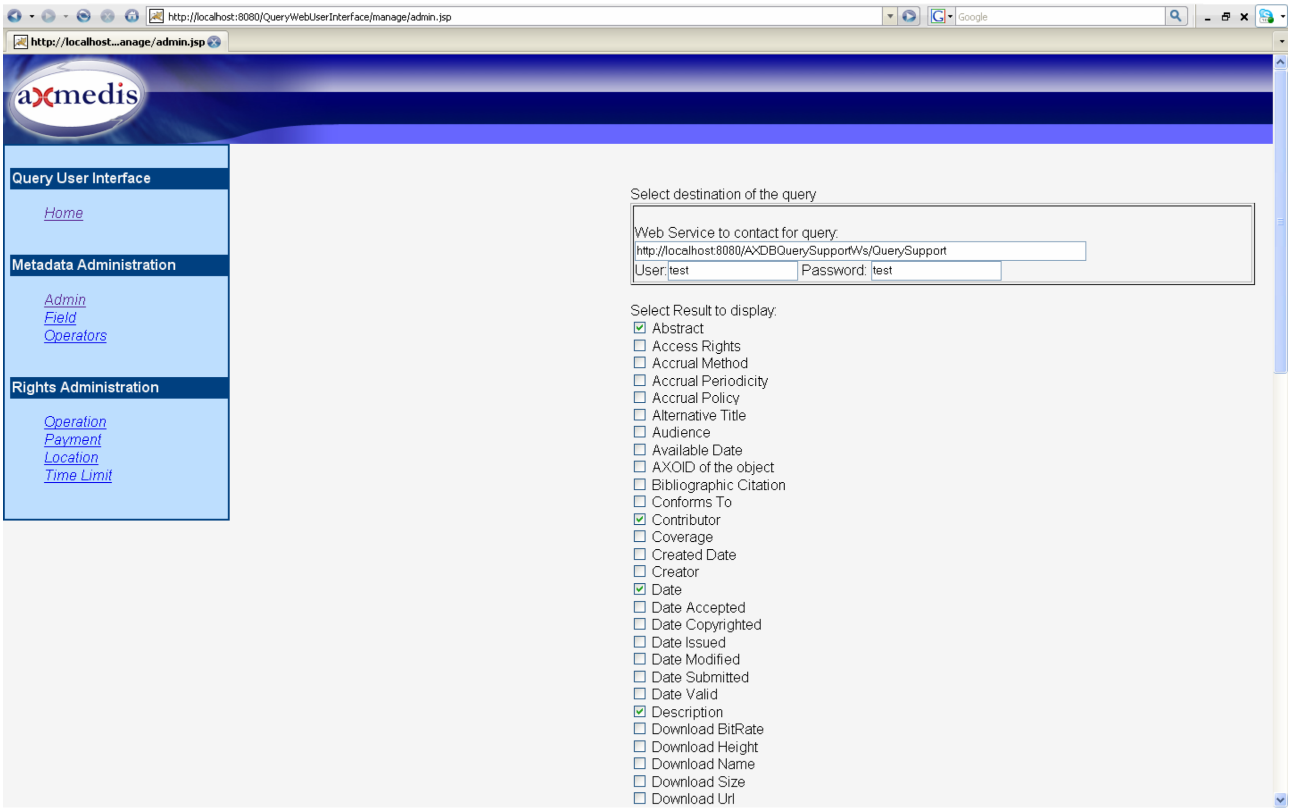
From the administrative side it will be possible to customize the fields to be queried, the operators allowed and the maximum number of constraints. The lower part is related to the rights we want to exploit on the objects and it is mainly divided in four sections that can appear on not on the basis of the administrative input.

These sections are:

- Operations, that are the operations that the user would like to perform on the object. This set of operation is completely customizable from the administrative interface
- Locations, that is very the object can be used. The countries to be inserted here are customizable from the administrative interface
- Payment, that is the part related to fees and whose content can be customized from the administarive interface
- Time limit, that is the limitation in time for which we would like to exploit rights on the object. Alsoi this part is fully customizable in the administrative side

It is important now to evidence the administarive interface that allows the customization of the user interface and that will be accessible only after an administrative login on the system.

The main screen is reported below:



http://localhost:8080/QueryWebUserInterface/manage/admin.jsp

Query User Interface

Home

Metadata Administration

Admin
Field
Operators

Rights Administration

Operation
Payment
Location
Time Limit

Select destination of the query

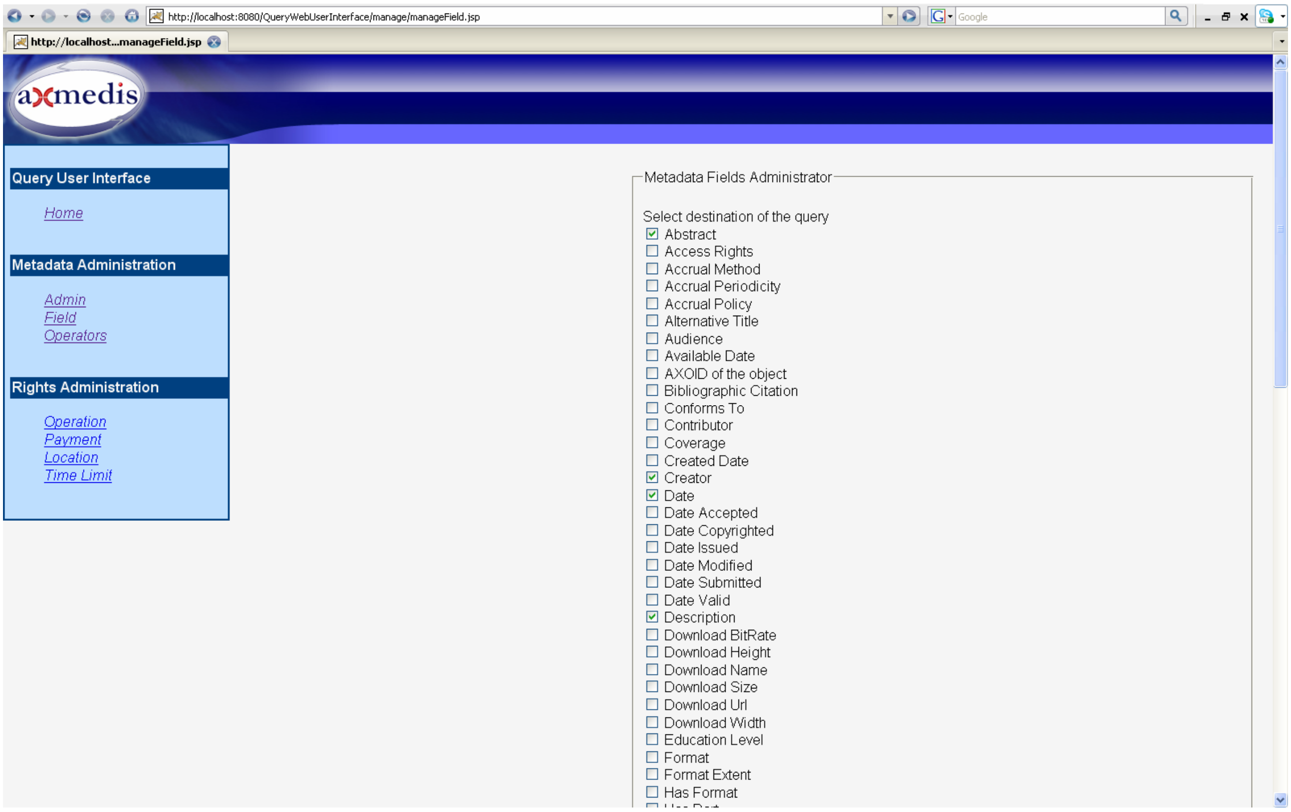
Web Service to contact for query:
http://localhost:8080/AXDBQuerySupportWs/QuerySupport
User: test Password: test

Select Result to display:

☑ Abstract
☐ Access Rights
☐ Accrual Method
☐ Accrual Periodicity
☐ Accrual Policy
☐ Alternative Title
☐ Audience
☐ Available Date
☐ AXOID of the object
☐ Bibliographic Citation
☐ Conforms To
☑ Contributor
☐ Coverage
☐ Created Date
☐ Creator
☑ Date
☐ Date Accepted
☐ Date Copyrighted
☐ Date Issued
☐ Date Modified
☐ Date Submitted
☐ Date Valid
☑ Description
☐ Download BitRate
☐ Download Height
☐ Download Name
☐ Download Size
☐ Download Url

In this main page the fields returned as a the object rdescription and the sources can be imposed.

The fields are managed by the following user interface:



http://localhost:8080/QueryWebUserInterface/manage/manageField.jsp

Query User Interface

Home

Metadata Administration

Admin
Field
Operators

Rights Administration

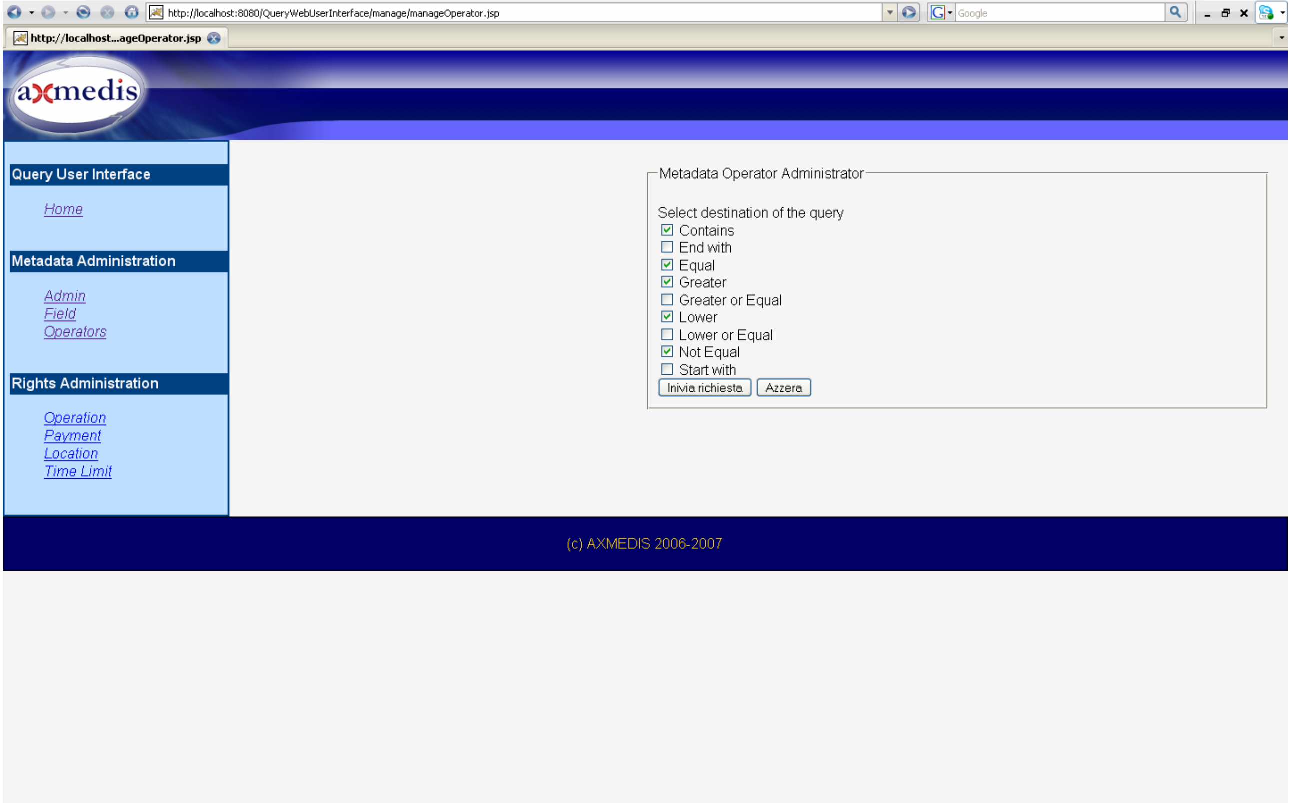
Operation
Payment
Location
Time Limit

Metadata Fields Administrator

Select destination of the query

☑ Abstract
☐ Access Rights
☐ Accrual Method
☐ Accrual Periodicity
☐ Accrual Policy
☐ Alternative Title
☐ Audience
☐ Available Date
☐ AXOID of the object
☐ Bibliographic Citation
☐ Conforms To
☐ Contributor
☐ Coverage
☐ Created Date
☑ Creator
☑ Date
☐ Date Accepted
☐ Date Copyrighted
☐ Date Issued
☐ Date Modified
☐ Date Submitted
☐ Date Valid
☑ Description
☐ Download BitRate
☐ Download Height
☐ Download Name
☐ Download Size
☐ Download Url
☐ Download Width
☐ Education Level
☐ Format
☐ Format Extent
☐ Has Format

The list of fields reported in this page will contain ALL the fields that are inside the mapping file.
In the following picture the configuration of the operators that can be used is reported:

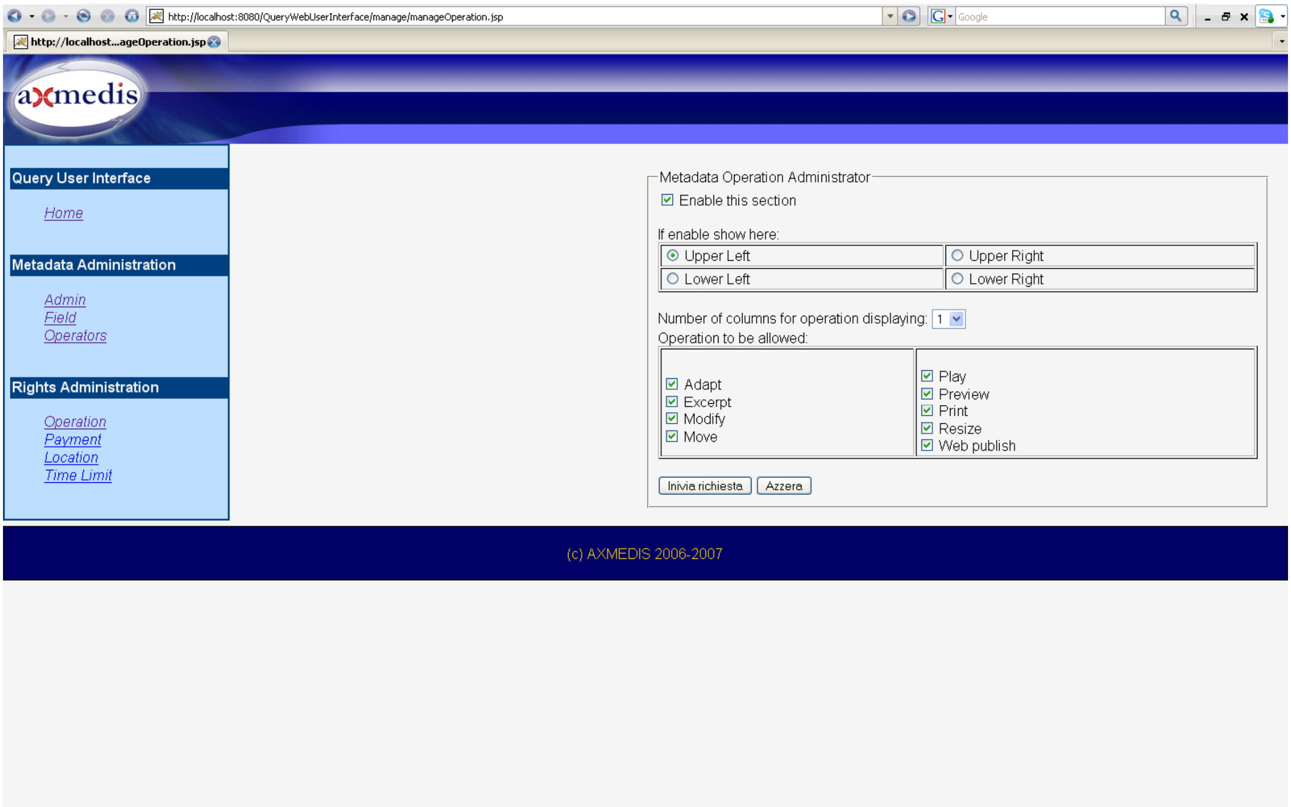


Metadata Operator Administrator

Select destination of the query

☑ Contains
□ End with
☑ Equal
☑ Greater
□ Greater or Equal
☑ Lower
□ Lower or Equal
☑ Not Equal
□ Start with

The management of rights is more complex and is divided in four sections. The first section is that of the operations to be allowed:



Metadata Operation Administrator

☑ Enable this section

If enable show here:

● Upper Left	○ Upper Right
○ Lower Left	○ Lower Right

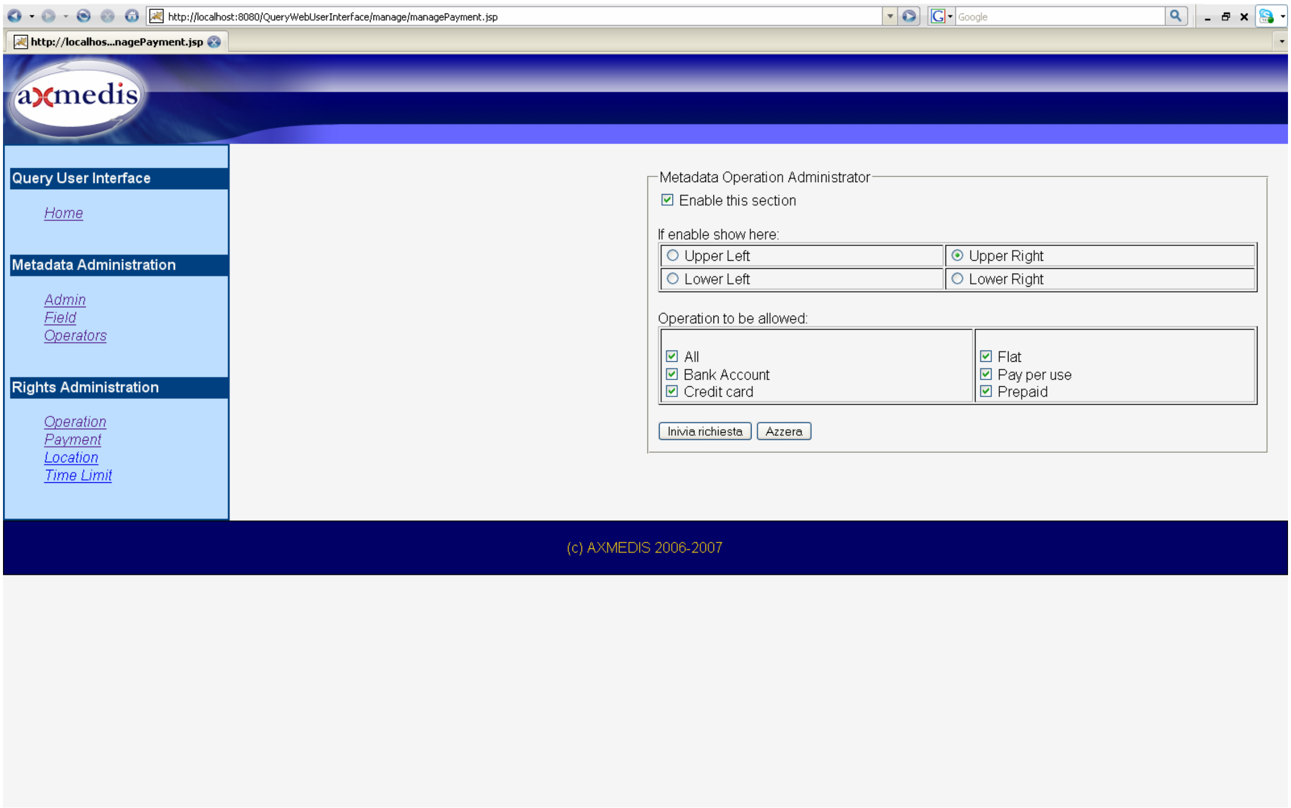
Number of columns for operation displaying: 1

Operation to be allowed:

☑ Adapt ☑ Excerpt ☑ Modify ☑ Move	☑ Play ☑ Preview ☑ Print ☑ Resize ☑ Web publish

The administration allows to enable on not the section, estabilish where the section has to be shown in the user interface and select the operations for which the user can ask. Since the number of operations can be also a large one, it is requested in how many columns the data will be displayed.

The other sections are very similar in the approach and for that reason only the user interface is displayed.



Metadata Operation Administrator

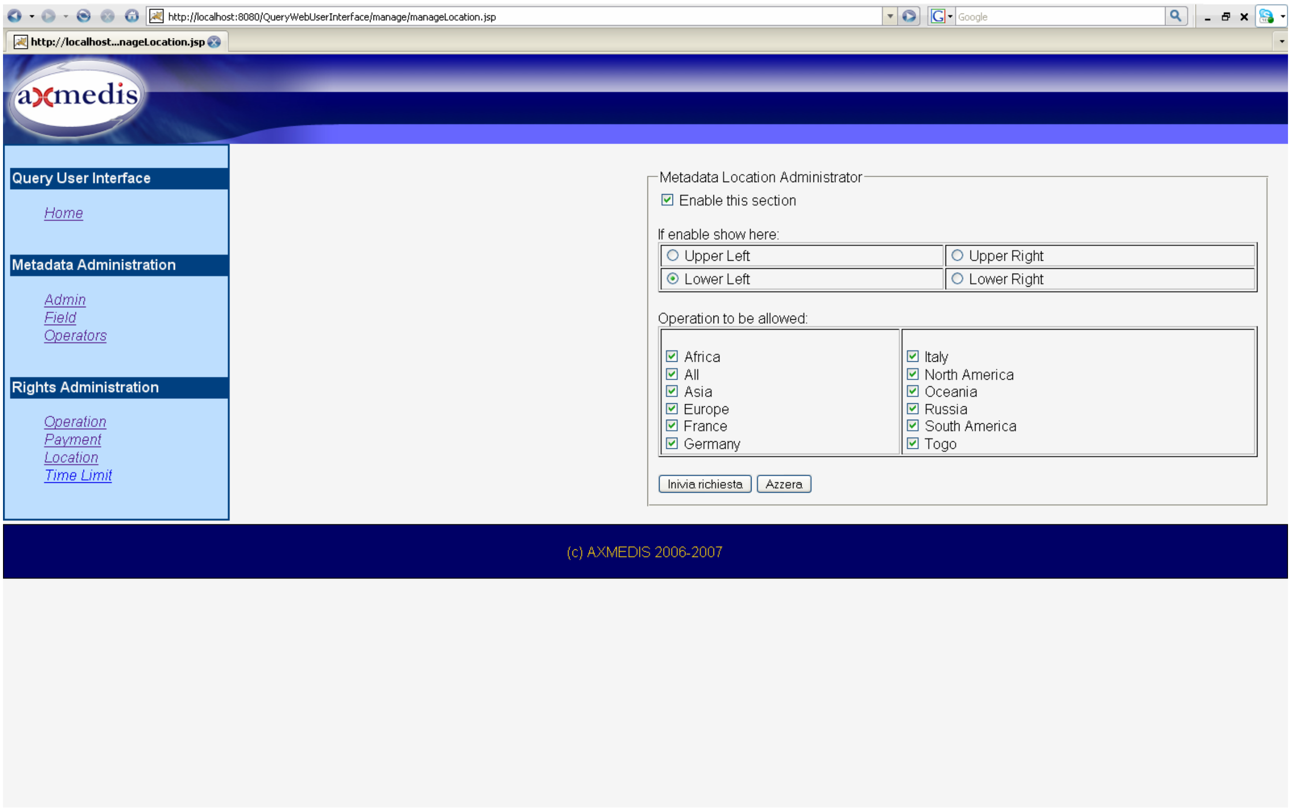
☑ Enable this section

If enable show here:

○ Upper Left	● Upper Right
○ Lower Left	○ Lower Right

Operation to be allowed:

☑ All ☑ Bank Account ☑ Credit card	☑ Flat ☑ Pay per use ☑ Prepaid



Metadata Location Administrator

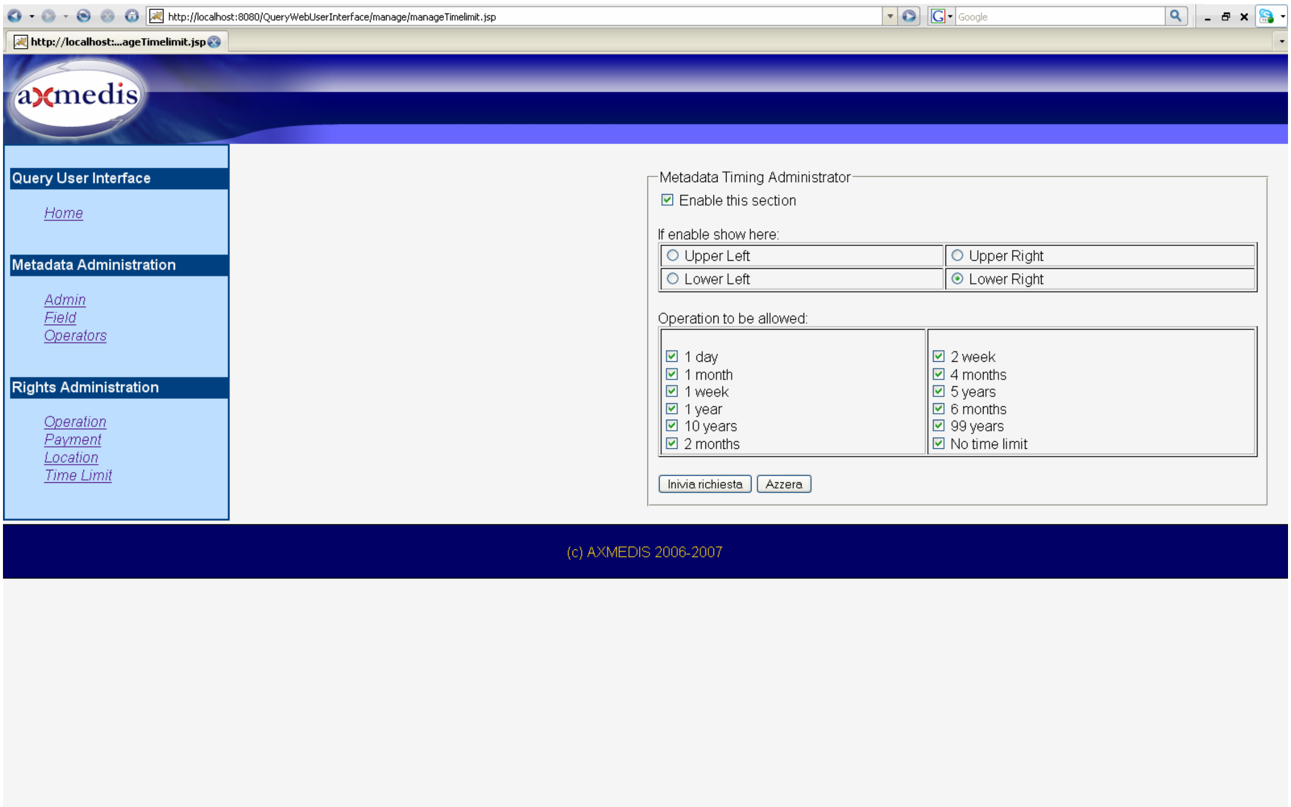
☑ Enable this section

If enable show here:

○ Upper Left	○ Upper Right
● Lower Left	○ Lower Right

Operation to be allowed:

☑ Africa ☑ All ☑ Asia ☑ Europe ☑ France ☑ Germany	☑ Italy ☑ North America ☑ Oceania ☑ Russia ☑ South America ☑ Togo



Metadata Timing Administrator

☑ Enable this section

If enable show here:

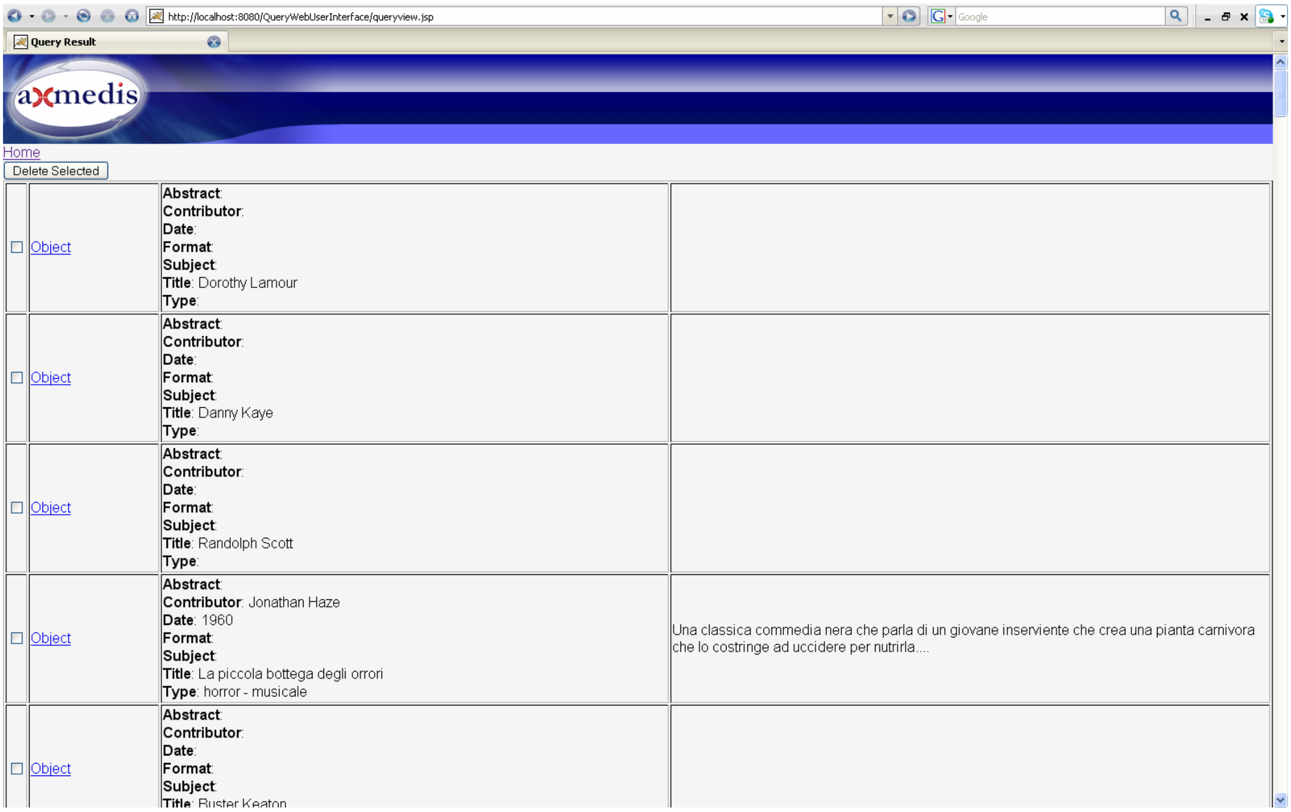
○ Upper Left	○ Upper Right
○ Lower Left	● Lower Right

Operation to be allowed:

☑ 1 day	☑ 2 week
☑ 1 month	☑ 4 months
☑ 1 week	☑ 5 years
☑ 1 year	☑ 6 months
☑ 10 years	☑ 99 years
☑ 2 months	☑ No time limit

Location are not limited to that reported here. This is only a simplification of the real situation to show the interface.

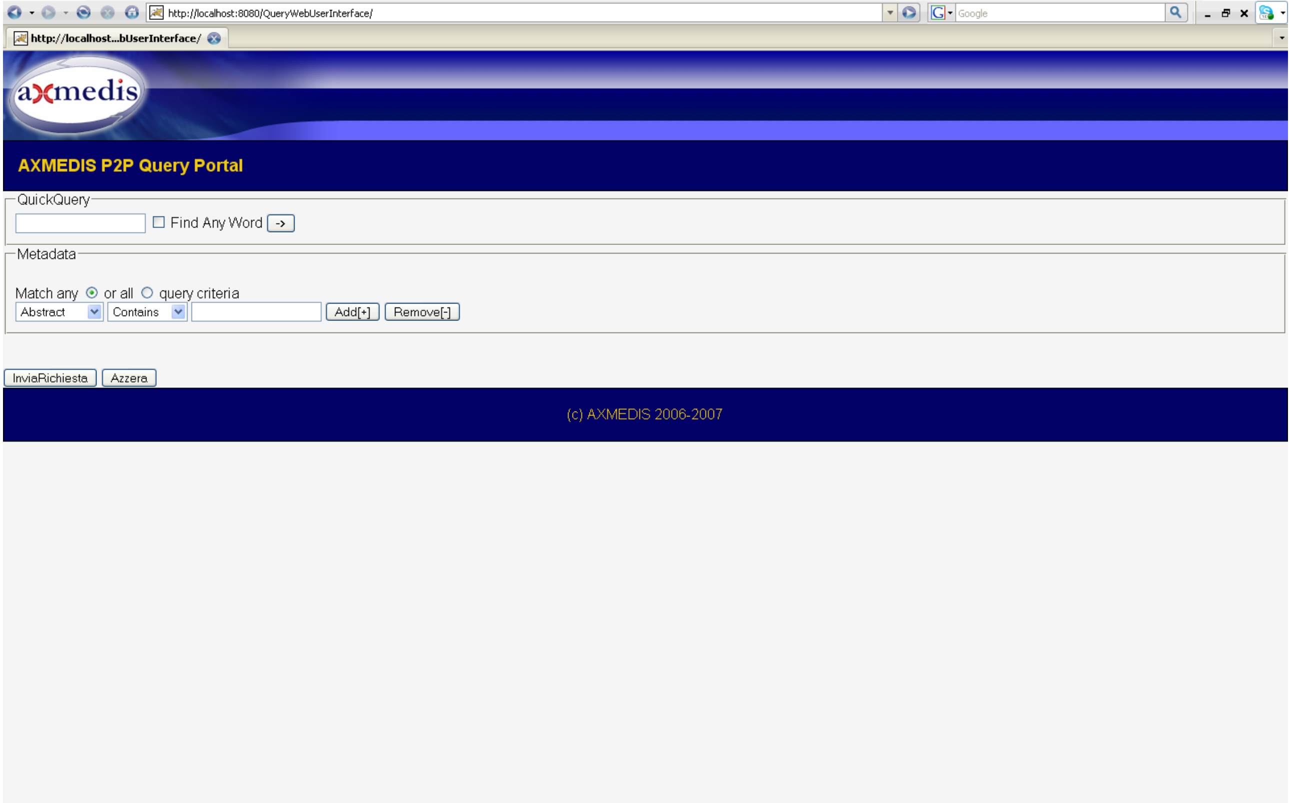
The result page shows all the results in a tabular form giving back to the user the possibility to download the file.



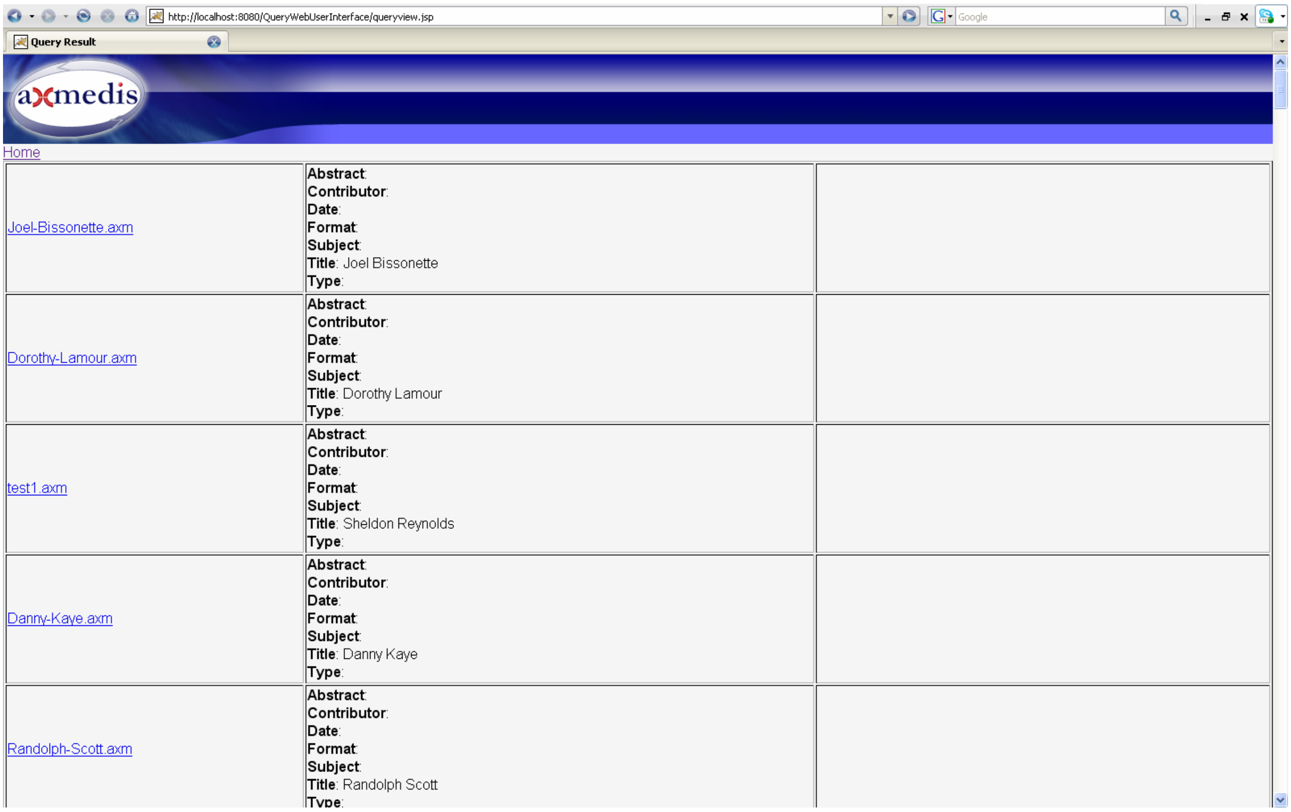
□	Object	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Dorothy Lamour **Type**:	
□	Object	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Danny Kaye **Type**:	
□	Object	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Randolph Scott **Type**:	
□	Object	**Abstract**: **Contributor**: Jonathan Haze **Date**: 1960 **Format**: **Subject**: **Title**: La piccola bottega degli orrori **Type**: horror - musicale	Una classica commedia nera che parla di un giovane inserviente che crea una pianta carnivora che lo costringe ad uccidere per nutrirla....
□	Object	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Buster Keaton	

13.3.1 User interface description: P2P specialization (EXITECH)

The main user interface for P2P usually contains only the metadata section.



The admin part is the same, while results are shown in a different manner for P2P requirements.



Home

Joel-Bissonette.axm	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Joel Bissonette **Type**:	
Dorothy-Lamour.axm	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Dorothy Lamour **Type**:	
test1.axm	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Sheldon Reynolds **Type**:	
Danny-Kaye.axm	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Danny Kaye **Type**:	
Randolph-Scott.axm	**Abstract**: **Contributor**: **Date**: **Format**: **Subject**: **Title**: Randolph Scott **Type**:	

13.4 Technical and Installation information (EXITECH)

Since the QUI is a web application, it is provided as a WAR to be deployed directly on Tomcat 5.5.12 or above.

13.5 Draft User Manual (EXITECH)

See section 13.3 where User Interface is described.

14 Selection User Interface and Query User Interface C++ WEB BASED (DSI)

Module/Tool Profile		
Selection User Interface		
Responsible Name	Ivan Bruno	
Responsible Partner	DSI	
Status (proposed/approved)	Approved	
Implemented/not implemented	Implemented	
Status of the implementation	70%	
Executable or Library/module (Support)	Library	
Single Thread or Multithread	Multithread	
Language of Development	C++	
Platforms supported	Windows, linux and Mac OSx	
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/Framework/source/selectioneditor/ https://cvs.axmedis.org/repos/Framework/include/selectioneditor/	
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/repos/Applications/ruleeditor/bin/win32/	
Reference to the AXFW location of the demonstrator executable tool for public download		
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	https://cvs.axmedis.org/repos/WebServices/QuerySupportWS/ https://cvs.axmedis.org/repos/WebServices/UserSelectionArchive/	
Test cases (present/absent)	Absent	
Test cases location		
Usage of the AXMEDIS configuration manager (yes/no)	Yes	
Usage of the AXMEDIS Error Manager (yes/no)	No	
Major Problems not solved	Customization of UI, minor editing problems	
Major pending requirements	Customization of UI	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Formats Used	Shared with	format name or reference to a section
Protocol Used	Shared with	Protocol name or reference to a section

Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not
XERCES		
WSDLPULL		
WxWidgtes		

14.1 General Description of the Module

This section describes the User Inteface of the Selection Editor. The selection Editor is the tool that provides full functionalities for managing the Selection and Query Result XML document according to XML schemas reported in this deliverable. The editor is a module written in C++ language and can be used in diffent applications. The editor provides two working modes: Selection Mode (full functionalities) and Single Qquery Mode(as Query User Interface application). In both modes the Selection Editor is supported by:

1. an internal client that allows communicating via WebServices with the Main Query Support module.
2. a manager of XML Selection documents
3. an XML validator
4. a customizable user interface

Selection Mode has been introduced to integrate the editor in the AXMEDIS AXCP Rule Editor for generating and editing Selections to be used as parameters for AXCP Rule as specified in the DE.3-1-2-2-6 or for resolving queries and then retrieving AXOIDs of AXMEDIS Objects returned as result.

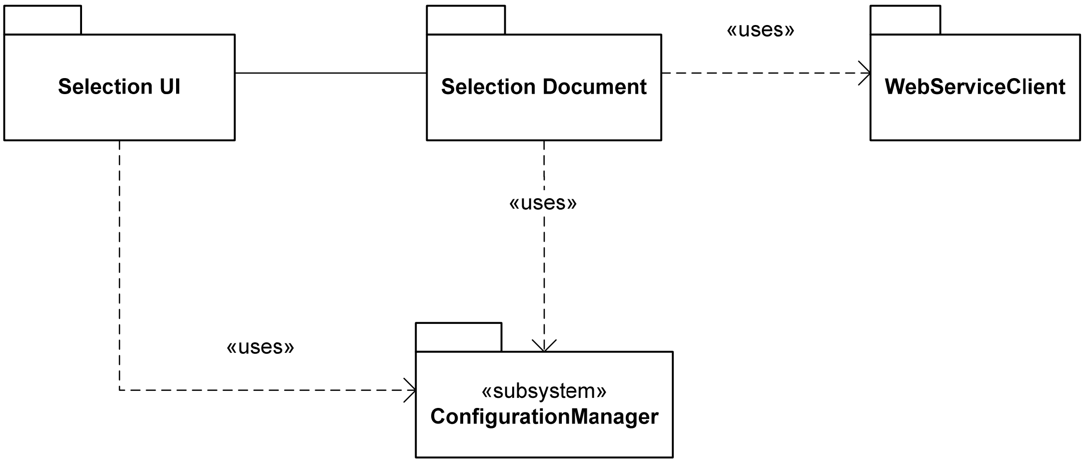
The Single Query Mode can be used to provide a query user interface to C++ applications that need to be interfaced with the Main Query Support, for instance the PnP Editor. In this modality the editor provides a restricted set of functionalities to cope with single query at time and allows retrieving AXOIDs of AXMEDIS Objects returned as result.

14.2 Module Design in terms of Classes

The following diagram shows the main components and relationships of the Selection Editor:

- *Selection UI* is the package for the GUI of the Editor
- *SelectionDocument* is the XML Selection document manager
- *WebServiceClient* is the client for accessing to WebServices of Main Query Support and User Selection Archive

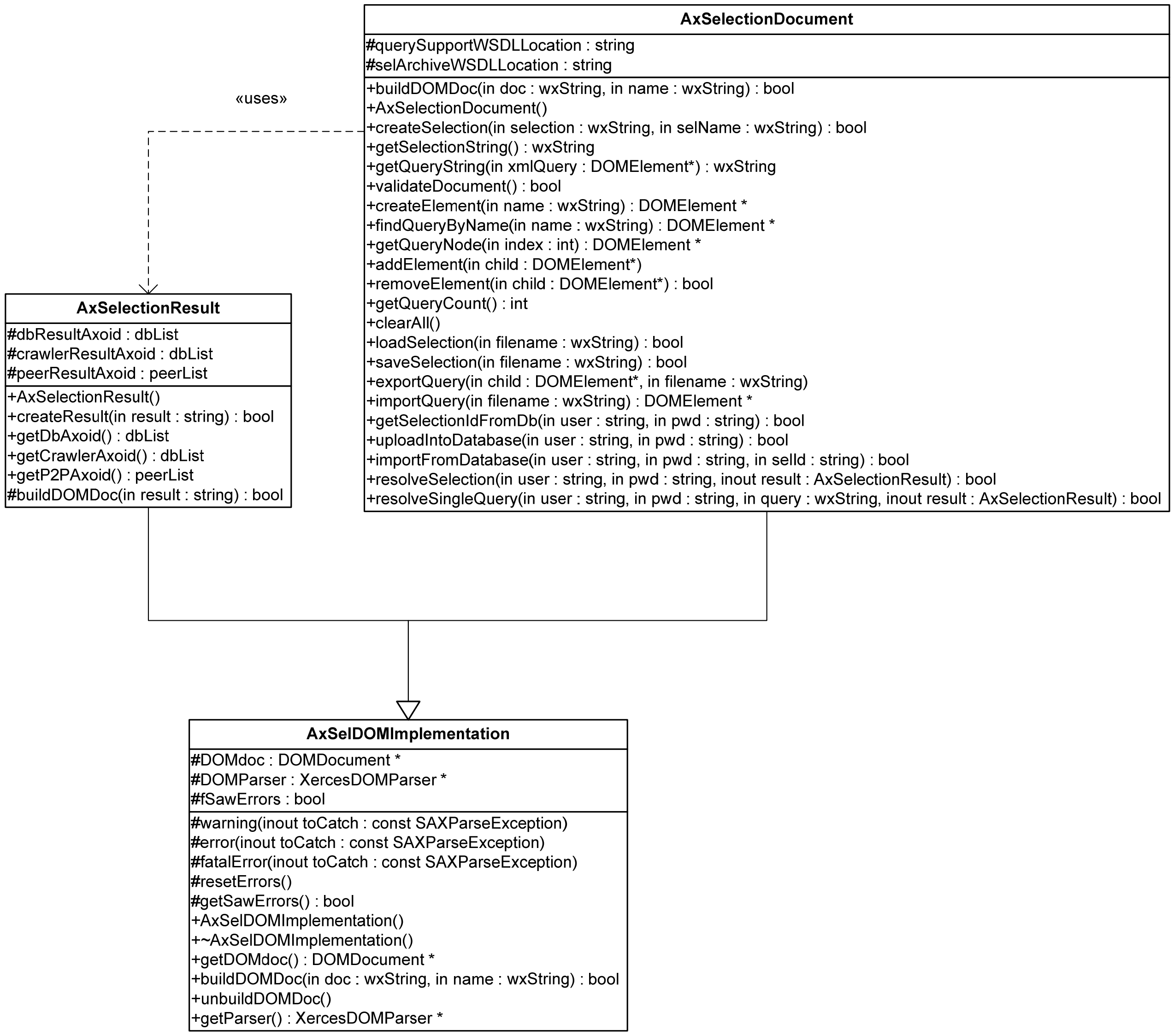
- *ConfigurationManager* is the module of the AXMEDIS Framework that allows managing the configuration of tools.



14.2.1 Selection Document

The following diagram shows the main classes and relationships of the Selection Document package:

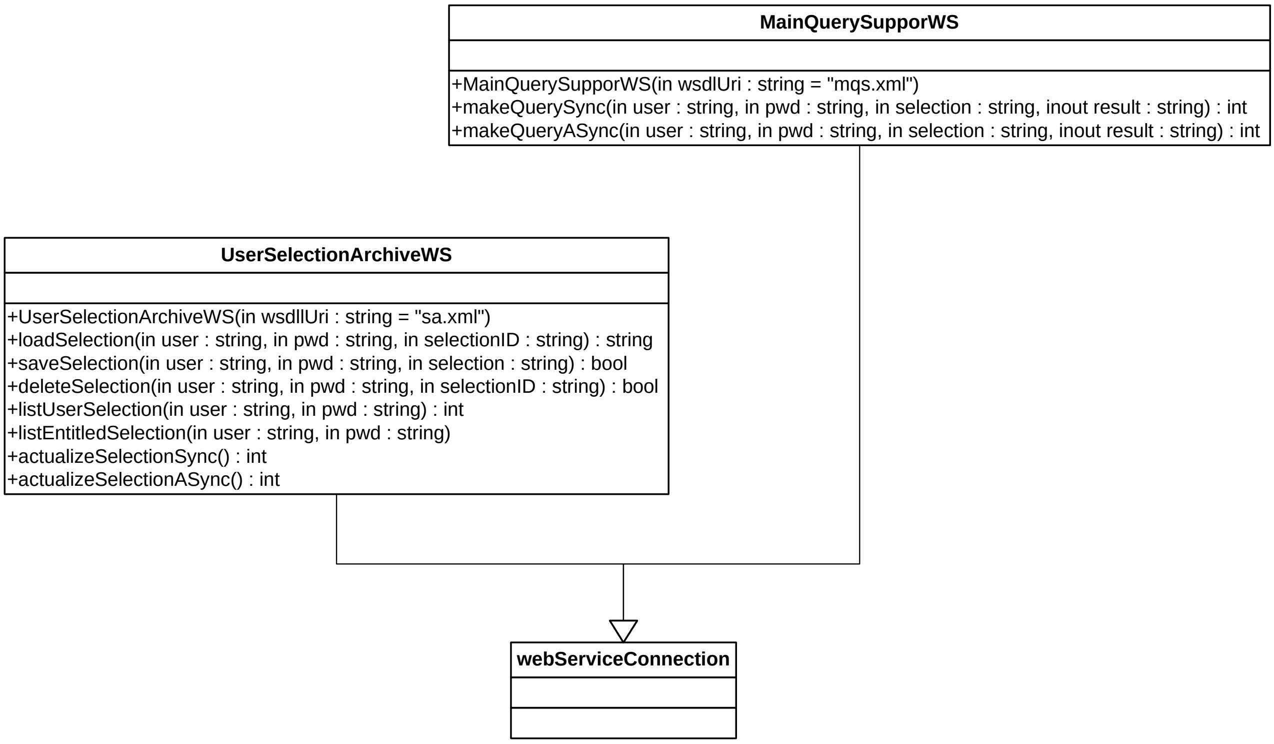
- *AxSelDOMImplementation* class that manages the XML DOM representation of the Selection Document
- *SelectionDocument*: it is the document manager and the interfance to the XML Selection Document. It provides primitives for editing, creating, storing, loading a selection document from disk or from the database of Selections, methods to resolve all queries inside the Selection or a specific query and a method for validate the whole XML according to the selection schema. Finally, it provides methods for importing an existing query on disk in the Selection document or exporting on disk a query of the selection.
- *AxSelectionResult* is the class that models an XML result document coming from the MainQuerySupport. It provides methods to access to results separating them by sources (AXDB, craweler, AxeptTool)


AxSelectionDocument
#querySupportWSDLLocation : string
#selArchiveWSDLLocation : string
+buildDOMDoc(in doc : wxString, in name : wxString) : bool
+AxSelectionDocument()
+createSelection(in selection : wxString, in selName : wxString) : bool
+getSelectionString() : wxString
+getQueryString(in xmlQuery : DOMElement*) : wxString
+validateDocument() : bool
+createElement(in name : wxString) : DOMElement *
+findQueryByName(in name : wxString) : DOMElement *
+getQueryNode(in index : int) : DOMElement *
+addElement(in child : DOMElement*)
+removeElement(in child : DOMElement*) : bool
+getQueryCount() : int
+clearAll()
+loadSelection(in filename : wxString) : bool
+saveSelection(in filename : wxString) : bool
+exportQuery(in child : DOMElement*, in filename : wxString)
+importQuery(in filename : wxString) : DOMElement *
+getSelectionIdFromDb(in user : string, in pwd : string) : bool
+uploadIntoDatabase(in user : string, in pwd : string) : bool
+importFromDatabase(in user : string, in pwd : string, in selId : string) : bool
+resolveSelection(in user : string, in pwd : string, inout result : AxSelectionResult) : bool
+resolveSingleQuery(in user : string, in pwd : string, in query : wxString, inout result : AxSelectionResult) : bool
«uses»
AxSelectionResult
#dbResultAxoid : dbList
#crawlerResultAxoid : dbList
#peerResultAxoid : peerList
+AxSelectionResult()
+createResult(in result : string) : bool
+getDbAxoid() : dbList
+getCrawlerAxoid() : dbList
+getP2PAxoid() : peerList
#buildDOMDoc(in result : string) : bool
AxSelDOMImplementation
#DOMdoc : DOMDocument *
#DOMParser : XercesDOMParser *
#fSawErrors : bool
#warning(inout toCatch : const SAXParseException)
#error(inout toCatch : const SAXParseException)
#fatalError(inout toCatch : const SAXParseException)
#resetErrors()
#getSawErrors() : bool
+AxSelDOMImplementation()
+~AxSelDOMImplementation()
+getDOMdoc() : DOMDocument *
+buildDOMDoc(in doc : wxString, in name : wxString) : bool
+unbuildDOMDoc()
+getParser() : XercesDOMParser *

14.2.2 WebServiceClient

The WebService package provides two classes that implement WebService clients:

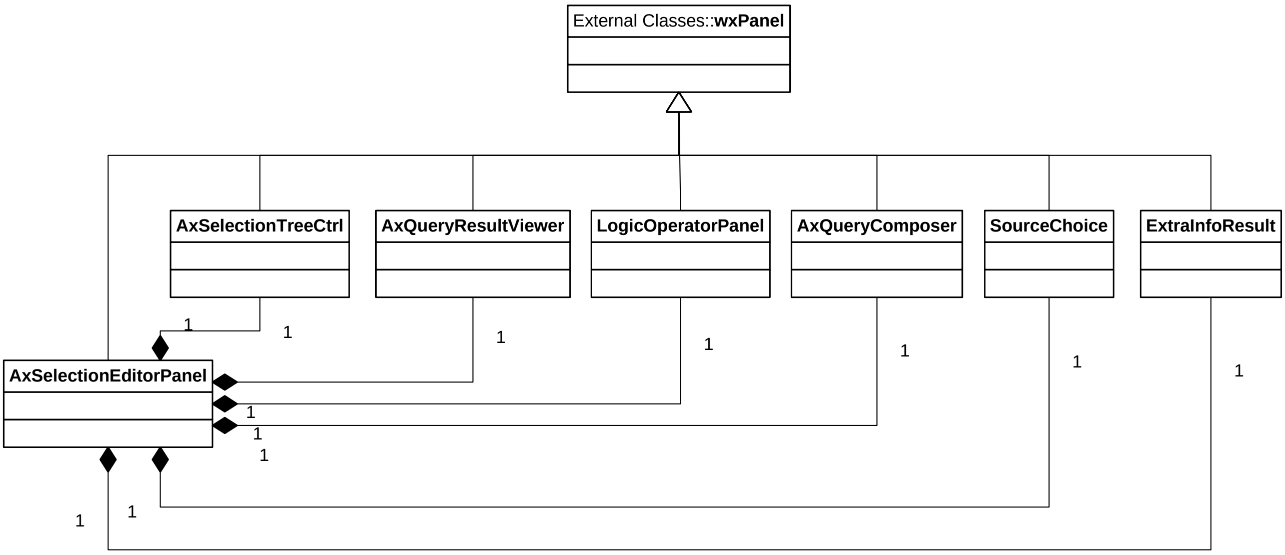
- *MainQuerySupportWS* is the class that models a client for accessing to the Main Query Support. The client is used to actualize the selection document.
- *UserSelectionArchiveWS* is the class that models a client for accessing to the User Selection Archive. The client is used as interface to the User Selection Archive for loading, storing, retrieving existing selection documents.


MainQuerySupporWS
+MainQuerySupporWS(in wsdlUri : string = "mqs.xml")
+makeQuerySync(in user : string, in pwd : string, in selection : string, inout result : string) : int
+makeQueryASync(in user : string, in pwd : string, in selection : string, inout result : string) : int
UserSelectionArchiveWS
+UserSelectionArchiveWS(in wsdllUri : string = "sa.xml")
+loadSelection(in user : string, in pwd : string, in selectionID : string) : string
+saveSelection(in user : string, in pwd : string, in selection : string) : bool
+deleteSelection(in user : string, in pwd : string, in selectionID : string) : bool
+listUserSelection(in user : string, in pwd : string) : int
+listEntitledSelection(in user : string, in pwd : string)
+actualizeSelectionSync() : int
+actualizeSelectionASync() : int
webServiceConnection

Both classes specialize the *webServiceConnection* class that allows to build at runtime a WebService client starting from the WSDL schema of the WebService. Each class works with the WSDL schema specified in this document. For more details about the *webServiceConnection* class please see DE3-1-2-2-6-Spec-of-AX-Content-Processing-upC.

14.2.3 Selection UI

The following diagram shows the main classes and relationships of the Selection UI package:



- *AxSelectionEditorPanel* – It is the main panel of the selection editor and the UI manager.
- *AxSelectionTreeCtrl* – It is the panel that shows the Selection Document as a tree.
- *AxQueryComposer* – It is the panel designed to be used as a query editor
- *AxQueryResultViewer* – It is the panel that shows results after the query/selection submission. It is list view n report mode

- *LogicOperatorPanel* – It is the panel that allows selecting the logic operation to be used for connecting query conditions
- *SourceChoice* – It is the panel that allows to choose the source where submit the query or the selection
- *ExtraInfoResult* – This panel provides the list of fields that can be added to the selection in order to obtain information as feedback from the Main Query Support when submitting queries

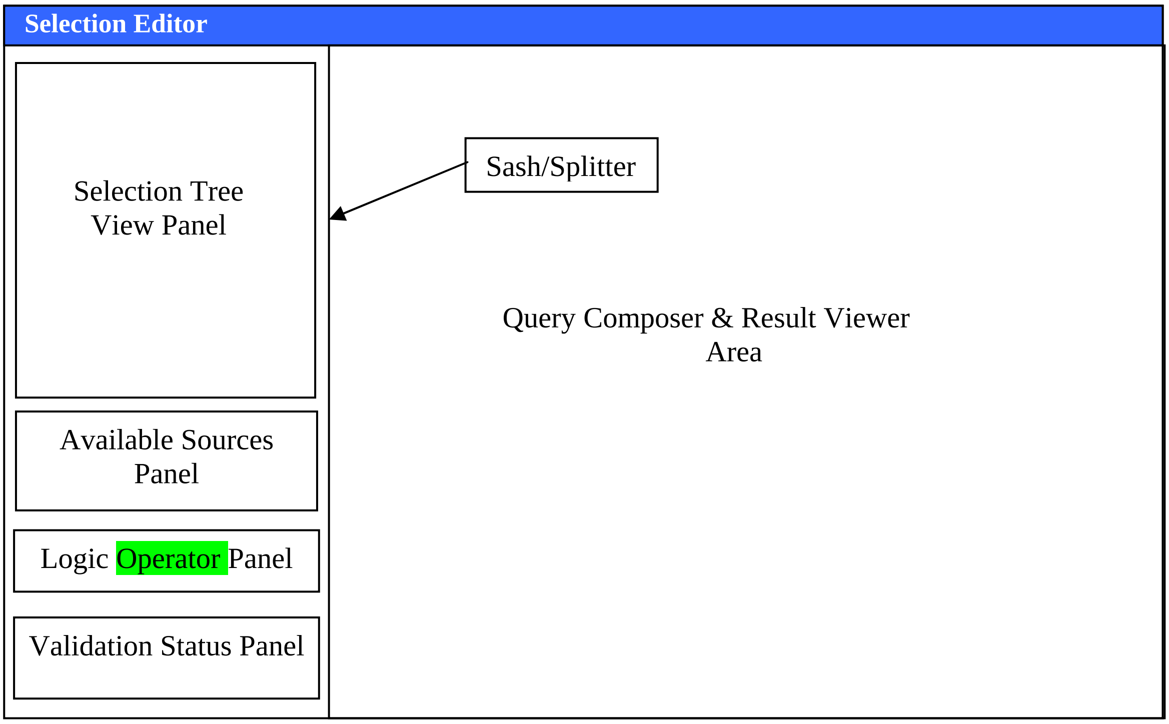
All panels specialize the wxPanel class of the wxWidget graphic library.

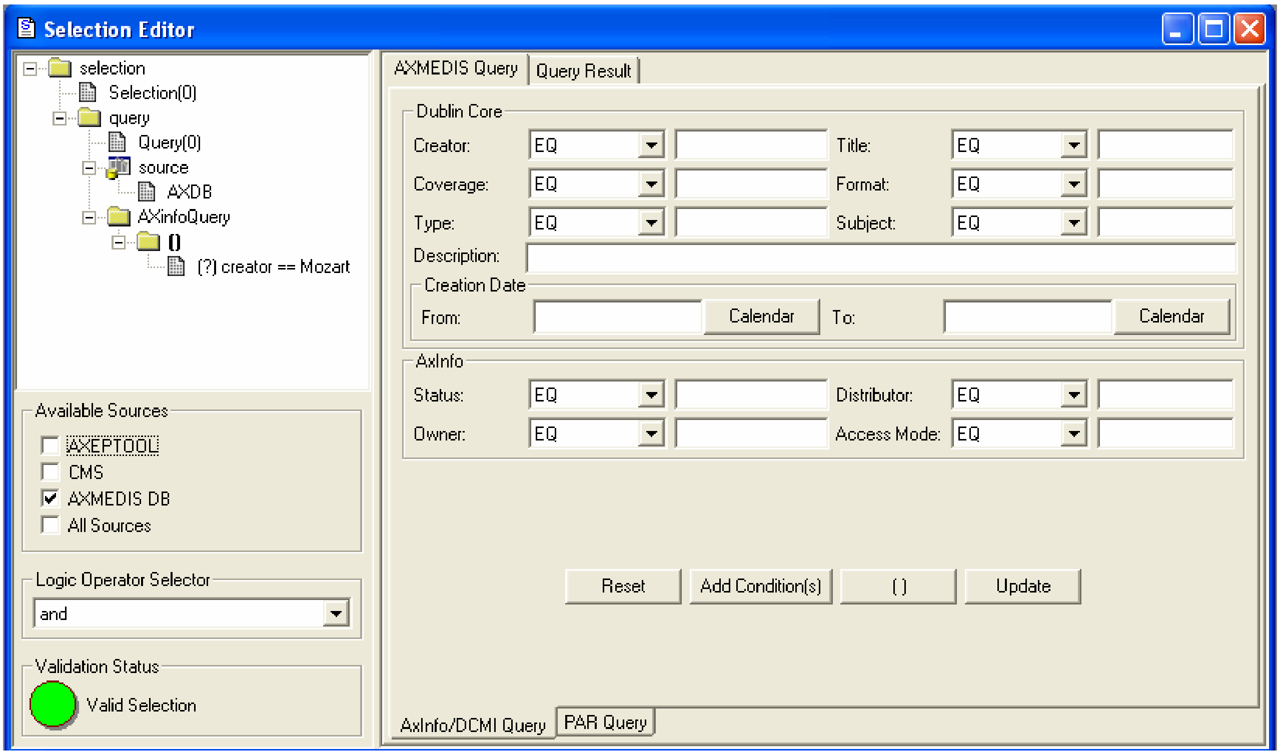
14.3 User interface description

Selection Editor Panel Full mode

The Selection Editor Panel is the main window of the Selection Editor UI. It is split in two main area divided by a draggable sash. As depicted in the following pictures, the panel is organised in subpanels:

- *Selection Tree View* displays a simplified tree view of the selection document
- *Available Sources* provides a set of choices about the sources where to submit querires
- *Logic Operator* contains a listbox to choice the logic operator to be used for connecting conditions
- *Validation Status Panel* displays the current status of the xml document according to the adopted schema for the selection document
- *Query Composer & Result Viewer Area* is a notebook and provides two pages:
    - *Query Composer* page for editing, creating, adding conditions to a query
    - *Result Viewer* page for displaying results of a submitted query or whole selection





selection
Selection(0)
query
Query(0)
source
AXDB
AXinfoQuery
()
(?) creator == Mozart

Available Sources
□ AXEPTOOL
□ CMS
☑ AXMEDIS DB
□ All Sources

Logic Operator Selector: and

Validation Status: ● Valid Selection

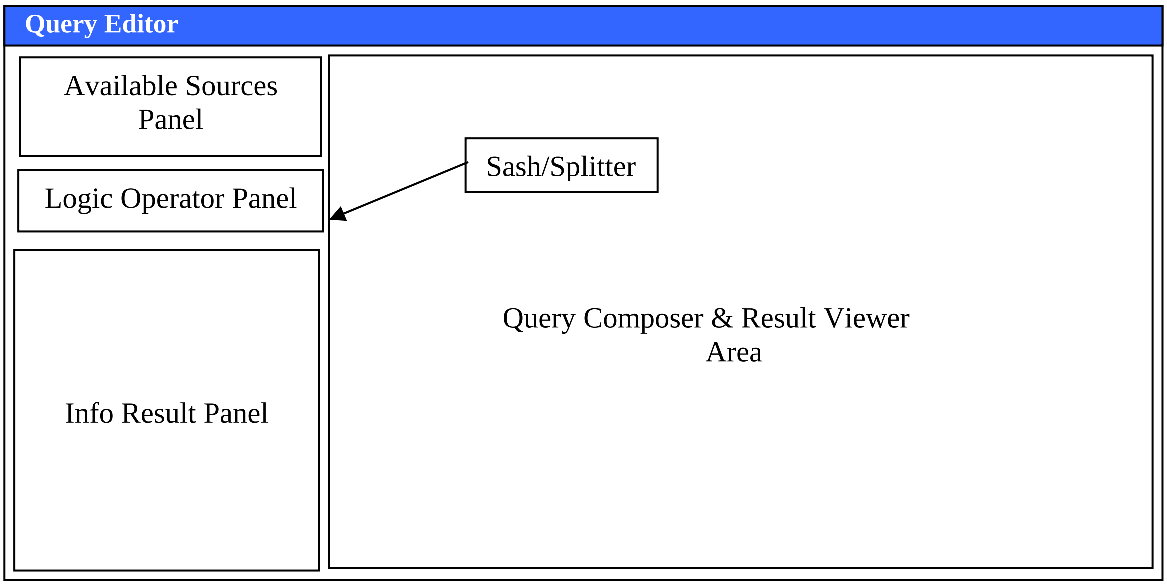
Creator: EQ; Title: EQ; Coverage: EQ; Format: EQ; Type: EQ; Subject: EQ; Status: EQ; Distributor: EQ; Owner: EQ; Access Mode: EQ

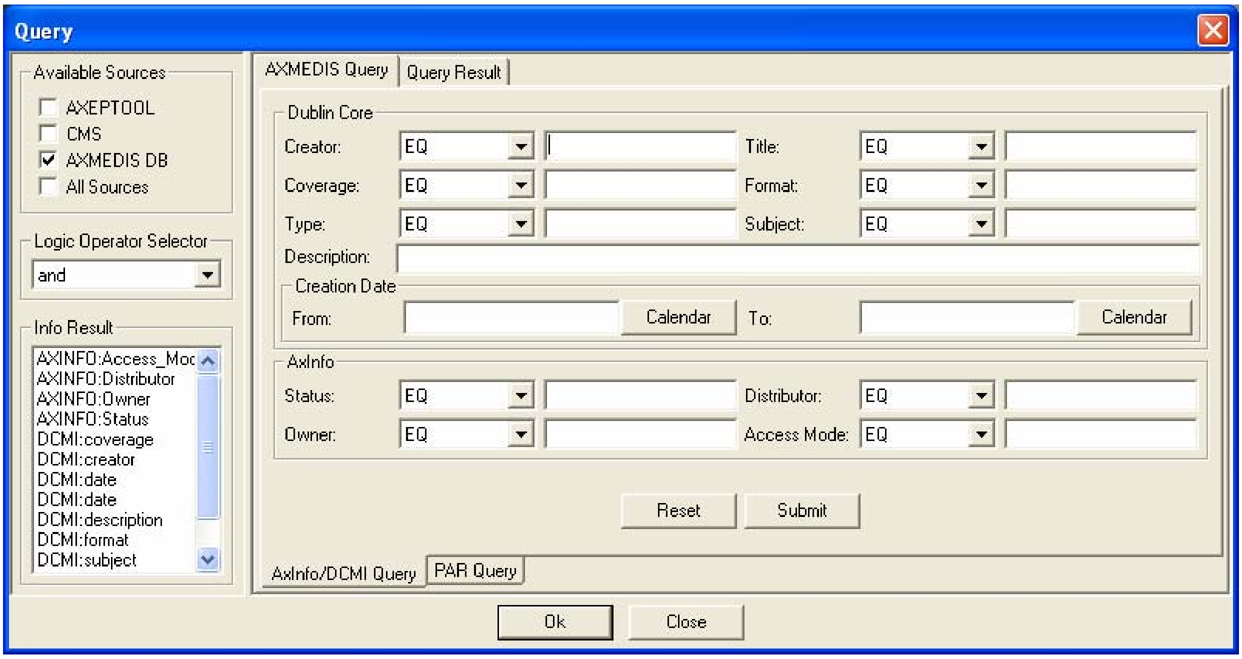
Selection Editor Panel as Query User Inteface

The Selection Editor used as Query User Inteface is split in two main area divided by a sash. As depicted in the following pictures, the panel is organised in subpanels:

- *Available Sources* provides a set of choices about the sources where to submit querires
- *Logic Operator* contains a listbox to choice the logic operator to be used for connecting conditions
- *Info Result Panel* displays the list of fields to use as information feeback from the Main Query Support
- *Query Composer & Result Viewer Area* is a notebook and provides two pages:
    - *Query Composer* page for editing, creating, adding conditions to a query, in this mode the composer allows submitting the current set of conditions (single query) to the Main Query Support.
    - *Result Viewer* page for displaying results of a submitted query or whole selection

Note: In this mode, the conditions of query are logically connected by the operator selected in the logic connector panel: query in OR or query in AND.





☐ AXEPTOOL
☐ CMS
☑ AXMEDIS DB
☐ All Sources

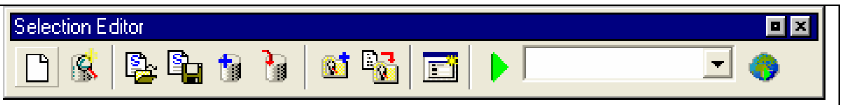
Logic Operator Selector: and

AXINFO:Access_Moc
AXINFO:Distributor
AXINFO:Owner
AXINFO:Status
DCMI:coverage
DCMI:creator
DCMI:date
DCMI:date
DCMI:description
DCMI:format
DCMI:subject

Creator: EQ; Title: EQ; Coverage: EQ; Format: EQ; Type: EQ; Subject: EQ; Status: EQ; Distributor: EQ; Owner: EQ; Access Mode: EQ

Selection Editor ToolBar

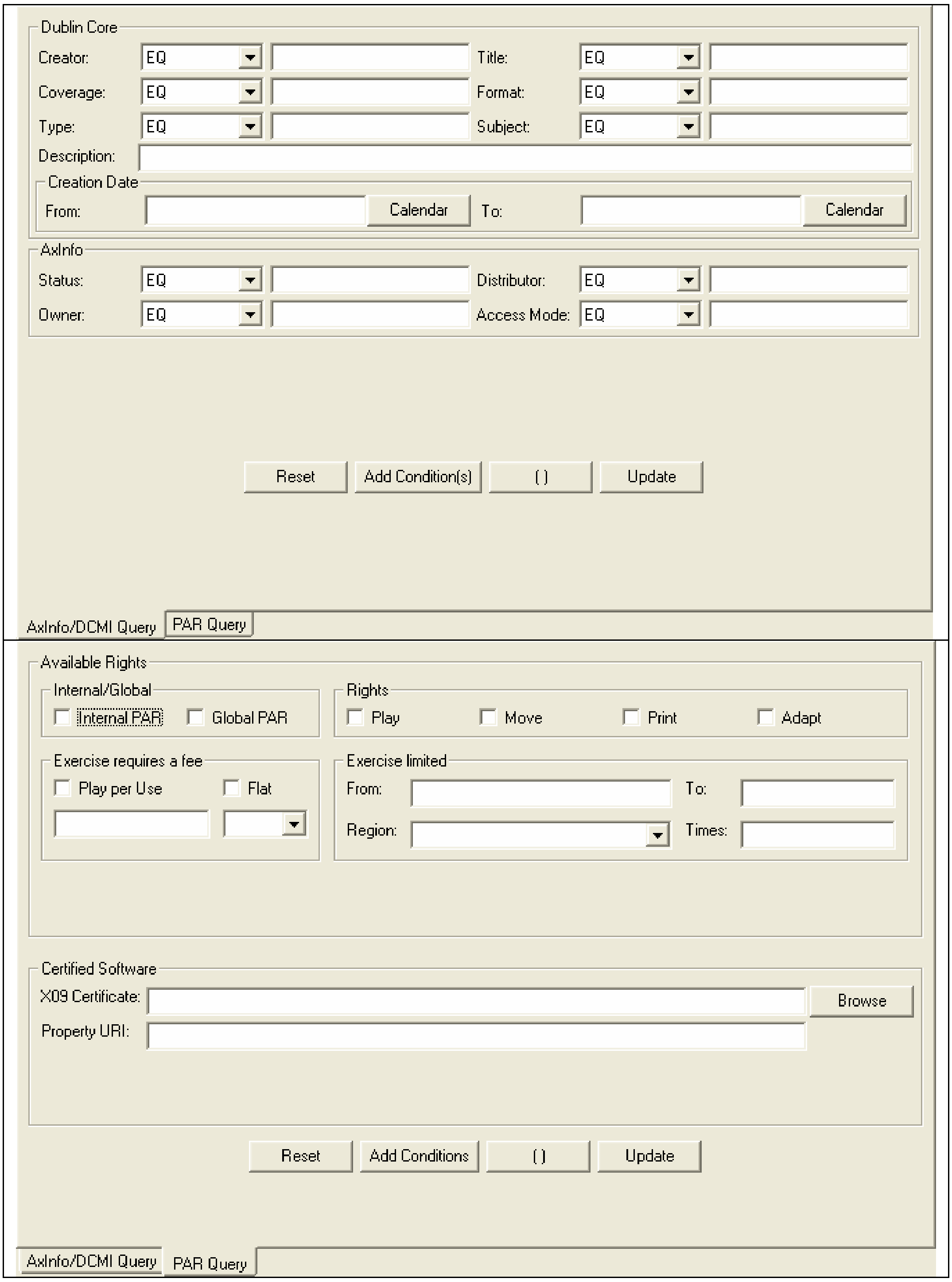
The selection editor provides a quick access to functions by means the following toolbar.

Tool List 1. Clear the selection 2. Insert a New Query 3. Open a Selection from disk 4. Save a Selection on disk 5. Import Selection from DB 6. Export Selection into DB 7. Import Query from Disk 8. Export Query to disk 9. Customize Query Panels 10. Run query 11. Selected query 12. Run selection	

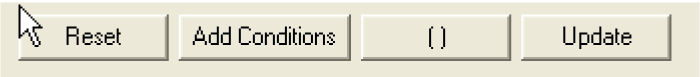
Query Composer Panel

The Query Composer Panel has been designed to work in two modes: Selection Mode and Single Query Mode

Selection Mode - In this mode, the Query Composer allows editing the selection document according to the XML schema and creating complex query by nesting conditions. The panel consists in a notebook with two page: AXMEDIS Query Page and the Query Result. Hereby, the AXMEDIS Query Page is described. It consists in a notebook and two pages as reported in the following picture. The first page provides a predefined set of fields organized by context: Dublin Core and AxInfo metadata. The second page provides a predefined set of fields regarding the DRM conditions.

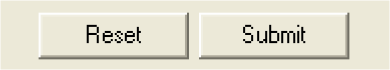


The user can edit and add those metadata to the current query by means the set of buttons in both page. Buttons are reported in the following picture



Reset – Clear the filled fields in the panel
Add Conditions – Insert conditions in the query by adding new filled fields
() – Insert a nesting level in the query
Update – modify fields shown in the panel and then update them in the query.

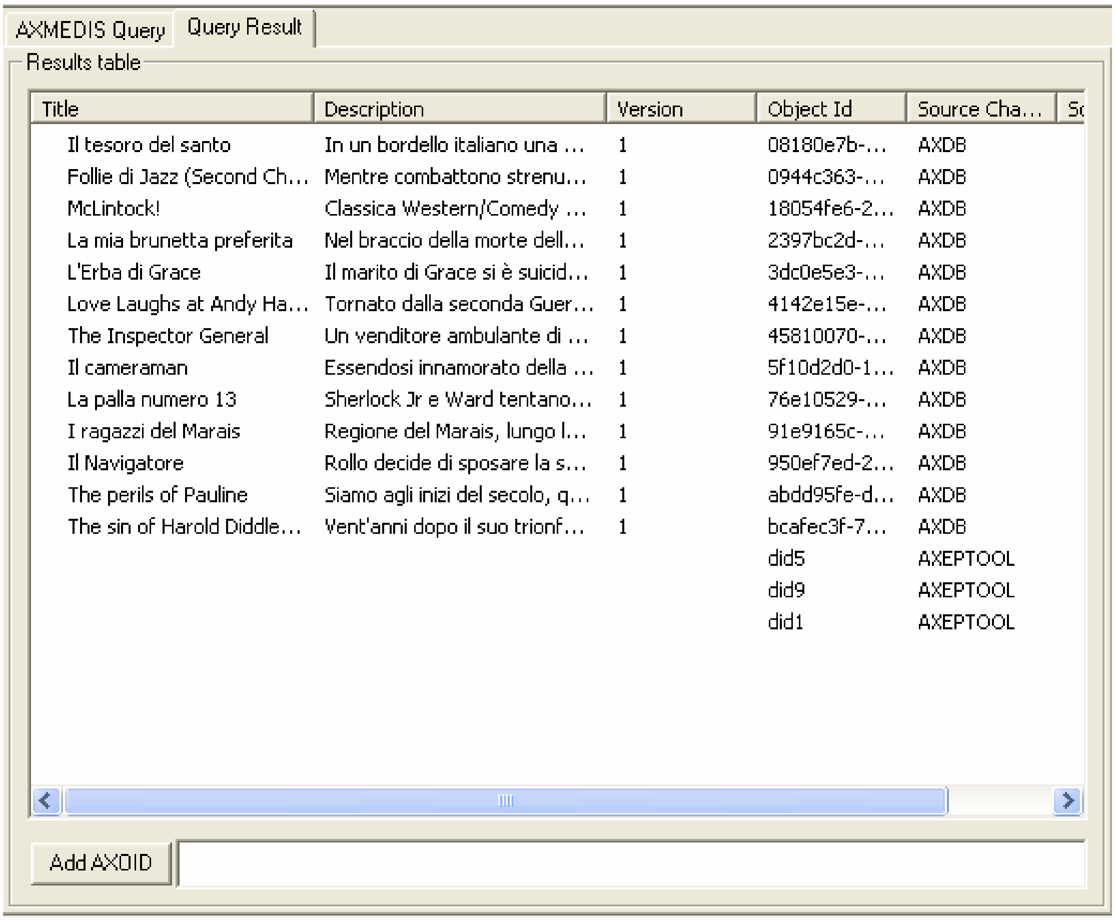
Single Query Mode
In this mode, the Query Composer allows editing the selection document consisting in a single query and according to the XML schema. This is a simplified version that does not allow creating a complex query by nesting conditions. The panel consists in a notebook with two page: AXMEDIS Query Page and the Query Result. Hereby, the AXMEDIS Query Page is described. It differs from the Selection Mode only for the set of buttons:



Reset – Clear the filled fields in the panel
Submit – Create the query and submit it to the Main Query Support

The user can edit and add those metadata to the current query but he can realize simple query.

Query Result Panel
The Query result panel reports results for the actualization of a query or a selection. Results are displayed in list report mode. The panel provides a popup menu and allows multiple selections by means of the mouse.



Title	Description	Version	Object Id	Source Cha...	Sc
Il tesoro del santo	In un bordello italiano una ...	1	08180e7b-...	AXDB	
Follie di Jazz (Second Ch...	Mentre combattono strenu...	1	0944c363-...	AXDB	
McLintock!	Classica Western/Comedy ...	1	18054fe6-2...	AXDB	
La mia brunetta preferita	Nel braccio della morte dell...	1	2397bc2d-...	AXDB	
L'Erba di Grace	Il marito di Grace si è suicid...	1	3dc0e5e3-...	AXDB	
Love Laughs at Andy Ha...	Tornato dalla seconda Guer...	1	4142e15e-...	AXDB	
The Inspector General	Un venditore ambulante di ...	1	45810070-...	AXDB	
Il cameraman	Essendosi innamorato della ...	1	5f10d2d0-1...	AXDB	
La palla numero 13	Sherlock Jr e Ward tentano...	1	76e10529-...	AXDB	
I ragazzi del Marais	Regione del Marais, lungo l...	1	91e9165c-...	AXDB	
Il Navigatore	Rollo decide di sposare la s...	1	950ef7ed-2...	AXDB	
The perils of Pauline	Siamo agli inizi del secolo, q...	1	abdd95fe-d...	AXDB	
The sin of Harold Diddle...	Vent'anni dopo il suo trionf...	1	bcafec3f-7...	AXDB	
			did5	AXEPTOOL	
			did9	AXEPTOOL	
			did1	AXEPTOOL	

Selection Tree View Panel
It shows the tree structure of the selection document. It provides a dynamic popup menu with a set of functions that allow managing the information directly from the tree view.

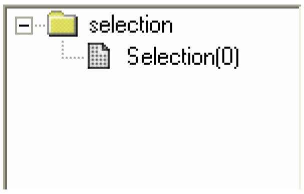
14.4 Technical and Installation information

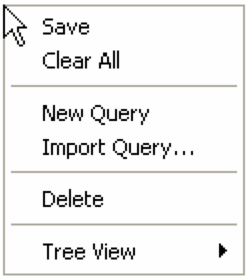
not available yet.

References to other major components needed	
Problems not solved	•
Configuration and execution context	

14.5 Draft User Manual

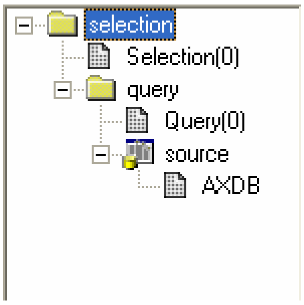
This draft manual describes how to create a new query in the Selection Document when the Selection Editor is used in the SELECTION MODE. Part of this manual is valid for the SINGLE QUERY MODE.
We assume to start from an empty selection document, but the procedure can be applied with a prefilled selection to insert further queries.



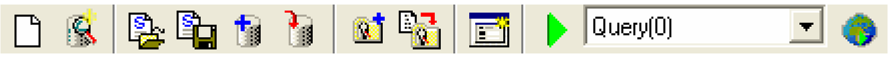


To add a new query the user can:

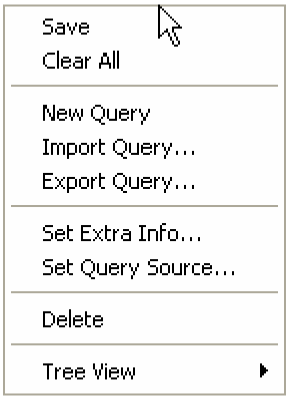
- Access to the popup menu on the tree view by right clicking on the "selection" item and then selecting the "New Query" function
- Use the AddQuery button of the toolbar



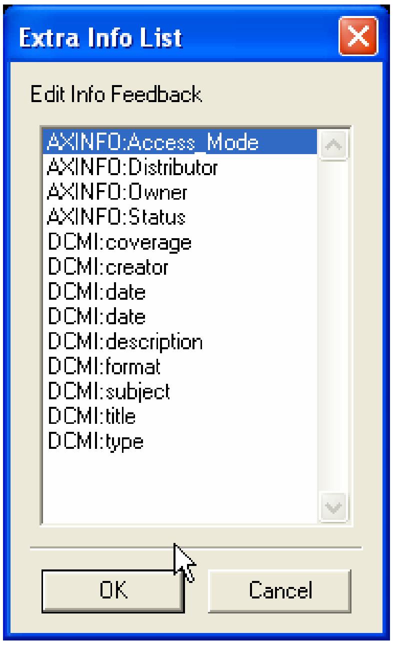
The new query becomes the current query and all operations affect it. The current query is displayed in the toolbar:



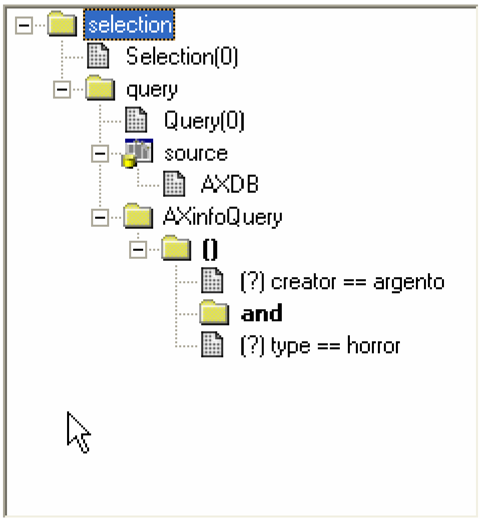
The new query is inserted and is filled with the source data choices currently set on the Available Sources. The source can be modify before adding or after the query. In both case the user has to selects sources in the corresponding panel. After the query insertion by right clicking on the new "query" item the popup menu on the tree view the user can select the "Set Query Source" function to apply the new sources.



By means the Set Extra Info function on the same popup menu, the user can fill the query with the list of information to retrieve when the query is submitted to the Main Query Support. The function opens a multiple choice dialog as following:

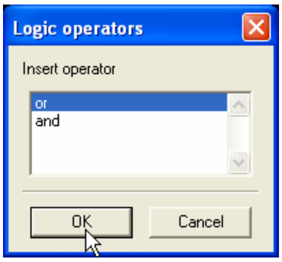


The user can start fo insert conditions in the query by filling the fields in the Query Composer in a single or both panel. After filling, the user has to press the “AddCondition(s)” button to add them in the query.

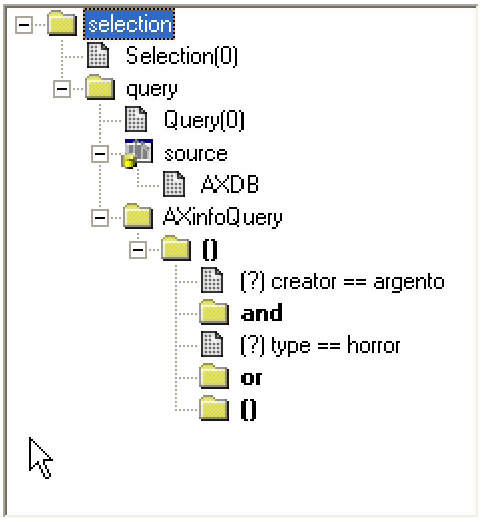


```
selection
    Selection(0)
    query
        Query(0)
        source
            AXDB
        AXinfoQuery
            ()
                (?) creator == argento
                and
                (?) type == horror
```

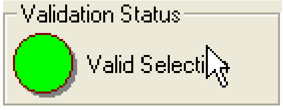
To add a nesting level, the user has to press the “()” button in the Query Composer. A dialog will appear asking for the logic connector to use

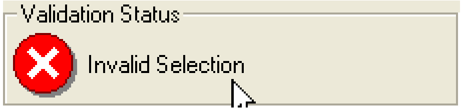


The “( )” will be inserted in the query and it will be the current level that can be edit. To change level the user has to select an existing nesting level.



During the editing the document is validated at runtime and the Validation Status panel provides the current status of the document:





To remove an item the user can select the item on the tree and by accessing to the popup menu he call the "delete" function.

14.6 Examples of usage

The Selection Editor has been integrated in:

1. AXCP Rule Editor in full mode
2. AXMEDIS Editor as Query User Interface
3. Pnp Editor as Query User Interface

14.7 Integration and compilation issues

In this version, the schema location for the selection and result xml document has to be initialized via software by the main application.

To host the Selection Editor in the full mode in own application is necessary to set the `axSELECTION_MODE` in the costructor. Otherwise to host the simply version for making single query it is necessary to set the `axSINGL_EQUERY_MODE` in the costructor. By default the mode is `axSELECTION_MODE`.

The following piece of source code reports the definition of the `wxEVT_SELECTION_ADD_AXOID` event according to the wxWidgets macro mechanism:

```
BEGIN_DECLARE_EVENT_TYPES()
    DECLARE_LOCAL_EVENT_TYPE(wxEVT_SELECTION_ADD_AXOID, -1)
END_DECLARE_EVENT_TYPES()

#define EVT_SELECTION_ADD_AXOID(func) \
            DECLARE_EVENT_TABLE_ENTRY( wxEVT_SELECTION_ADD_AXOID, \
                            -1,                          \
                            -1,                          \
                            (wxObjectEventFunction)      \
                                         (wxEventFunction)            \
                            (wxCommandEventFunction) & func, \
                            (wxObject *) NULL ),
```

The event can be used to receive the feeback from the Query Result Viewer panel when the “Add Axoid” command is called. This allows retrieving the selected element.

14.7.1 Registration of events on the main application and getting the toolbar

The toolbar is available only in the Selection Editor Full mode. In the current version the toolbar has been realized using the wxDynamicToolbar class of the FL library provided by the wxWidgets as contribution. A toolbar wxToolBar based will be implemented in the next version.
The `createDynToolBar` is the method to be used to ask for adding a dynamic toolbar. It needs to know the mpLayout pointer for instance of `wxFrameLayout`.
The `connectEvents` is the method of the wxWidgets library that allows connecting events of Selection Editor to the Event Handler of the main application.

This is an example how events and the toolbar of the Selection Editor have been added to the AXCP Rule Editor.

```
wxMDIChildFrame* AxRuleEditorFrame::newSelectionEditor(AxRuleItemData* item,wxString *document)
{
    if(childFrames.GetCount()<MAXCHILD && AxSelectionEditorPanel::getInstance()==NULL)
    {
            AxSelectionEditorFrame* ptr = new AxSelectionEditorFrame(this,item,axID_RULE_SELECTION_EDITOR);
            childFrames.Append((wxMDIChildFrame*) ptr);
            wxDynamicToolBar *bar =AxSelectionEditorPanel::getInstance()->createDynToolbar(mpLayout);
            short pos = GetMenuBar()->FindMenu("Workflow");
            if(pos!=-1)
            GetMenuBar()->Insert(pos+1,AxSelectionEditorPanel::getInstance()->getMenu(),"&"+AxSelectionEditorPanel::getInstance()-
>GetName());
            AxSelectionEditorPanel::getInstance()->connectEvents(this);
            AxSelectionEditorPanel::getInstance()->setSelectionDoc(document, item->getLabel());
            if(bar)
            {
                    mpLayout->RefreshNow();
                    bar->Refresh();
            }
            updateMenu();
            return ptr;
    }
    else
            return NULL;
}
```

14.8 Configuration Parameters

Configuration parameters are stored in the configuration file of the application that hosts the instance of Selection Editor. Parameters are mainly used by the Selection Document and define the URL for the internal webServices clients. The following table reports them and an excerpt of a configuration file with the section regarding the AXMEDIS_SELECTION module.

Config parameter	Possible values
QUERY_SUPPORT_WSDL	http://bellini-mobile:8080/MainQuerySupportWS/mqs?WSDL
SELECTION_ARCHIVE_WSDL	http://bellini-mobile:8080/UserSelectionArchiveWS/sa?WSDL

```
<Module category="" id="AXMEDIS_SELECTION">
    <Parameter name="MAIN_QUERY_SUPPORT_WSDL" type="string">http://bellini-
mobile:8080/MainQuerySupportWS/mqs?WSDL</Parameter>
    <Parameter name="SELECTION_ARCHIVE_WSDL" type="string">http://bellini-
mobile:8080/UserSelectionArchiveWS/sa?WSDL</Parameter>
  </Module>
```

14.9 Errors reported and that may occur

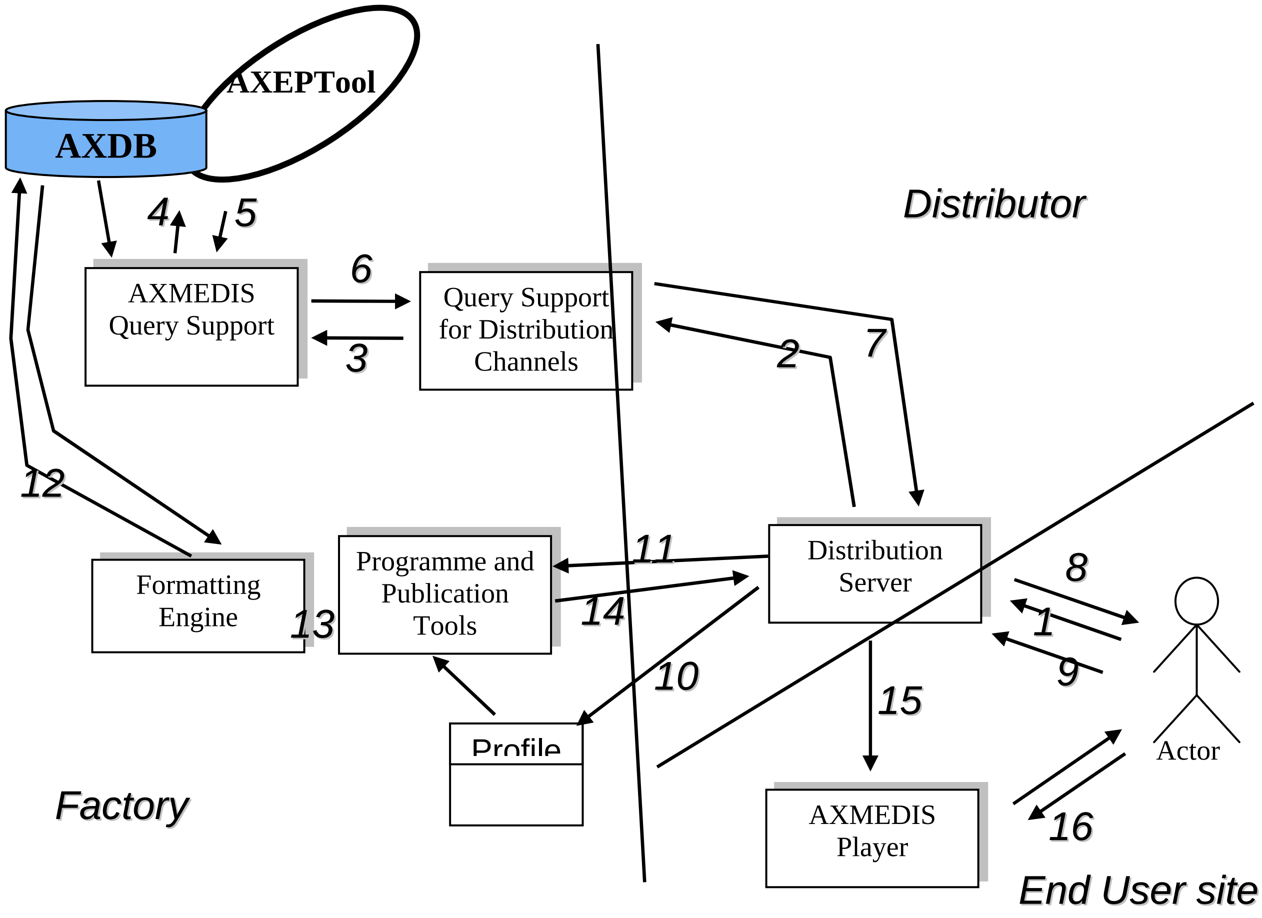
Error code	Description and rationales

15 Query Support for Production on Demand (FHGIGD)

Module/Tool Profile	
Query Support for Production on Demand (Query Support for Distribution Channels)	
Responsible Name	Peter Ebinger
Responsible Partner	FHGIGD
Status (proposed/approved)	Approved
Implemented/not implemented	Implemented
Status of the implementation	Not integrated with distribution channels
Executable or Library/module (Support)	Module
Single Thread or Multithread	Multithread
Language of Development	Java
Platforms supported	All platforms that support Java
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/Framework/source/query_on_demand/distribution_channel/
Reference to the AXFW location of the demonstrator executable tool for internal download	Internal module without user interface, see section “Query Support for Clients “for demonstatrators.
Reference to the AXFW location of the demonstrator executable tool for public download	Internal module without user interface, see section “Query Support for Clients “for demonstatrators.
Address for accessing to WebServices if any, add accession information (user and Passwd) if any	No web service provided
Test cases (present/absent)	present
Test cases location	http://www.axmedis.org/documenti/view_documenti.php?doc_id=1781, test case TC10.8,
Usage of the AXMEDIS configuration manager (yes/no)	no
Usage of the AXMEDIS Error Manager (yes/no)	no

Major Problems not solved	-- --	
Major pending requirements	-- --	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Distribution Server	Distribution Server	SSL
Query Support for Clients	Query Support Web Service Interface	
Formats Used	Shared with	format name or reference to a section
XML		
XSL		
Protocol Used	Shared with	Protocol name or reference to a section
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
No direct user interface	Java	
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not
XML parser	Xerces Java 2	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
Control layer for web application	Apache Struts	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
Logging framework	Apache logging, log4j-1.2.8.jar	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
JSP Standard Tag Library	Apache Jakarta	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
Web Services	Java Web Service Development Package 1.5	jwsdp-1.5-license, relaxngDatatype.jar 1.0 License

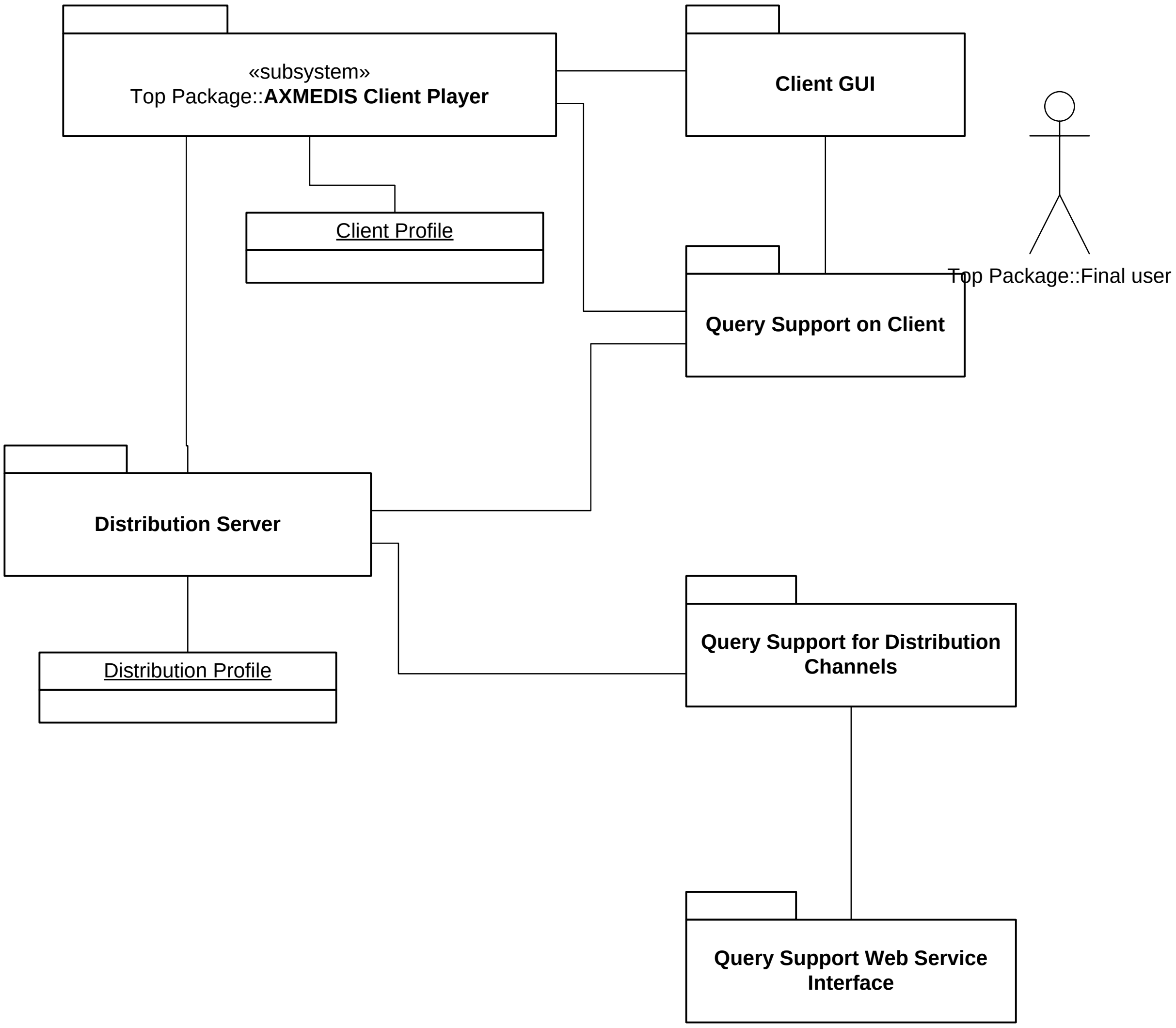
15.1 General Description of the Module



The final user can enter a query in the AXMEDIS database using the Client GUI. The Query Support on the Client creates a query message in XML based on a simple query schema as defined in the next section.

1. This XML query request including the client profile is sent to the Distribution Server.
2. The Distribution Server adds his distribution profile and passes the query on to the Query Support for Distribution Channels.
3. The Query Support for Distribution Channels verifies and adjusts the query according to client and distribution profile. The simplified XML file is transferred in a AXMEDIS XML query as described in section 11.1.2.1 “Schema for an AXMEDIS query” . Therefore, some parameters are added and adjusted according to the client and the distribution profile. Then the query is sent to the Query Support Web Service Interface.
4. The Query Support Web Service Interface passes the request into the AXMEDIS database and/or AXEPT tool.
5. The query results are returned in an XML file according to the schema defined for AXMEDIS query results in section 11.1.2.2 “Schema for an AXMEDIS query result”.
6. The query results are passed on to the Query Support for Distribution Channels.
7. The Query Support for Distribution Channels passes the query results on to the Distribution Server.
8. The query results are returned to the Query Support for Clients and presented to the final user with the Client GUI.
9. The user selects a specific content using the Client GUI and the Query Support for Clients passes this selection on to the Distribution Server.

Afterwards the Distribution Server requests the selected content using the Programme and Publication Tools and returns the results to the client. The final user can now access the newly acquired content using the AXMEDIS client player.



15.1.1 Query Support for Distribution Channels (FHGIGD)

Distribution channel can interact with AXMEDIS query support via the Query support Web service interface that is a standard technology for sharing information in Internet by using a standard protocol with a standard definition language for service interface.

The Query Support for Distribution Channels gets a client queries from the Distribution Server which include a client and a distribution profile. The Query Support for Distribution Channels verifies and adjusts ouch a query according to client and distribution profile.

Since queries are received according to a simplified XML schema, they have to be transferred in an AXMEDIS XML query as described in section 11.1.2.1 “Schema for an AXMEDIS query”. Therefore, some parameters are added and adjusted according to the client and the distribution profile. Then the queries are sent to the Query Support Web Service Interface.

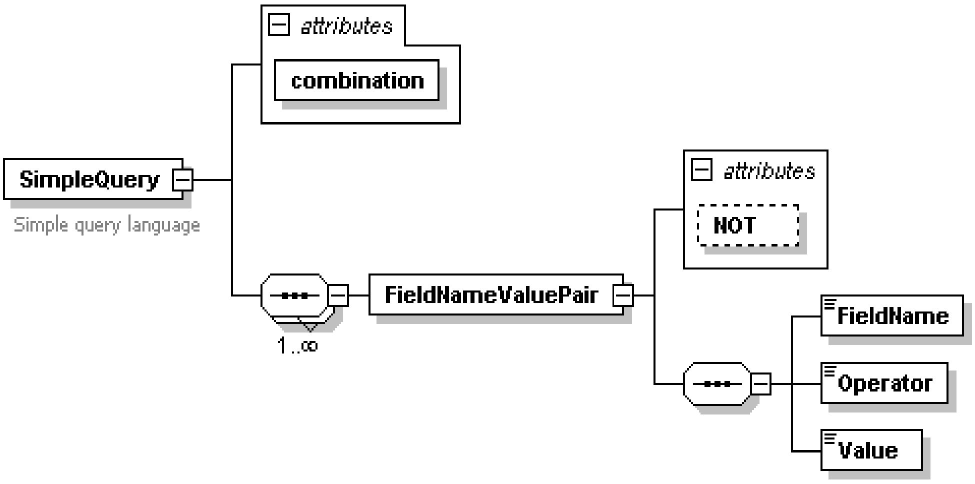
Query results are returned to the Query Support for Distribution Channels in a XML file according to the schema defined for AXMEDIS query results in section 11.1.2.2 “Schema for an AXMEDIS query result”.. and passed on to the Distribution Server.

The simple XML query language used by the final client is consists mainly of a sequence of pairs of

- field name and
- value.

Additionally simple XML query language permits the specification of a logic connection of all search terms, as well as the specification of comparison operators concerning the individual terms. Furthermore the pattern defines a range of valid values for the individual elements, which permits a simple validation of the query at an early stage.

A graphical representation of the schema can be found below the following text. A formal specification can be found in the appendix of this document.



Simple Query Schema

The field name can refer to a field in the AXINFO, e.g. to one of the following Dublin Core fields:

- **Title:** A name given to the resource.
- **Subject and Keywords:** The topic of the content of the resource.
- **Description:** An account of the content of the resource.
- **Resource Type:** The nature or genre of the content of the resource.
- **Date:** A date associated with an event in the life cycle of the resource.
- **Format:** The physical or digital manifestation of the resource.

15.1.2 Client Profile (FHGIGD)

The client profile stores information about the hardware and software capabilities of the client device. Since there is a high number and big variety of devices that are connected to the Internet, there is a corresponding need to deliver content that is tailored to the capabilities of different devices. As part of a framework for content adaptation and contextualization, a general purpose profile format is required that can describe the capabilities of a client and preferences of its user.

Specific Information for the different distribution channels can be found in the specification documents DE 9.x.1.

MPEG21 Digital Item Adaptation (DIA) Profiles

MPEG21 DIA (ISO MPEG-21, Part 7: Digital Item Adaptation) specifies the syntax and semantics of tools that may be used to assist the adaptation of digital items. The tools could be used to satisfy transmission, storage and consumption constraints, as well as Quality of Service management by the various Users

The goal is to achieve interoperable transparent access to (distributed) advanced multimedia content by shielding Users from network and terminal installation, management and implementation issues. Therefore MPEG21 DIA is used in AXMEDIS to describe the specific client device and adaptation requirements in the Client Profile.

Distribution Profile

The distribution profile stores general information about the distribution channel, e.g. bandwidth. Further information has to be available, including

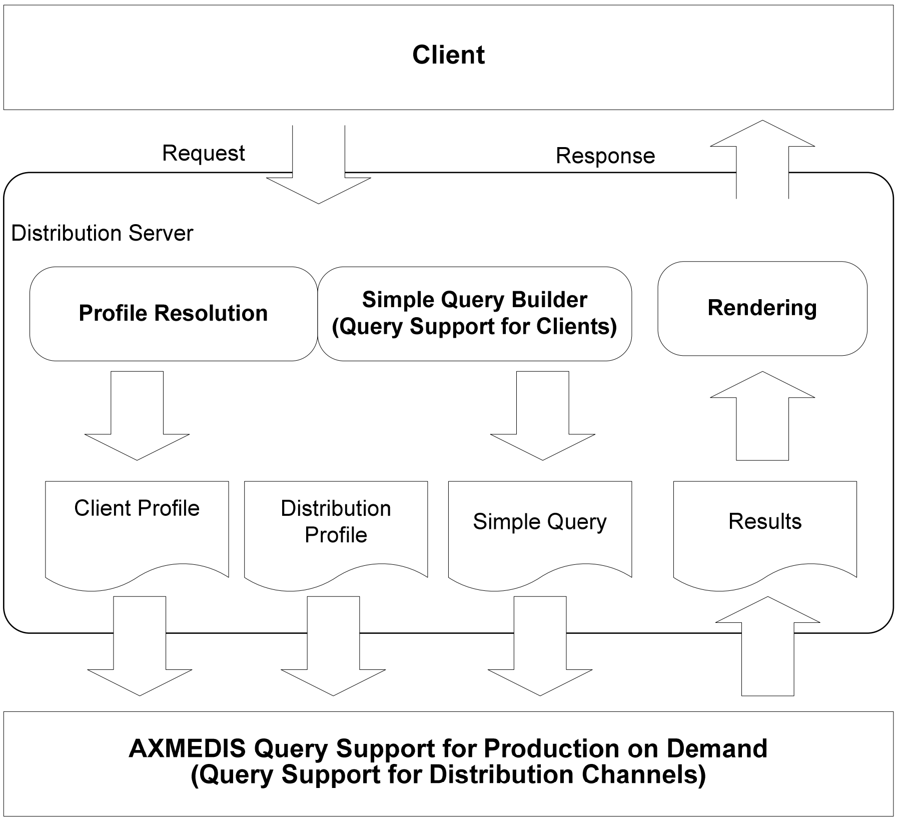
- a default client profile: This is important if a client profile is not accessible, e.g. if content is bought for a specific device different from the requesting device [OPTIONAL].
- further parameters: Distributor specific settings, e.g. whether only the local AXMEDIS database should be searched and other distributor related preferences have to be stored in the distribution profile.

15.2 Module Design in terms of Classes and Formal Description of Algorithms

Communication with the AXDB Server

This following figure describes the modules and the different processing steps that are involved during the submission of a query and the presentation of the search results.

After the Client Profile is determined the request parameters are sorted and grouped into the respective query terms. By means of the JAXB (Java XML being thing) the following chart shows the operational sequence described.

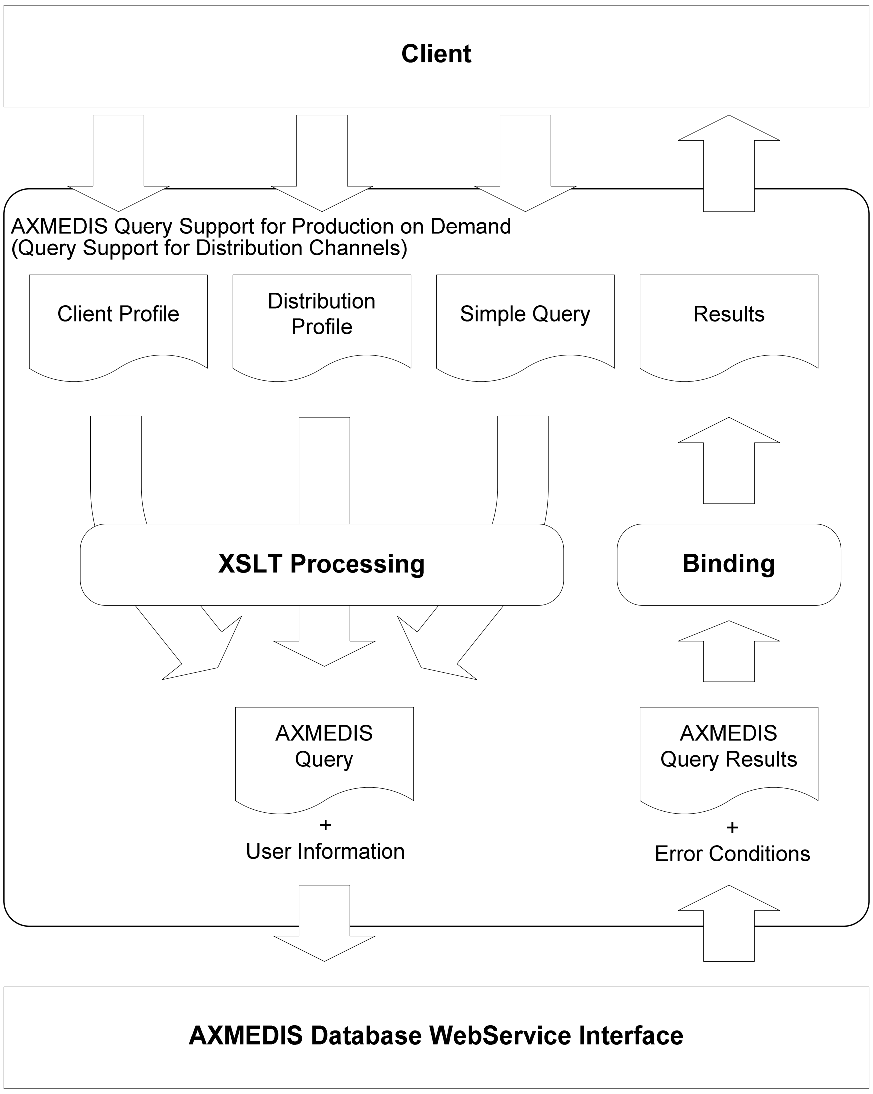

Client
Request
Response
Distribution Server
Profile Resolution
Simple Query Builder (Query Support for Clients)
Rendering
Client Profile
Distribution Profile
Simple Query
Results
AXMEDIS Query Support for Production on Demand (Query Support for Distribution Channels)

Query Transcoding and Executing

The following figure describes the modules and the processing steps that are involved during query transcoding and execution. This is the link between the different distribution channels and the AXMEDIS Query Support. The reduced scope of the Simplified Query Language for a simple interaction with the final user requires an appropriate transformation of this query language into the more complex, but thereby also more powerful, server-side AXDB query language.

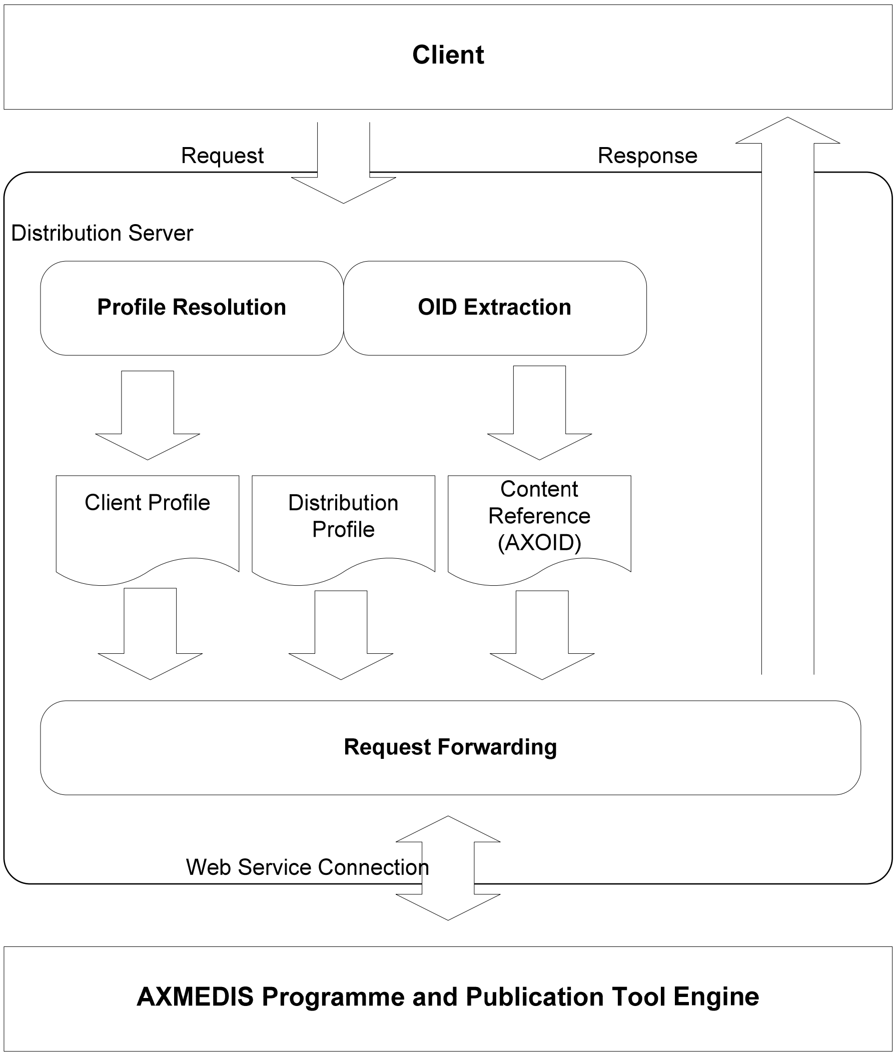
Apart from this transformation it is the task of this module to include further information from the Client and Distribution Profile into the query, in order to retrieve suitable content for the client device.

In the next step the refined query is sent via the AXDB web service interface to the AXDB server. The response of the server is again examined, checked for errors and passed on to the querying distribution channel.



Communication with the P&P Tool

The following figure describes the modules and the processing steps that are involved during the selection of a specific content from the presented list of results and the retrieval of this content from the P&P tool. After the determination of the Client and the Distribution Profile and the desired content (by its AXOID), the resulting request is sent to the Programme and Publication Tool. The response of the P&P Tools is transferred, if no errors occurred, into a response. After receiving this response, the client opens an appropriate viewer based on the returned MIME type.



15.3 User interface description

The Query Support for Production on Demand (Query Support for Distribution Channels) is a back-end module with no direct user interface. Please refer to the section about the Query Support for Clients which describes the demonstrator that provides a web interface to use the Query Support for Production on Demand.

15.4 Technical and Installation information

The Query Support for Production on Demand (Query Support for Distribution Channels) is a Java module which has to be integrated into a specific distribution channel.

References to other major components needed	Distribution Server, Query Support for Clients
Problems not solved	Integration with distribution channels.
Configuration and execution context	Java runtime environment

Please refer to the section about the Query Support for Clients which describes the demonstrator that provides a web interface to use the Query Support for Production on Demand.

15.5 User Manual

No direct user interface, please refer to Query Support for Clients.

15.6 Examples of usage

No direct user interface, please refer to Query Support for Clients.

15.7 Integration and compilation issues

Since this module is implemented in Java in can be compiled and executed on any platform that supports Java. So far no platform dependant problems are known.

15.8 Configuration Parameters

The wrong parameters have to be set in the file qod_build.properties which is located in the directory Framework/source/query_on_demand/distribution_channel.

Config parameter	Possible values
libPath	<library path>, e.g. C:/eclipse/axmedis/QOD_LIB/
catalina.home	<Catalina home directory>, e.g. C:/Programme/Apache Software Foundation/Tomcat 5.5
endpoint.address	<end point address>, e.g. -lhttp://81.73.104.125:10080/MainQuerySupportWS/MainQuerySupportWS

15.9 Errors reported and that may occur

Errors can occur when the Query Support for Production on Demand tries to access the AXDB WS interface due to misconfiguration or if the WS is not available.

16 Query Support for Clients (FHGIGD)

Module/Tool Profile	
Query Support for Clients (Query Support for PC using Web Interface)	
Responsible Name	Peter Ebinger
Responsible Partner	FHGIGD
Status (proposed/approved)	Approved
Implemented/not implemented	Implemented
Status of the implementation	Prototype implemented as demonstrator, not integrated with distribution channels.
Executable or Library/module (Support)	Module has to be deployed in Tomcat server.
Single Thread or Multithread	Single Thread
Language of Development	Java
Platforms supported	All platforms that support Java
Reference to the AXFW location of the source code demonstrator	https://cvs.axmedis.org/repos/Framework/source/query_on_demand/web_interface/
Reference to the AXFW location of the demonstrator executable tool for internal download	https://cvs.axmedis.org/repos/Framework/source/query_on_demand/web_interface/
Reference to the AXFW location of the demonstrator executable tool for public download	---
Address for accessing to WebServices if any, add accession information (user aNd Passwd) if any	---
Test cases (present/absent)	present
Test cases location	http://www.axmedis.org/documenti/view_documenti.php?doc_id=1781, test case TC10.8,
Usage of the AXMEDIS configuration manager (yes/no)	No
Usage of the AXMEDIS Error Manager (yes/no)	No
Major Problems not	---

solved		
Major pending requirements	---	
Interfaces API with other tools, named as	Name of the communicating tools References to other major components needed	Communication model and format (protected or not, etc.)
Query Support for Distribution Channels	Distribution Server	
Distribution Server		
Client Browser		
Formats Used	Shared with	format name or reference to a section
XML		
HTML		
Protocol Used	Shared with	Protocol name or reference to a section
HTTP		
Used Database name		
User Interface	Development model, language, etc.	Library used for the development, platform, etc.
Web Interface	Java	
Used Libraries	Name of the library and version	License status: GPL. LGPL. PEK, proprietary, authorized or not
XML parser	Xerces Java 2	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
Control layer for web application	Apache Struts	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
Logging framework	Apache logging, log4j-1.2.8.jar	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
JSP Standard Tag Library	Apache Jakarta	Apache Software License, http://xml.apache.org/dist/LICENSE.txt
Web Services	Java Web Service Development Package 1.5	jwsdp-1.5-license, relaxngDatatype.jar 1.0 License

16.1 General Description of the Module

16.1.1 Query Support for Clients (FHGIGD)

The query support for clients is located on the distributor side. It is either implemented on the distribution server or it is integrated into the client application.

The final user can enter a query in the AXMEDIS database using the Client GUI. The Query Support for the Client creates a query message in XML based on the query schema defined in the section above. A client profile is added to such a XML query and then it is sent to the Distribution Server.

When query results are returned to the Query Support for Clients they are presented to the final user with the Client GUI. The user can select a specific content using the Client GUI and the Query Support for Clients passes this selection on to the Distribution Server.

Query Support for PC using Web Interface

In the following is as an example the implementation of the Query Support for PC using Web Interface described.

The query support for PC using Web Interface is integrated in the distribution server. The final user can enter a query in the AXMEDIS database on the web server of the distributor using the client browser. The Query Support for the Client on the distribution server creates a query message in XML based on the query schema defined in the section above. A client profile is added to such a XML query and then it is sent to the Distribution Server.

When query results are returned to the Query Support for Clients on the distribution server they are presented to the final user on a web page. The user can select a specific content by clicking on the respective link and the Query Support for Clients passes this selection on to the P&P Engine via the Distribution Server.

16.1.2 Client Profile (FHGIGD)

The client profile stores information about the hardware and software capabilities of the client device. As described in the section about Query Support for Production on Demand the MPEG 21 Digital Item Adaptation (DIA) Profiles are used within AXMEDIS to specify the adaptation that is needed for a specific client device.

CC/PP and User Agent PROFile (**UAProf)**

Composite Capability/Preference Profiles (CC/PP) and User Agent PROFile (**UAProf)** are designed to described the capabilities of a client device and is available for many commonly used devices. Therefore they are used within AXMEDIS to gather the relevant data for the Client Profile.

The data that was gathered using CC/PP and UAProf has to be transformed into an AXMEDIS Client Profile which is based on MPEG 21 Digital Item Adaptation (DIA).

The “Structure and Vocabularies 1.0” of CC/PP can be found at http://www.w3.org/TR/CCPP-struct-vocab/.

Three different categories of information exist:

- **Hardware capabilities** of the client device include the device class (e.g. PDA or mobile phone), the vendor, the device type, its processor, RAM, and general capabilities like graphics or sound capabilities.

- **Software capabilities** include the operating system of the client and its communication, encryption, compression, rendering (e.g. installed viewers or players) capabilities.

- **User preferences** are certain setting defined by the user or owner of the device. This allows the user to set certain parameters (capabilities) according to his needs.

E.g. due to the client settings defined by the user the profile has to be stored on the client device.

The CC/PP structure covers two main areas:

- CC/PP profile components
- CC/PP profile defaults

Profile Components

The general structure of a CC/PP client profile is a two-level tree: components and attributes, with provision for each component to reference an externally defined set of default attribute values.

A CC/PP profile contains *one or more components*, and each component contains *one or more attributes*. CC/PP profiles are constructed using RDF. The RDF data model represents CC/PP attributes as named properties linking a subject resource to an associated object resource or RDF literal value.

To describe client capabilities and preferences, the client being described is a resource whose features are described by labelled graph edges from that resource to corresponding object values. The graph edge labels identify the client feature (CC/PP attribute) being described and the corresponding object values are the feature values.

RDF statement describing a client attribute:
[Client component resource] --attributeName--> (Attribute-value)

CC/PP attribute labels are represented by XML name values, which may include a namespace prefix. Attribute values may be of simple or structured data types.

Distribution Profile

The distribution profile stores general information about the distribution channel, e.g. bandwidth. Further information has to be available, including

- a default client profile: This is important if a client profile is not accessible, e.g. if content is bought for a specific device different from the requesting device [OPTIONAL].

- further parameters: Distributor specific settings, e.g. whether only the local AXMEDIS database should be searched and other distributor related preferences have to be stored in the distribution profile.

16.2 Module Design in terms of Classes and Formal Description of Algorithm

The Query Support for Clients has to compose a client query which includes a client and a distribution profile. These queries are specified in a simplified XML schema.

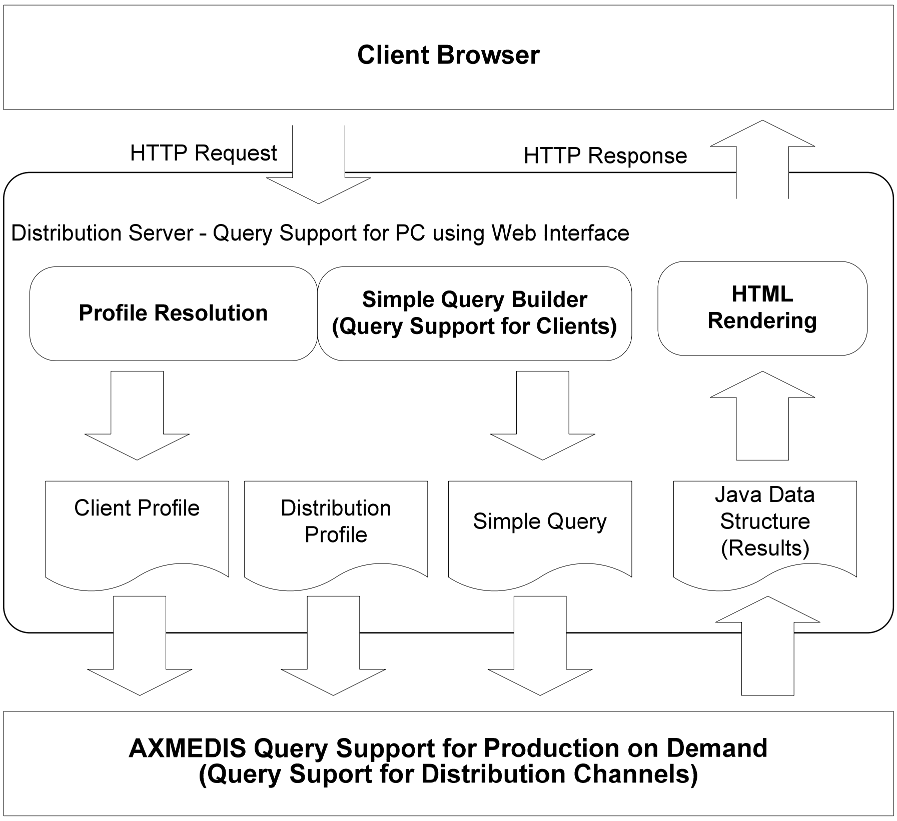
The simple XML query language used by the final client is consists mainly of a sequence of pairs of

- field name and
- value.

The field name can refer to a field in the AXINFO, e.g. to one of the following Dublin Core fields as title, subject and keywords: description, resource type, date and format: For a detailed description please refer to the section about the Query Support for Production on Demand.

The Client Profile has to be determined using CC/PP and UAProf and transformed into an AXMEDIS Client Profile based on MPEG 21 DIA.

The Query Support for Clients sorts the request parameters and groups them into the respective query terms. This is done by means of the JAXB (Java XML being thing) the following chart shows the operational sequence described.


Client Browser
HTTP Request
HTTP Response
Distribution Server - Query Support for PC using Web Interface
Profile Resolution
Simple Query Builder (Query Support for Clients)
HTML Rendering
Client Profile
Distribution Profile
Simple Query
Java Data Structure (Results)
AXMEDIS Query Support for Production on Demand (Query Suport for Distribution Channels)

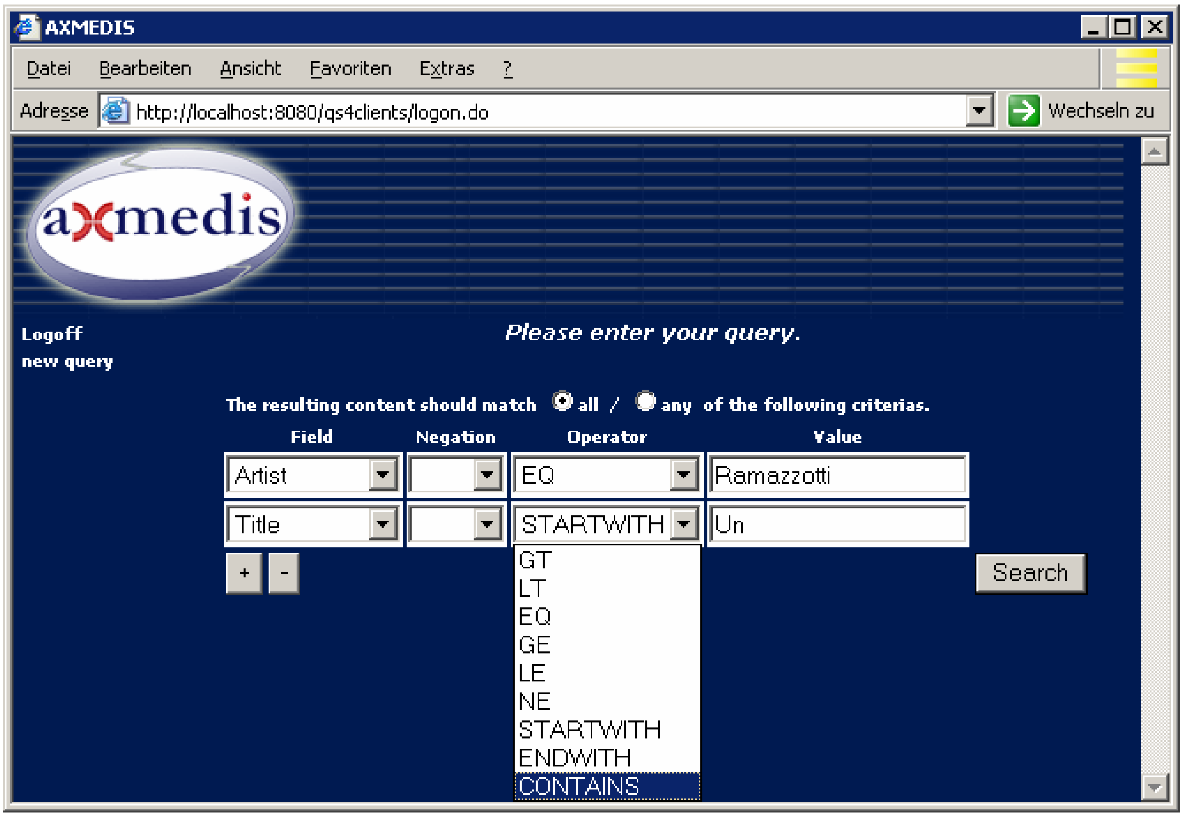
16.3 User interface description

The demonstrator web interface provides the possibility to query the AXDB-Server for information about the available multi-media content. The AXDB server provides a powerful query language that is too complex to be handled by an end user in this case. For this purpose the AXMEDIS specification defines the simplified query language schema. The user/web interface therefore creates a request based on this language schema, which is processed by the Query Support for Distribution Channels module as described in the next section.

After the server has evaluated the user request and returned the results, the demo client presents a list of matching content to the user. The user then chooses the content she/he wants to see or goes back to enter another query. If the user requests content, this request is delegated to the Programme and Publication Tools module, which prepares the content, based on the determined AXOID.

Since the AXMEDIS platform needs to support also various other devices besides personnel computers, it is necessary that the Web interface contains further functionalities. Besides the base functionality described above, the provision and determination of Distribution and Client Profiles is needed. On the one hand these profiles are used to refine the queries of a user and on the other hand to generate the requested content adapted for the specific device.

Input of Query
The user is requested to enter her/his query. This procedure is supported on the client with JavaScript to minimize the number of client/server requests and therefore to ensure a smooth and intuitive operation of the query interface.



AXMEDIS
Datei Bearbeiten Ansicht Favoriten Extras ?
Adresse http://localhost:8080/qs4clients/logon.do
Wechseln zu
axmedis
Logoff
new query
Search
+
-
GT
LT
EQ
GE
LE
NE
STARTWITH
ENDWITH
CONTAINS

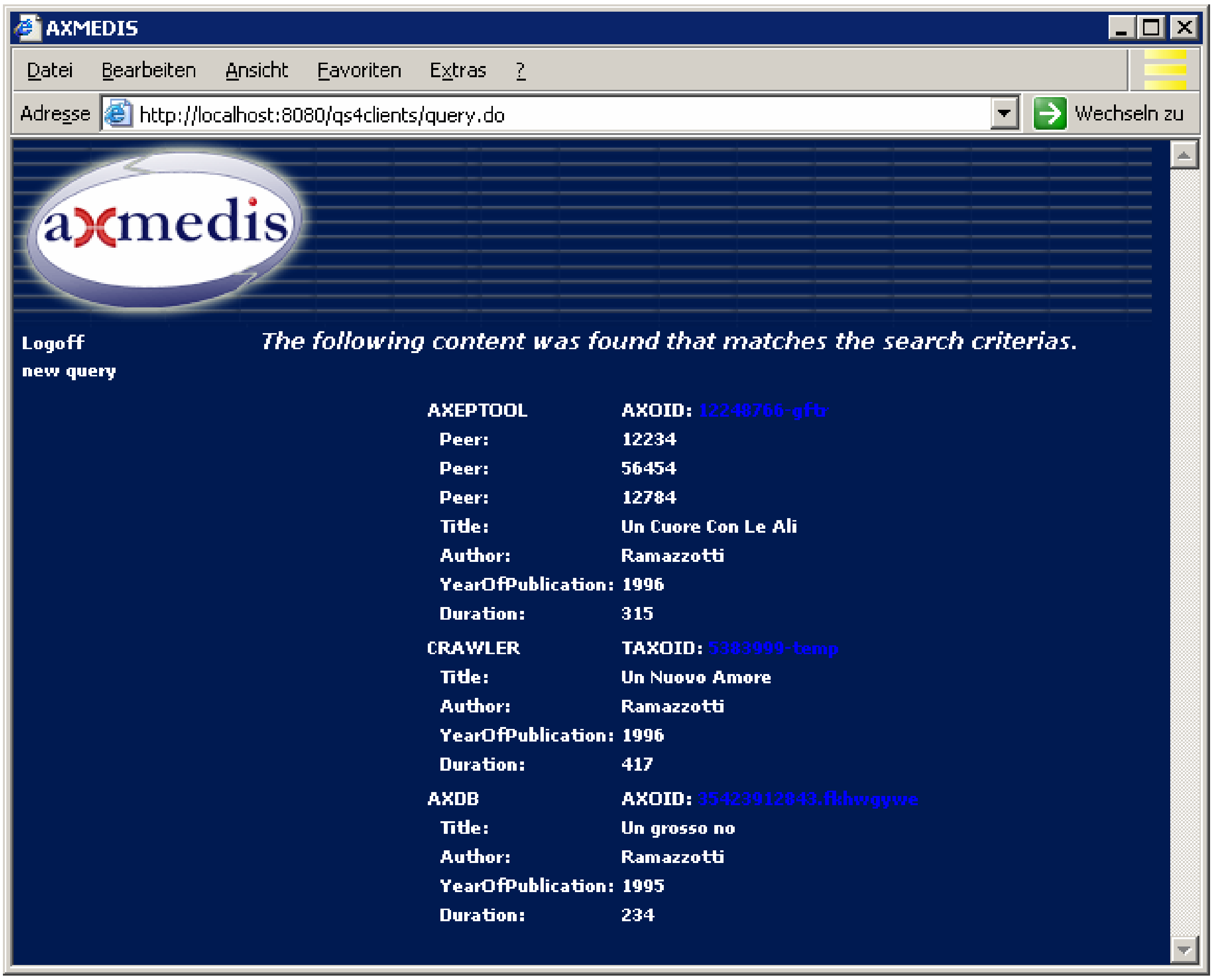
Please enter your query.

The resulting content should match ● all / ○ any of the following criterias.

Field	Negation	Operator	Value
Artist		EQ	Ramazzotti
Title		STARTWITH	Un

Display of Results

After the user query was sent to the distribution server, the results that were determined by the AXMEDIS system are presented to the user.



The following content was found that matches the search criterias.

```
AXEPTOOL              AXOID: 12248766-gftr
  Peer:               12234
  Peer:               56454
  Peer:               12784
  Title:              Un Cuore Con Le Ali
  Author:             Ramazzotti
  YearOfPublication:  1996
  Duration:           315
CRAWLER               TAXOID: 5383999-temp
  Title:              Un Nuovo Amore
  Author:             Ramazzotti
  YearOfPublication:  1996
  Duration:           417
AXDB                  AXOID: 35423912843.fkhwgywe
  Title:              Un grosso no
  Author:             Ramazzotti
  YearOfPublication:  1995
  Duration:           234
```

16.4 Technical and Installation information

References to other major components needed	Tomcat Server, Query Support for Distribution Channels (including AXDB WS Interface), P&P Tool Engine
Problems not solved	Integration with other distribution channels
Configuration and execution context	Tomcat server

16.5 User Manual

Please refer tot the “Demonstration Guide:Query Support for Production on Demand” which is available at http://www.axmedis.org/documenti/view_documenti.php?doc_id=899.

This document includes a step-by-step description of the Query Support for Clients demonstrator (Query Support for PC using Web Interface).

16.6 Examples of usage

Please refer tot the “Demonstration Guide:Query Support for Production on Demand” which is available at http://www.axmedis.org/documenti/view_documenti.php?doc_id=899.

16.7 Integration and compilation issues

Since this module is implemented in Java in can be compiled and executed on any platform that supports Java. So far no platform dependant problems are known.

16.8 Configuration Parameters

Config parameter	Possible values
tomcat_home	<tomcat directory>, e.g..C:/Programme/Apache Software Foundation/Tomcat 5.5
path	<context path for this application>, e.g. /qs4clients
url	<URL to access the Tomcat Manager application>, e.g. http://localhost:8080/manager
username	<user name to access the Tomcat Manager application>, e.g. admin
password=	<password to access the Tomcat Manager application>, e.g. admin
libPath	<library path>, e.g. C:/eclipse/axmedis/QOD_LIB
qs4dc.jar	<directory of Query Support for Distribution Channel jar file, qs4dc.jar>, e.g. C:/eclipse/axmedis/QOD_DC

16.9 Errors reported and that may occur

Errors can occur when the Query Support for Production on Demand (Query Support for Distribution Channel) is not available or if it can not process the query because the AXDB WS interface is not available. The same holds for the P&P tool engine. Another potential source of errors is misconfiguration (see section above).

17 Provided API named AXDB-Core

AXDB interface has a set of JAVA API that allows to access directly to the DAO objects that maps the DB. The API are not reported here since they are too many but the last version can be directly accessed in the AXMEDIS repository at the following address:
https://cvs.axmedis.org/repos/Framework/doc/code/axdb/index.html

Time by time the specification is updated the snapshot of the API is provided in attach to this document

18 Table description for database AXDB (EXITECH)

AXMEDIS relational DB is a general purpose relational database that can be selected among: Postgres, Mysql, Oracle, DB2 and MSSQL and therefore the system is scalable to fit the need of small end user and of corporate AXMEDIS user.

The AXMEDIS architecture must be independent by such component and must work with different databases.

Since in the database can be stored protected and unprotected objects, it is evident that the security of the unprotected object is not demanded to the database but must be guaranteed by external network equipments such as firewall and other anti intrusion tools that can limit the accesses to the object repository only to privileged network identities.

The requirements collected in the first phase of the project has defined some key feature of the general AXMEDIS Database architecture. The main requirements that have been collected in the analysis phase and that have a strong impact on the technical specification are reported and commented.

1. AXDB have to ***be independent from changes to the AXMEDIS Model structure schema***: this is a strong requirement since it impose that the AXDB schema must be flexible enough to cover changes in the AXMEDIS data model. At the implementation level, as discussed in detail in this chapter, it means that AXDB must match all the requirements of MPEG21 objects, that is all the data that can be stored in a MPEG21 object have to be mapped in the AXDB schema, and that additional metadata have to mapped with a flexible mechanism of couple (fields, value) in order follow the changes in the AXMEDIS model. This point also allow to access to the structure of the AXMEDIS object allowing the possibility of locating, extracting and searching items inside the object, if the object is not encrypted. This is true also for the part regarding the PAR and DRM.
2. AXDB has to ***manage AXMEDIS objects both in binary and xml formats***; this is implemented considering that objects can be external references (binary objects) or BLOB fields in the database capable of fitting objects embedded in XML metadata. For each object a table for taking in account if the object is locked and who owns the lock is created.
3. AXDB has to ***support versioning of AXMEDIS objects***: the versioning mechanism is intended in its simplest form, by tracking the new versions of an AXMEDIS object (given its AXOID) and storing an automatic version number when the object is uploaded to the AXDB. In order to verify if an object is a new version or a copy of an existing version, together with the object must be stored an hash value or a unique fingerprint in order to compare the object to be added with the set of versions of the same object that are already present in the AXDB. This means also that operation such as version recovering and deleting will be possible and that the elimination of an object will erase all its versions in the AXDB. When a new version of an object is uploaded AXEPTOOL has to be notified.
4. AXDB has to ***store administrative information like the transactions performed on the clients as well as the accounting information like licences, contracts etc.***: the database schema has tables that store simple administrative information such as the so called Action-Logs that can be adopted to rebuild accounting information. This means that an Action-Log is a collection of information, detailed in the following, such the object on which the operation is performed (OID), the operation performed with its parameters (if not a simple information, as play, print and so on is stored; for example for streaming also the starting point of the streaming and the duration must be taken in account), who performed the operation (UID), the tool that has been adopted (TID). These information will be available for querying also.
5. AXDB has to ***store the licences sold to the customers***: this means that AXDB has a table with three fileds, the user who received the licence, the licence issuer and the licence itself. The possibility to query the licence table will be given in terms of statistic information and not by querying inside the licenses.
6. AXDB has to ***support user management, to set what the user can do in the DB***: the user and group management can be done at different levels. By adopting the user and group capability that each DB has or by implementing an autonomous system for users and group, leaving to the DB the management of low level user access fro administrative purpose or API accessing. Since not all DBs support the same mechanism for user, groups, grant and authorization, it is better to establish API for managing user in a separate manner with respect to the Database users. A special set of tables for users, for groups and for grants will be created. Since users can be useful also for other parts of the

system (such as for Workflow), it is better to create a simplified provisioning system that manages users and groups and authentication for the whole AXMEDIS system, while grants that are specific for the tool or engines will be managed internally. This provisioning system will interface all AXMEDIS tools and engines that need to interact with user and will authenticate users, by using internal authentication or authentication by operating system procedures (Active directory for Windows, Samba or UNIX authentication for Unix like systems, etc). User activity on AXDB have to be monitored in order to track operations.

Grants that each user can have are selected among:

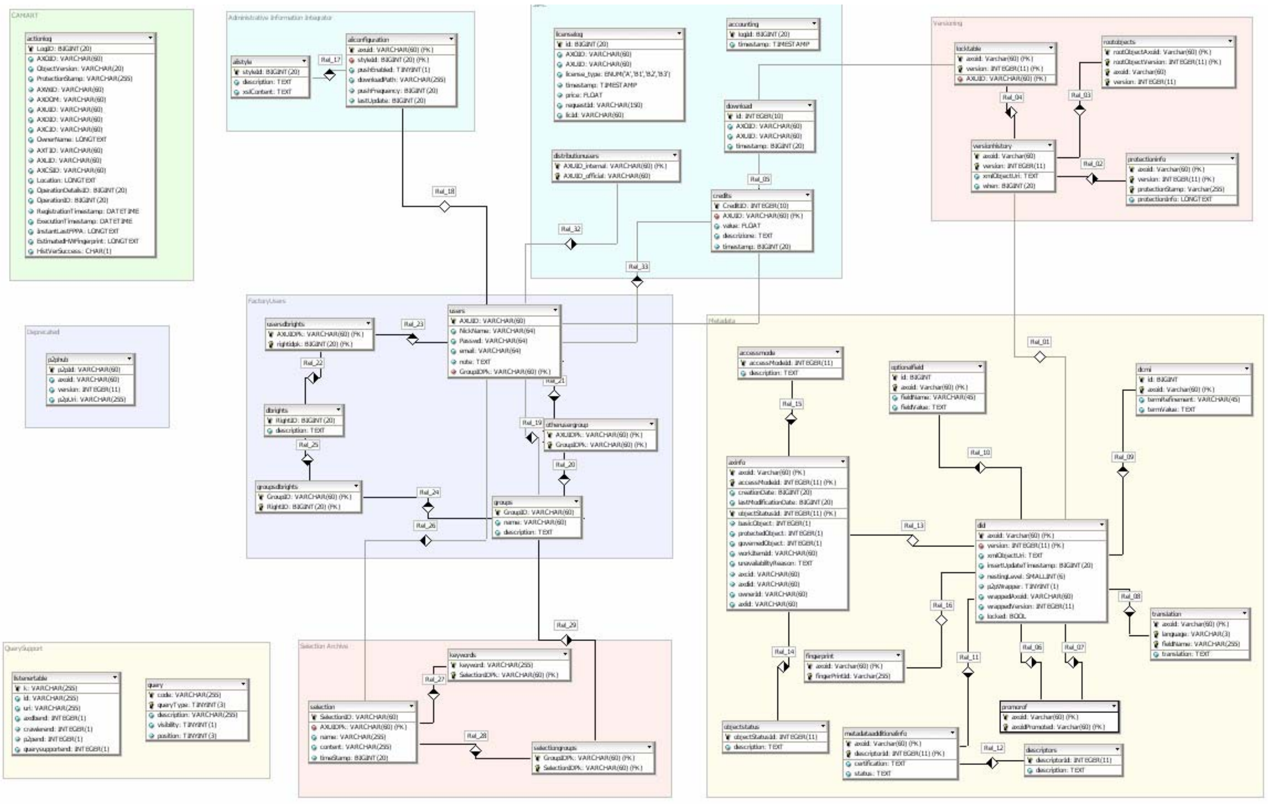
- read object
- upload new object
- lock/unlock object
- upload new version of object
- remove object
- change profile of user
- manage other users
- create users or group (this functionality is in the provisioning system)

7. ***The database has to provide efficient ways of searching inside the stored objects***: this means that DB has to provide indexing on the field that are present in MPEG21 format (the core or the system) and full text search in all the other fields. In the first prototype of the system the full text engine of a database will be employed, and in a second step it will be evaluated if a full text search engine must be created over the database structure since not all the DB support full text indexing and the full text engine that non commercial database can provide are limited in the functionalities. Customized full text indexing have to support the following characteristics: (i) list of words to be excluded from indexing, (ii) on-line indexing when the object is uploaded and not off-line indexing in batch, (iii) table for associating words with worded, in order to avoid duplication as much as possible, (iv) table for associating AXOID with the word ID in order to allow section of AXOID that match a word or a set of words. This means that a more efficient indexing will be provided for AXInfo metadata and a less efficient for other metadata not considered in the standard set.
8. ***AXMEDIS Database Administration Tools has to allow to …***: this means that a database administration tool shall exist. This tool is a web based tool for searching the AXDB, download for opening all the versions of an AXMEDIS object, manage object and manage user by linking to the provisioning system.

Several data that are contained in the AXDB are needed also in the WFDB, and therefore it is necessary to estabilish a way to sync the content of AXDB with that of WFDB. Some data are in charge to WFDB and are also present in AXDB: in that case the WFDB must communicate to AXDB any changes in such data; on the contraty, the data that are in charge to AXDB and are used also in the WFDB must be communicated to WFDB as soon as they change.

The previous list of commented requirements are the first milestones for the AXDB system specification and will be detailed in the paragraph of the different AXMEDIS components that are present in the AXMEDIS database area.

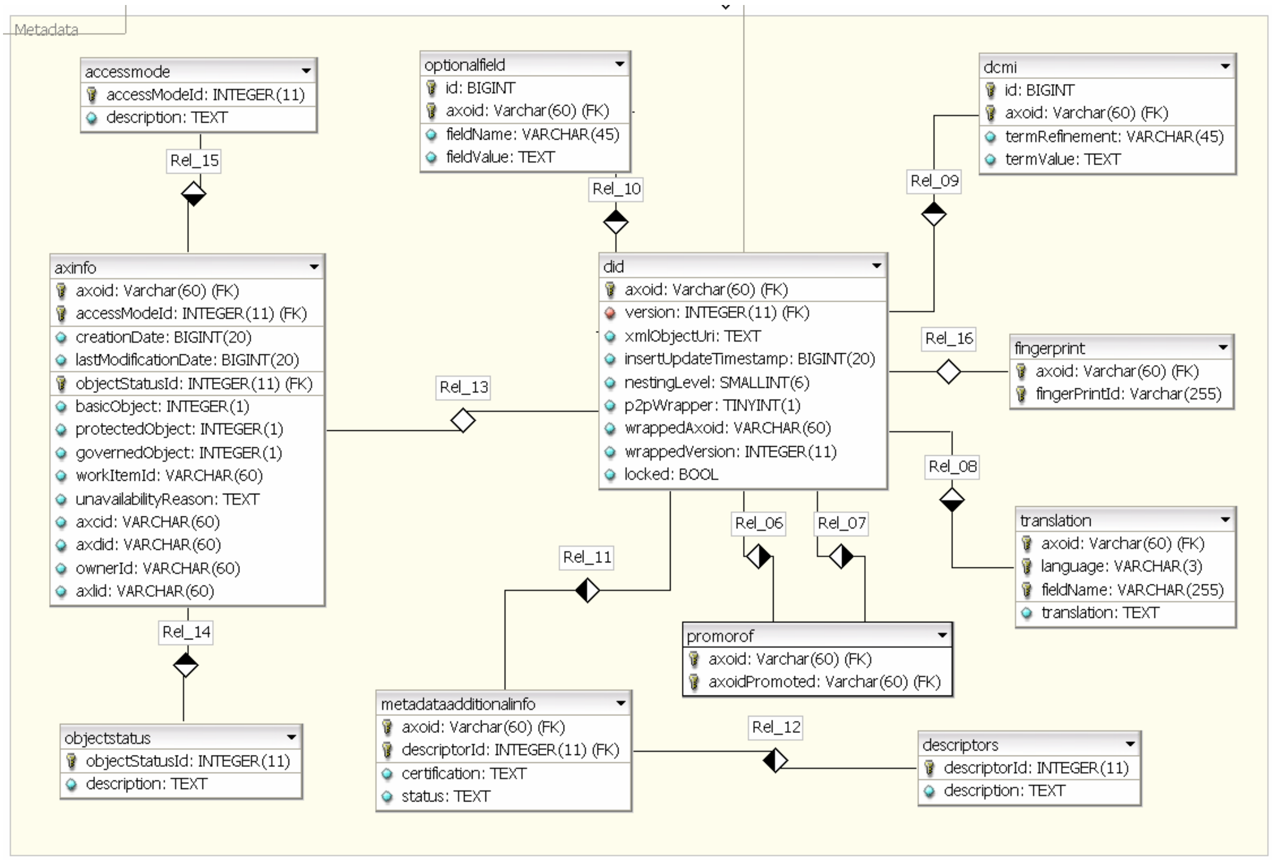
18.1 AXDB general Relation Schema (EXITECH, DSI)



18.2 Descriptors Metadata mapping in Relational Schema (EXITECH, DSI)

Inside AXDB several metadata are indexed. Among these it can be noted that a subset of the AXINFO data are collected (see axinfo table), while all the DCMI terms can be addressed by dcmi table.

Metadata mapped by the user or custom medatata are contained in the optionalfieldtable and when possible the translations are put in the translation table.



Apart from the textual description of the tables with types and relationship among keys that is reported in section 18.5, in the next subsections a conceptual description of the Tables meaning is reported.

18.2.1 AXInfo table

This table allows to store all the fields of the AXInfo of the Object in order to have a fast indexing of such mandatory fields. The model is compliant to that reported in Part A of the Specification.

18.2.2 DID

This table is present only because it is referenced by AXInfo and because it contains the information on the version of the object.

18.2.3 AccessMode

In order to reduce duplication and in order to improve search capabilities, a table for standardizing the description of AccessMode has been created. All the different AccessMode will be put there and in the AXInfo table a foreign key links to the correct Access Mode.
List of possible access mode are:

- ReadOnly
- ReadWrite
- …….

18.2.4 ObjectStatus

This table has been created with the same purpose of AccessMode for standardizing the Status of the Objects.
List of possible Object Status are:

- Published
- In production

- Not Available (in order to manage the situation in which you have an object that is no more available for you, but for which you can also have some older versions that you can still retain).
- …..

18.2.5 FingerPrint

Since each Axmedis Object can have more that one fingerprint associated, this table is created as a bridge between the Fingerprints and the AXOID. Fingerprints values are not stored in the AXDB, but in the AXCS. They are generated on the fly on the basis of the content and checked against the values in the AXCS for verification.

18.2.6 PromorOf

An Axmedis object can be a Promor of other objects. This table implements this 1:n relationships between other object (DID table) and the current object on the AXINFO.

18.2.7 Translations

It is very difficult to find a suitable way of representing translations for mandatory and optional fields. The solution is general and flexible enough to obtain such objective but is not optimized for searches.

The table has the following fields:

- Languages will be inserted following the ISO 639-2 format standard.;
- Fieldname will be the fully qualified name of the field in dotted notation (DCMICreators.Creator, DublinCore.Mediator, and so on);
- Translation, that is the translation of the Field (all the fields in the main tables are assumed to be in english, while extra languages will be stored here);
- AXOID identify the object to which the translation is referred to.

18.2.8 MetadataAdditionalInfo

This table is useful to implement the 1:n relation between the AXInfo and the certification of multiple metadata descriptors set. For the moment the model implements a number of descriptors to be standardized, but that can be also added at run time. See Descriptor Table for more details.

18.2.9 OptionalField

This table allow the insertion of a variable number of optional fields that can be mapped from the metadata of the object to the database for allowing searches. The table mainly contains a set of couples (Name, Value) related to an AXOID.

18.2.10 RootObjects

This table takes care of the objects embeeded in other objects for each object and version.

18.2.11 Dcmi

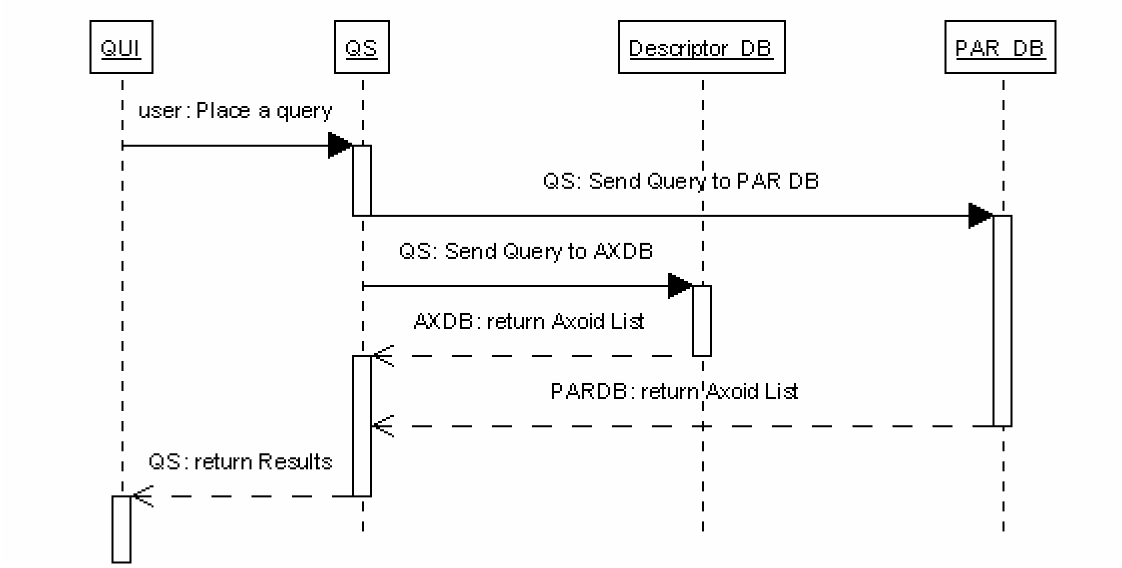
This table contains all the Dublin core terms and refinement

18.3 Integration between PAR DB and descriptors DB for making queries (EXITECH, UPC)

The model for resolving the an AXMEDIS query in a global way (and therefore finding all AXOB that satisfy both descriptors and PAR criteria) requires that QS receive the query as is, distribute the query to both Descriptor DB (AXDB Query Support) and PAR DB that should implement a similar interface.

These two subsystems will return to the query support the list of AXOID that match their criteria and therefore and AND operation is performed on the two returned set in order to get only the common AXOID.

The model can be summarized in the following diagram.



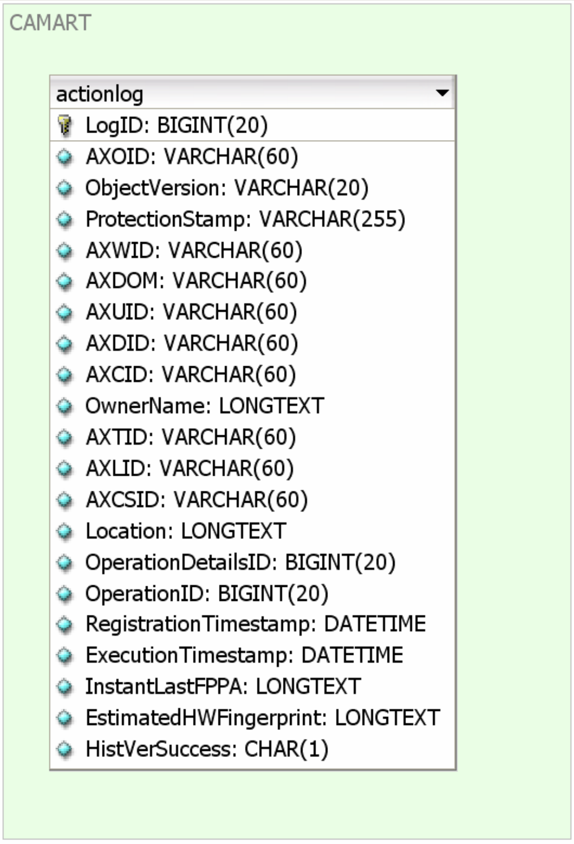
The other operation for distribution of queries will be realized in parallel, so that the results of all the sources (P2P, Crawler, other QS) will be merged together with that of AXDB.

In order to have an easy integration it is necessary that PAR DB implements the same web service interface with respect to AXDBQuerySupport as a server in synchronous manner, so that results are immediately collected by QS.

18.4 Account Log for AXMEDIS Objects repository (EXITECH, DSI)

The Action-Logs that are the basis for the accounting must be stored in the database by the Core manager and reporting Tool that read such data from AXCS.

The AXCS transfers only the results for which the user is eligible and therefore the information that are stored locally are safe from the point of view of the privacy.



CAMART

actionlog

- LogID: BIGINT(20)
- AXOID: VARCHAR(60)
- ObjectVersion: VARCHAR(20)
- ProtectionStamp: VARCHAR(255)
- AXWID: VARCHAR(60)
- AXDOM: VARCHAR(60)
- AXUID: VARCHAR(60)
- AXDID: VARCHAR(60)
- AXCID: VARCHAR(60)
- OwnerName: LONGTEXT
- AXTID: VARCHAR(60)
- AXLID: VARCHAR(60)
- AXCSID: VARCHAR(60)
- Location: LONGTEXT
- OperationDetailsID: BIGINT(20)
- OperationID: BIGINT(20)
- RegistrationTimestamp: DATETIME
- ExecutionTimestamp: DATETIME
- InstantLastFPPA: LONGTEXT
- EstimatedHWFingerprint: LONGTEXT
- HistVerSuccess: CHAR(1)

The Account Log database is structured to take care of Action-Logs and once it will be standardized will have to implement all the necessary parts of MPEG21 Event Reporting.
By now, the structure that have to be defined in more details during the specification of the single work packages is divided (apart from users and DID table that have been discussed yet or will be discussed in the rest of the document and that are reported here only as a reminder for referential integrity) in three main tables:

- ActionLog Table: this table store the Action-Log as it is with some fixed information such as the AXOID on which the operation is performed, the ActorID that performed the operation, the registration timestamp and execution timestamp (that can be different because of off-line operations performed on the objects) also a reference to an external table that will contain the OperationDetails;
- OperationDetails Table: this table will contain the details of the operations such as channel, duration and other details that are to be specified in deep. Thse details will be taken from the Operation detail table of the AXCS.

18.5 Database Schema for supporting AXMEDIS (EXITECH)

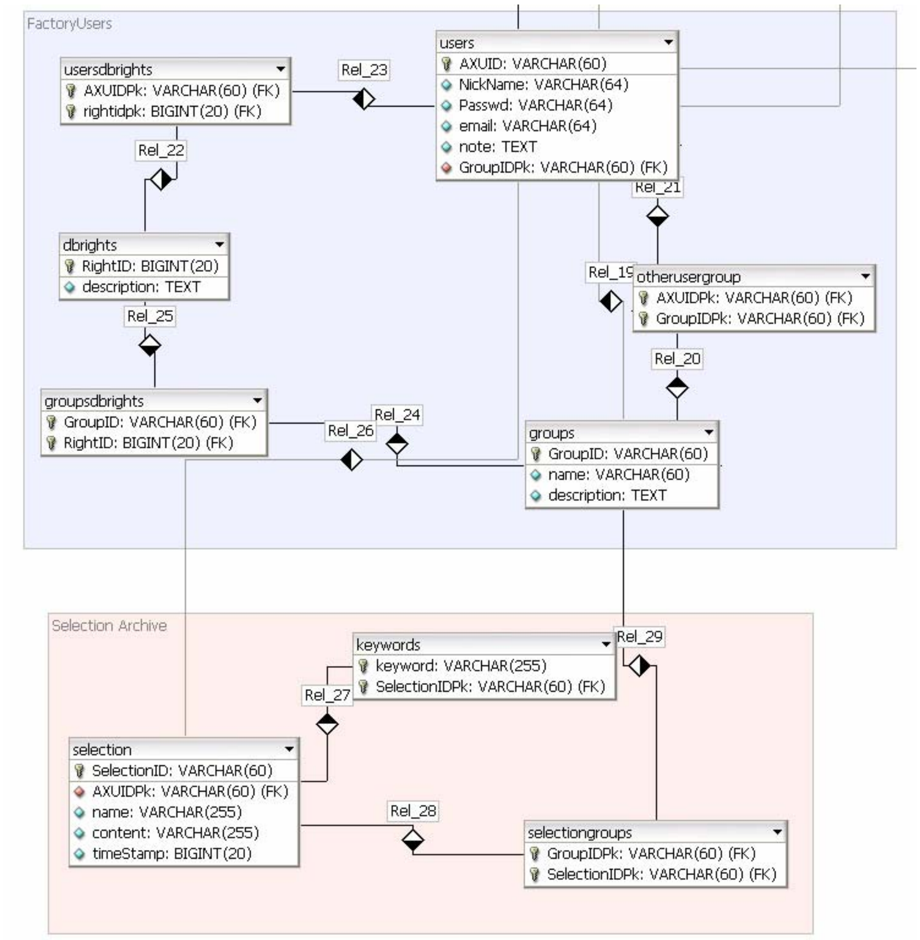
This section will cover all the aspects not covered in the previous sections such as user and group management with rights for users of the AXMEDIS system, Query and Selection Archive for Each User, etc

18.5.1 User and groups and selection archive

Regarding users a flexible and scalable approach has been followed by implementing a main table with user details (only a few are reported at the moment), main group to which an user belong and additional groups to which the user can optionally be inserted. The group table apart from a description can contain also some other fields to be defined at the moment.
The rights table lists the operation that a user can perform and to each user a set of operations are associated by the means of UserRights table.
To each group of user can also be assigned a set of rights in order to create group based rightowner.

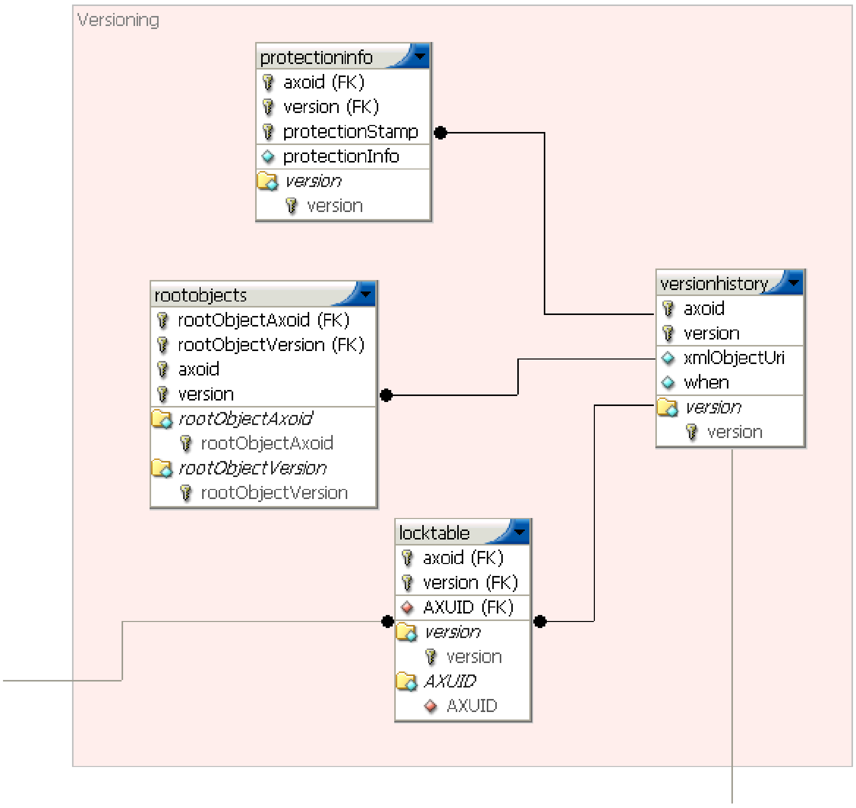
In order to support query and selection archive for each user the ER reported below has been created.
Since a query can be considered a special case for a selection (a selection with a single symbolic query inside) only selection have been addressed.
The ER is mainly based on two table:

- The selection table contains a reference to the user that has generated it, the name of the selection and the timestamp. This will allow to have different selection with the same name actualized in different moment and therefore with different timestamps. The GroupIDPk field has been inserted with the aim of giving to the selection a visibility to a group of users according to the decision of the selection author
- The SelectionContent table contains the list of items (AXOID or queries) that are contained in the selection



18.5.2 Version history and Protection Info

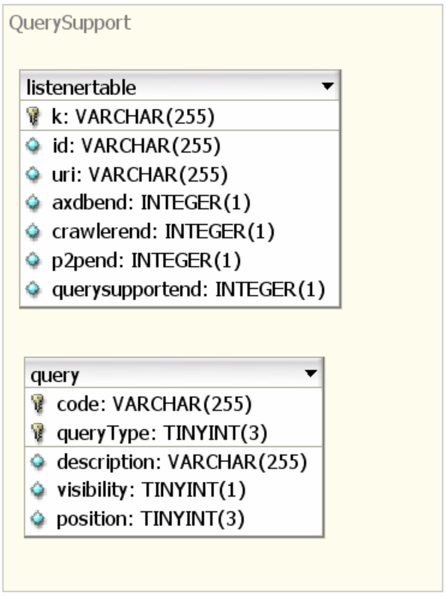
A table that contains all the versions of each object is necessary in order to have immediately available the needed object in its native format. If the contents are stored in the object or in the filesystem by the means of the mechanism described in the versioning section, it is enough to store in the VersionHistory table together with the AXOID, the version, the timestamp and the object related to that version that can be recovered from DID table once a check-in of a new version is performed.



This database, apart from the already discussed DID table, contains a VersionHistory table that store the URI of each version of the AXOID with the information related to the creation of the version.
ProtectionInfo is a table that contains the ProtectionStamp that bound the objects to the different protection model that can be applied to the object; ProtectionInfo field is also reported.

18.5.3 Query Distribution and Integration

In this section the ER diagram that is needed by the Query Support Web Interface for distributing and Integrating query and query results is reported. The description of the table and their meaning is reported in section 11.1.4.

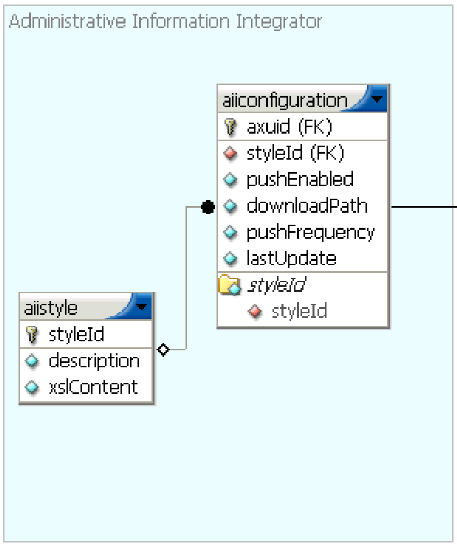


.

18.5.4 Administrative Information Integrator

In this section the ER for the simple database that have to support the Administrative Information Intergrator is drawn.

This database is quite simple since it needs only to store the XSLT styles for transforming the logs to the appropriate format for the CMS, in order to support multiple platforms, the configuration of the user preferencies and a tagle of logs with all download made by users.



Administrative Information Integrator

aiiconfiguration

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axuid	**VARCHAR(60)**	PK	NN				
styleId	BIGINT(20)		NN				
pushEnabled	TINYINT(1)		NN				
downloadPath	VARCHAR(255)		NN				
pushFrequency	BIGINT(20)		NN				
lastUpdate	BIGINT(20)		NN				

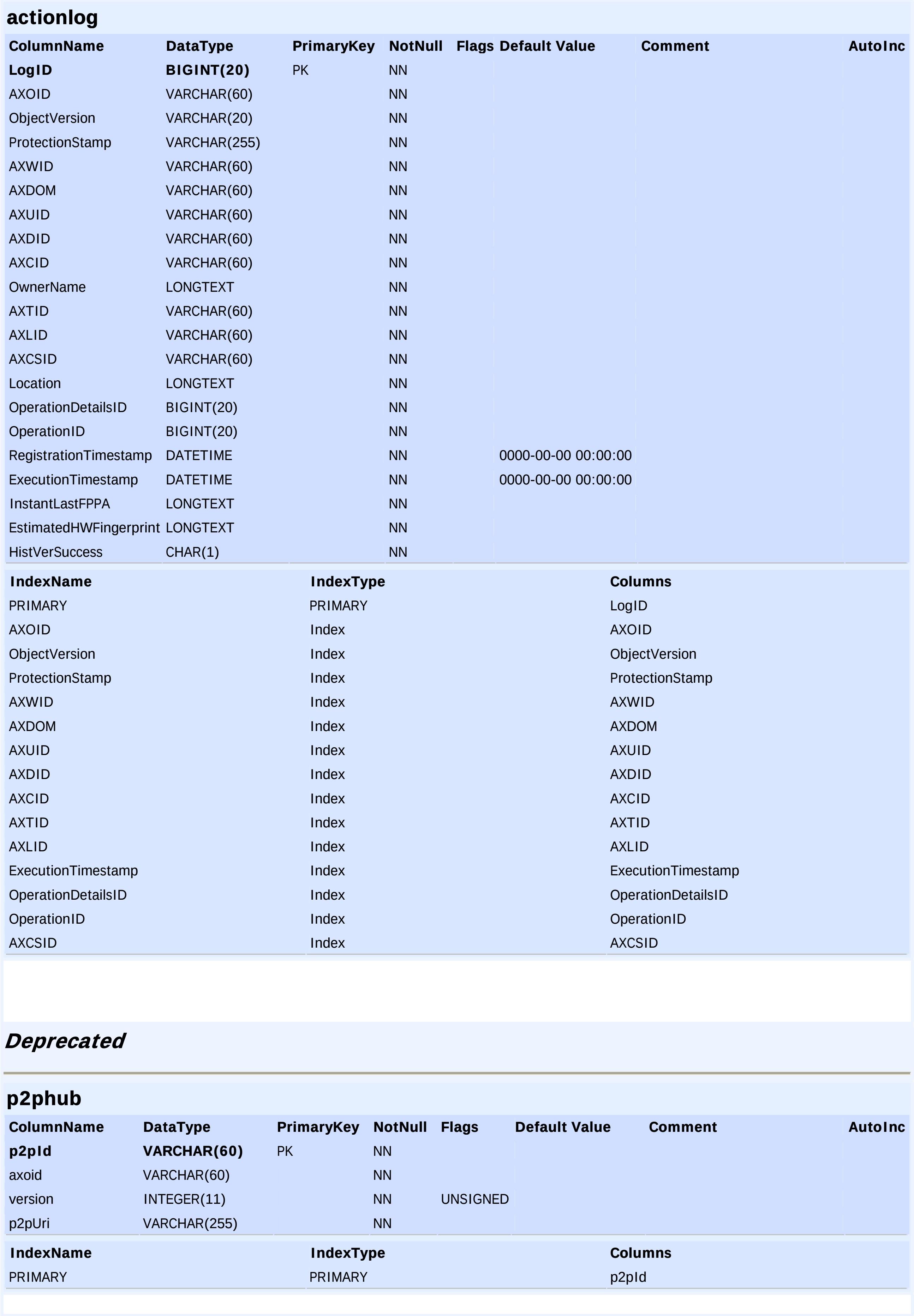
IndexName	IndexType	Columns
PRIMARY	PRIMARY	axuid
styleId	Index	styleId

aiistyle

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
styleId	**BIGINT(20)**	PK	NN				
description	TEXT		NN				
xslContent	TEXT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	styleId

CAMART



actionlog

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
LogID	**BIGINT(20)**	PK	NN				
AXOID	VARCHAR(60)		NN				
ObjectVersion	VARCHAR(20)		NN				
ProtectionStamp	VARCHAR(255)		NN				
AXWID	VARCHAR(60)		NN				
AXDOM	VARCHAR(60)		NN				
AXUID	VARCHAR(60)		NN				
AXDID	VARCHAR(60)		NN				
AXCID	VARCHAR(60)		NN				
OwnerName	LONGTEXT		NN				
AXTID	VARCHAR(60)		NN				
AXLID	VARCHAR(60)		NN				
AXCSID	VARCHAR(60)		NN				
Location	LONGTEXT		NN				
OperationDetailsID	BIGINT(20)		NN				
OperationID	BIGINT(20)		NN				
RegistrationTimestamp	DATETIME		NN		0000-00-00 00:00:00		
ExecutionTimestamp	DATETIME		NN		0000-00-00 00:00:00		
InstantLastFPPA	LONGTEXT		NN				
EstimatedHWFingerprint	LONGTEXT		NN				
HistVerSuccess	CHAR(1)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	LogID
AXOID	Index	AXOID
ObjectVersion	Index	ObjectVersion
ProtectionStamp	Index	ProtectionStamp
AXWID	Index	AXWID
AXDOM	Index	AXDOM
AXUID	Index	AXUID
AXDID	Index	AXDID
AXCID	Index	AXCID
AXTID	Index	AXTID
AXLID	Index	AXLID
ExecutionTimestamp	Index	ExecutionTimestamp
OperationDetailsID	Index	OperationDetailsID
OperationID	Index	OperationID
AXCSID	Index	AXCSID

Deprecated

p2phub

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
p2pId	**VARCHAR(60)**	PK	NN				
axoid	VARCHAR(60)		NN				
version	INTEGER(11)		NN	UNSIGNED			
p2pUri	VARCHAR(255)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	p2pId

FactoryUsers

dbrights

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
RightID	**BIGINT(20)**	PK	NN				
description	TEXT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	RightID

groups

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
GroupID	**VARCHAR(60)**	PK	NN				
name	VARCHAR(60)		NN				
description	TEXT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	GroupID

groupsdbrights

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
GroupID	**VARCHAR(60)**	PK	NN				
RightID	**BIGINT(20)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	GroupID RightID
rightid	Index	RightID

otherusergroup

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
AXUIDPk	**VARCHAR(60)**	PK	NN				
GroupIDPk	**VARCHAR(60)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	AXUIDPk GroupIDPk
GroupIDPk	Index	GroupIDPk

users

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
AXUID	**VARCHAR(60)**	PK	NN				
NickName	VARCHAR(64)		NN				
Passwd	VARCHAR(64)		NN				
email	VARCHAR(64)		NN				
note	TEXT		NN				
GroupIDPk	VARCHAR(60)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	AXUID
Users_FKIndex1	Index	GroupIDPk

usersdbrights

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
AXUIDPk	**VARCHAR(60)**	PK	NN				
rightidpk	**BIGINT(20)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	AXUIDPk rightidpk
RigthIDPk	Index	rightidpk

Metadata

accessmode

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
accessModeId	**INTEGER(11)**	PK	NN				
description	TEXT		NN				

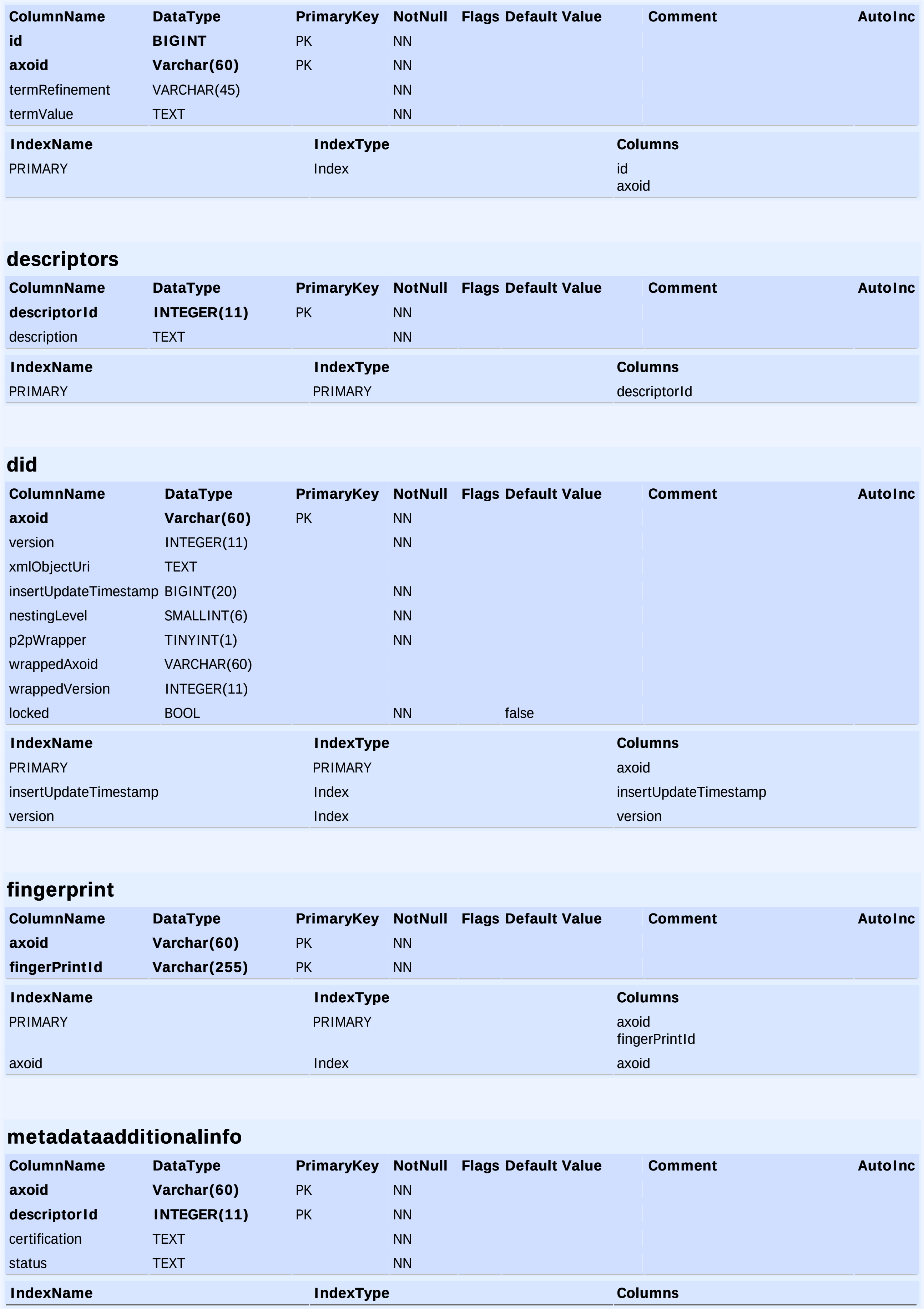
IndexName	IndexType	Columns
PRIMARY	PRIMARY	accessModeId

axinfo

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
accessModeId	**INTEGER(11)**	PK	NN				
creationDate	BIGINT(20)		NN				
lastModificationDate	BIGINT(20)		NN				
objectStatusId	**INTEGER(11)**	PK	NN		1		
basicObject	INTEGER(1)		NN		1		
protectedObject	INTEGER(1)		NN				
governedObject	INTEGER(1)		NN				
workItemId	VARCHAR(60)		NN				
unavailabilityReason	TEXT		NN				
axcid	VARCHAR(60)		NN				
axdid	VARCHAR(60)		NN				
ownerId	VARCHAR(60)		NN				
axlid	VARCHAR(60)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid accessModeId objectStatusId
creationDate	Index	creationDate
lastModificationDate	Index	lastModificationDate
basic	Index	basicObject
axcid	Index	axcid
axinfo_objectstatusid	Index	objectStatusId
axinfo_accessmode	Index	accessModeId
axinfo_axlid	Index	axlid

dcmi



ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
id	**BIGINT**	PK	NN				
axoid	**Varchar(60)**	PK	NN				
termRefinement	VARCHAR(45)		NN				
termValue	TEXT		NN				

IndexName	IndexType	Columns
PRIMARY	Index	id axoid

descriptors

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
descriptorId	**INTEGER(11)**	PK	NN				
description	TEXT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	descriptorId

did

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
version	INTEGER(11)		NN				
xmlObjectUri	TEXT						
insertUpdateTimestamp	BIGINT(20)		NN				
nestingLevel	SMALLINT(6)		NN				
p2pWrapper	TINYINT(1)		NN				
wrappedAxoid	VARCHAR(60)						
wrappedVersion	INTEGER(11)						
locked	BOOL		NN		false		

IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid
insertUpdateTimestamp	Index	insertUpdateTimestamp
version	Index	version

fingerprint

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
fingerPrintId	**Varchar(255)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid fingerPrintId
axoid	Index	axoid

metadataadditionalinfo

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
descriptorId	**INTEGER(11)**	PK	NN				
certification	TEXT		NN				
status	TEXT		NN				

IndexName	IndexType	Columns

PRIMARY	PRIMARY	axoid descriptorId
descriptorId	Index	descriptorId

objectstatus

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
objectStatusId	**INTEGER(11)**	PK	NN				
description	TEXT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	objectStatusId

optionalfield

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
id	**BIGINT**	PK	NN				
axoid	**Varchar(60)**	PK	NN				
fieldName	VARCHAR(45)		NN				
fieldValue	TEXT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	id axoid

translation

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
language	**VARCHAR(3)**	PK	NN				
fieldName	**VARCHAR(255)**	PK	NN				
translation	TEXT		NN				

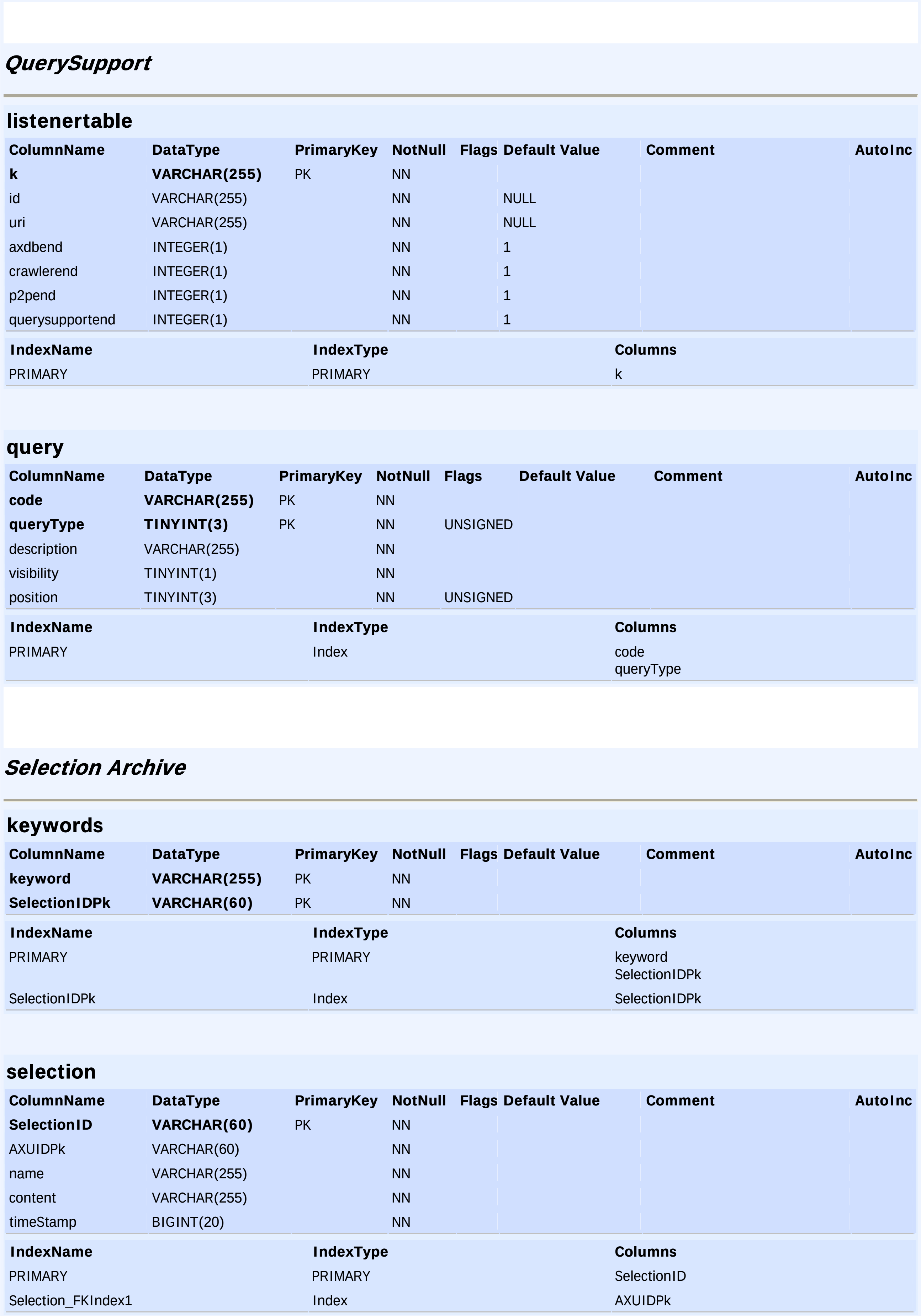
IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid language fieldName
axoid	Index	axoid
fieldname	Index	fieldName

promorof

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
axoidPromoted	**Varchar(60)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid axoidPromoted
axoid	Index	axoid
axoidPromoted	Index	axoidPromoted

P2PSupport



QuerySupport

listenertable

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
k	**VARCHAR(255)**	PK	NN				
id	VARCHAR(255)		NN		NULL		
uri	VARCHAR(255)		NN		NULL		
axdbend	INTEGER(1)		NN		1		
crawlerend	INTEGER(1)		NN		1		
p2pend	INTEGER(1)		NN		1		
querysupportend	INTEGER(1)		NN		1		

IndexName	IndexType	Columns
PRIMARY	PRIMARY	k

query

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
code	**VARCHAR(255)**	PK	NN				
queryType	**TINYINT(3)**	PK	NN	UNSIGNED			
description	VARCHAR(255)		NN				
visibility	TINYINT(1)		NN				
position	TINYINT(3)		NN	UNSIGNED			

IndexName	IndexType	Columns
PRIMARY	Index	code queryType

Selection Archive

keywords

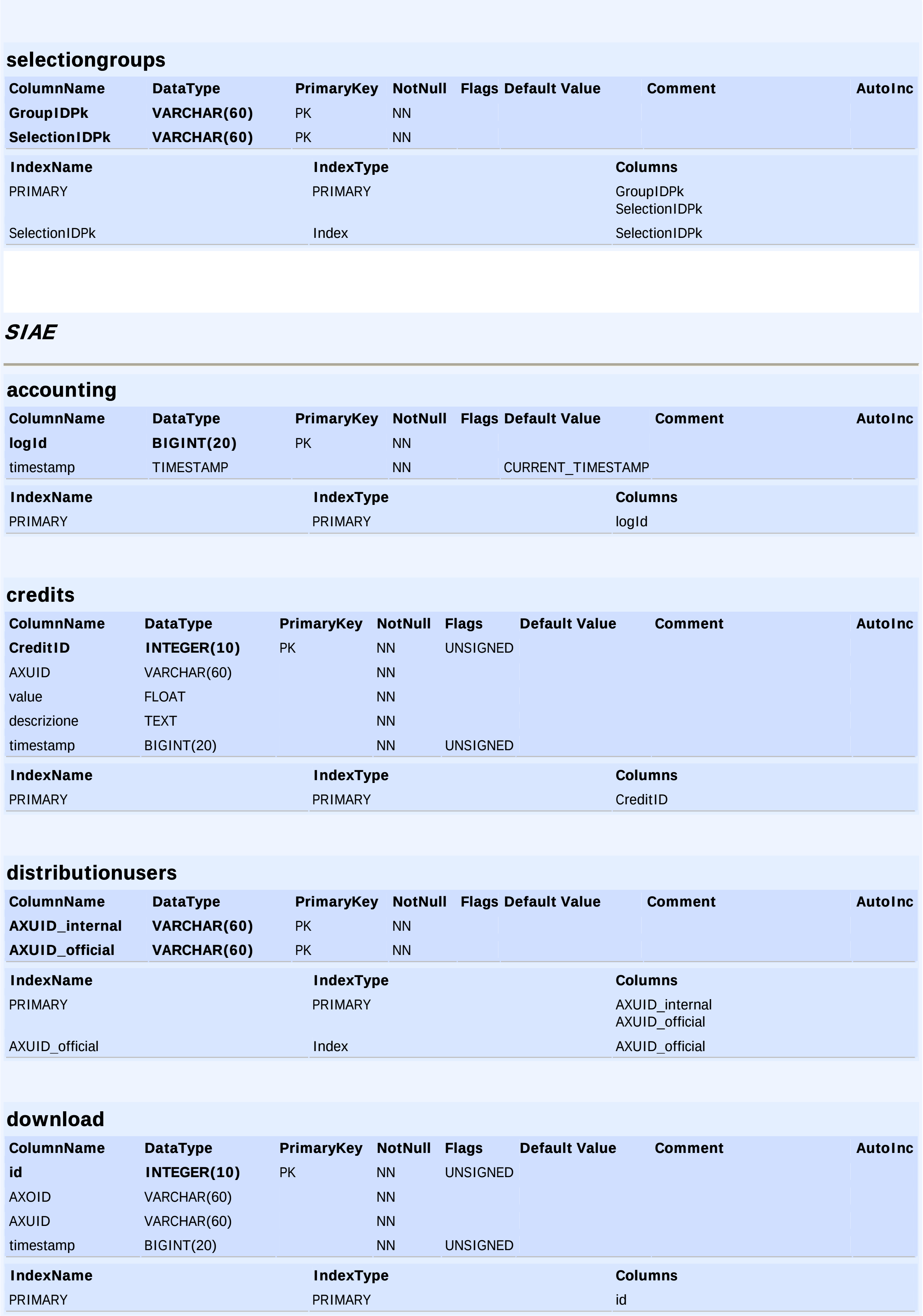
ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
keyword	**VARCHAR(255)**	PK	NN				
SelectionIDPk	**VARCHAR(60)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	keyword SelectionIDPk
SelectionIDPk	Index	SelectionIDPk

selection

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
SelectionID	**VARCHAR(60)**	PK	NN				
AXUIDPk	VARCHAR(60)		NN				
name	VARCHAR(255)		NN				
content	VARCHAR(255)		NN				
timeStamp	BIGINT(20)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	SelectionID
Selection_FKIndex1	Index	AXUIDPk



selectiongroups

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
GroupIDPk	**VARCHAR(60)**	PK	NN				
SelectionIDPk	**VARCHAR(60)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	GroupIDPk SelectionIDPk
SelectionIDPk	Index	SelectionIDPk

SIAE

accounting

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
logId	**BIGINT(20)**	PK	NN				
timestamp	TIMESTAMP		NN		CURRENT_TIMESTAMP		

IndexName	IndexType	Columns
PRIMARY	PRIMARY	logId

credits

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
CreditID	**INTEGER(10)**	PK	NN	UNSIGNED			
AXUID	VARCHAR(60)		NN				
value	FLOAT		NN				
descrizione	TEXT		NN				
timestamp	BIGINT(20)		NN	UNSIGNED			

IndexName	IndexType	Columns
PRIMARY	PRIMARY	CreditID

distributionusers

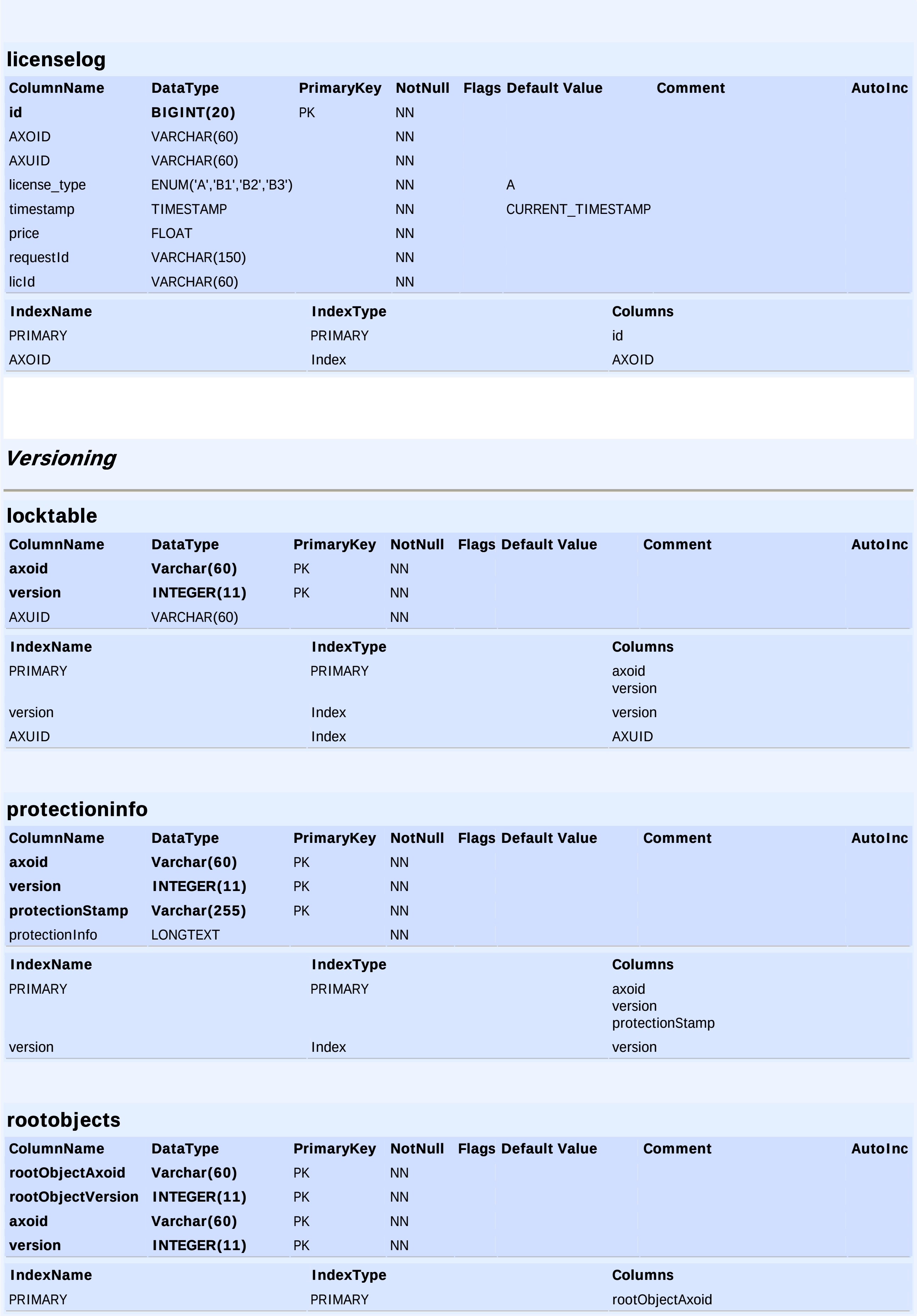
ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
AXUID_internal	**VARCHAR(60)**	PK	NN				
AXUID_official	**VARCHAR(60)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	AXUID_internal AXUID_official
AXUID_official	Index	AXUID_official

download

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
id	**INTEGER(10)**	PK	NN	UNSIGNED			
AXOID	VARCHAR(60)		NN				
AXUID	VARCHAR(60)		NN				
timestamp	BIGINT(20)		NN	UNSIGNED			

IndexName	IndexType	Columns
PRIMARY	PRIMARY	id



licenselog

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
id	**BIGINT(20)**	PK	NN				
AXOID	VARCHAR(60)		NN				
AXUID	VARCHAR(60)		NN				
license_type	ENUM('A','B1','B2','B3')		NN		A		
timestamp	TIMESTAMP		NN		CURRENT_TIMESTAMP		
price	FLOAT		NN				
requestId	VARCHAR(150)		NN				
licId	VARCHAR(60)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	id
AXOID	Index	AXOID

Versioning

locktable

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
version	**INTEGER(11)**	PK	NN				
AXUID	VARCHAR(60)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid version
version	Index	version
AXUID	Index	AXUID

protectioninfo

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
version	**INTEGER(11)**	PK	NN				
protectionStamp	**Varchar(255)**	PK	NN				
protectionInfo	LONGTEXT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid version protectionStamp
version	Index	version

rootobjects

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
rootObjectAxoid	**Varchar(60)**	PK	NN				
rootObjectVersion	**INTEGER(11)**	PK	NN				
axoid	**Varchar(60)**	PK	NN				
version	**INTEGER(11)**	PK	NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	rootObjectAxoid

		rootObjectVersion axoid version
rootObjectAxoid	Index	rootObjectAxoid
rootObjectVersion	Index	rootObjectVersion

versionhistory

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axoid	**Varchar(60)**	PK	NN				
version	**INTEGER(11)**	PK	NN				
xmlObjectUri	TEXT		NN				
when	BIGINT(20)		NN				

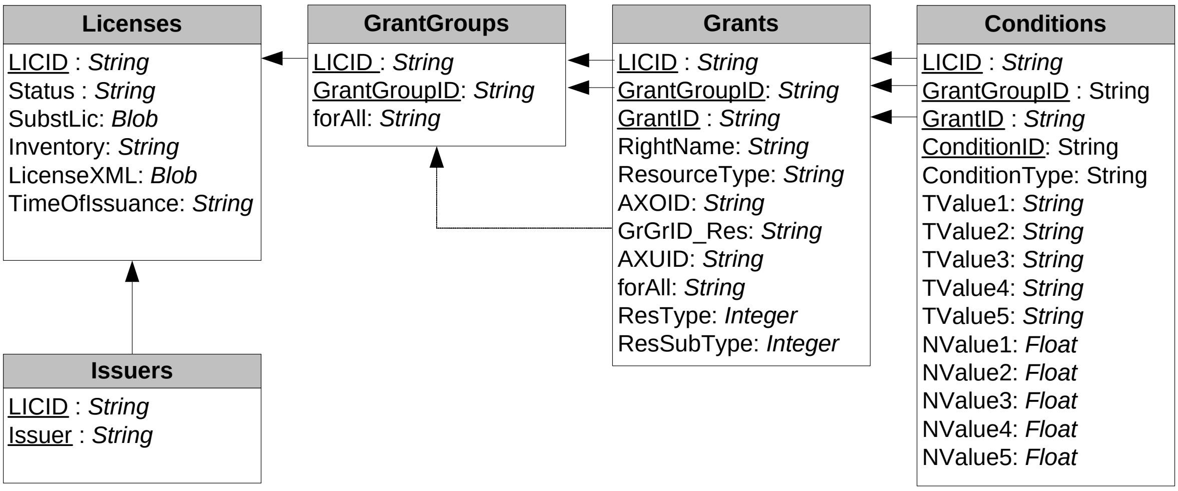
IndexName	IndexType	Columns
PRIMARY	PRIMARY	axoid version
version	Index	version

19 Table description for database PAR and Licenses (UPC)

19.1 ER diagram for Licenses

To represent the content of a license in an Entity-Relationship diagram, we have to focus on the relations with a multiplicity 0..n. These relations show us the number of different tables that we need to store the represented information. The relations with a multiplicity of 1 – 1 can be stored always in the same table.

The next diagram shows how to create the different tables to store the license information. This solution provides the model for storing End-user Licenses, and also for storing Distributor Licenses.



ER diagram for licenses

Licenses

Number of indexes: ?

Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
AXLID	**PK**	**I**	C-Large Length	Not allowed	
Status			C-Large Length	Not allowed	
SubsLic			C-Large Length	Allowed	
Inventory			C-Large Length	Allowed	
TimeofIssuance			C-Large Length	Not allowed	
LicenseXML			C-Blob	Not allowed	

Column details

1. AXLID (PK)

Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String representing the unique identifier for the license

2. Status

Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String that contains the status of the license, possible values are valid or revoked

3. SubsLic

Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String that contains the MPEG-21 REL license that replaces the revoked one, if any

4. Inventory

Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String that contains the variables defined in the license, that can be referenced throught this license

5. TimeofIssuance

Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String that represents the specific date and time at which the license has been issued

6. LicenseXML

Physical data type: BLOB
Allow NULLs: Not allowed
Notes: contains the XML MPEG-21 REL license

Issuers

Number of indexes: ?
Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
AXLID	**PK**	**I**	C-Large Length	Not allowed	
AXUID	**PK**	**I**	C-Large Length	Not allowed	

Column details

1. AXLID (PK)

Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String representing the unique identifier for the license

2. AXUID (PK)

Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String thar represents the unique identifier of the AXMEDIS user that has issued the license

GrantGroups

Number of indexes: ?
Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
AXLID	**PK**	**I**	C-Large Length	Not allowed	
GrantGroupID	**PK**	**I**	C-Large Length	Not allowed	
forAll			C-Large Length	Allowed	

Column details

1. AXLID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String representing the unique identifier for the license

2. GrantGroupID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String containing the unique identifier of the grantGroup

3. forAll
Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String that contains variables whose scope is this entire grantGroup uniquely identified by the GrantGroupID

Grants

Number of indexes: ?
Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
AXLID	**PK**	**I**	C-Large Length	Not allowed	
GrantGroupID	**PK**	**I**	C-Large Length	Not allowed	
GrantID	**PK**	**I**	C-Large Length	Not allowed	
Right			C-Large Length	Not allowed	
ResourceType			C-Large Length	Not allowed	
AXOID		**I**	C-Large Length	Allowed	
GrGrID_Res	**FK**	**I**	C-Large Length	Allowed	
AXUID			C-Large Length	Not allowed	
forAll			C-Large Length	Allowed	
ResType			Integer	Not allowed	
ResSubType			Integer	Not allowed	

Column details

1. AXLID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String representing the unique identifier for the license

2. GrantGroupID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String containing the unique identifier of the grantGroup

3. GrantID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String containing the unique identifier of the grant

4. Right
Physical data type: LONGTEXT

Allow NULLs:	Not allowed
Notes:	String that specifies the right granted

5. ResourceType

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String that specifies the type of object against which the principal of this grant has the right to perform an action. If the resouceType is Resource, then the object against which the AXMEDIS user can exercise the right is an AXMEDIS object, and if the resourceType is GrantGroup then the object is a grant or granGroup, typically for distribution licenses

6. AXOID

Physical data type:	LONGTEXT
Allow NULLs:	Allowed
Notes:	String containing ths unique identifier of the AXMEDIS object

7. GrGrID_Res (FK)

Physical data type:	LONGTEXT
Allow NULLs:	Allowed
Notes:	String containing ths unique identifier of the grantGroup that can be issued

8. AXUID

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String identifying the AXMEDIS user to whom this grant conveys rights

9. forAll

Physical data type:	LONGTEXT
Allow NULLs:	Allowed
Notes:	String that contains variables whose scope is the entire grant uniquely identified by the GrantID

10. ResType

Physical data type:	INTEGER
Allow NULLs:	Not Allowed
Notes:	If ResourceType is “Resource” this field sets the type of the “reference” to the resource found in AXOID. 0 → Digital Item Item, 1→ Digital Item Reference

11. ResSubType

Physical data type:	INTEGER
Allow NULLs:	Not Allowed
Notes:	If ResType is 0 (Digital Item Item) this field sets the type of the reference. 0(id), 1(uri), 2(type)

Conditions

Number of indexes:	?
Number of foreign keys:	?

Columns		idx	Data type	Allow NULLs	Value/Range
AXLID	**PK**	**I**	C-Large Length	Not allowed	
GrantGroupID	**PK**	**I**	C-Large Length	Not allowed	
GrantID	**PK**	**I**	C-Large Length	Not allowed	
ConditionID	**PK**	**I**	C-Large Length	Not allowed	
ConditionType		**I**	C-Large Length	Not allowed	
TValue1			C-Large Length	Allowed	
TValue2			C-Large Length	Allowed	
TValue3			C-Large Length	Allowed	
TValue4			C-Large Length	Allowed	
TValue5			C-Large Length	Allowed	
NValue1			C-Float	Allowed	
NValue2			C-Float	Allowed	
NValue3			C-Float	Allowed	
NValue4			C-Float	Allowed	
NValue5			C-Float	Allowed	

Column details

1. AXLID (PK)

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String representing the unique identifier for the license

2. GrantGroupID (PK)

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String containing the unique identifier of the grantGroup

3. GrantID (PK)

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String containing the unique identifier of the grant

4. ConditionID

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String containing the unique identifier of the condition

5. ConditionType

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String representing the type of the condition. This field can have the values specificied in the ConditionType column of the Condition Table. For example, this field can take the values territory or validityInterval

6. TValue(1-5)

Physical data type:	LONGTEXT
Allow NULLs:	Allowed
Notes:	String that contains information related to the condition according to the ConditionType as defined in the Condition Table. For example, if the ConditionType is validityInterval, the TValue1 contains a String that represents the date at which the interval of time defined by this condition begins and the Tvalue2 contains a String that represents the date at which the interval of time defined by this condition ends

7. Nvalue(1-5)

Physical data type:	FLOAT
Allow NULLs:	Allowed
Notes:	Numeric value that contains information related to the condition according to the ConditionType as defined in the Condition Table. For example, if the ConditionType is exerciseLimit, the NValue1 represents the limit on the number of times that certain exercises may occur

The relation between Tables and Classes is:

Table (ER)	Classes (UML) stored in the table
Licenses	License
Issuers	Issuer
GrantGroups	GrantGroup
Grants	Grant, Right, Resource, Principal
Conditions	Condition (all types)

To represent all type of conditions, we have decided to store the data in one unique table with a set of “standard” fields. Each field of this table corresponds to an attribute of the condition depending on the condition type.

We provide a table where we describe the mapping between the standard fields of the table (ER) and the condition attributes (UML).

In this model is very easy to add new types of conditions to the system without causing the reimplementation of a lot of modules. And, moreover, it makes easier and much more efficient the search of the information needed in the authorisation model.

19.1.1 Information inside the Conditions table

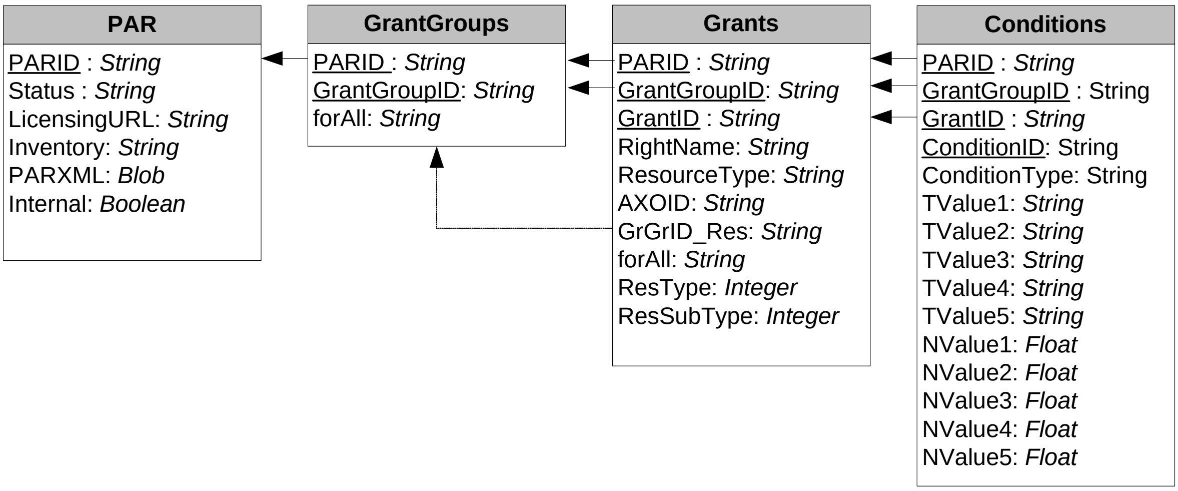
ConditionType	TValue1	TValue2	TValue3	TValue4	TValue5	NValue1	NValue2
territory	location (country, region, state, city, postalCode, street)	domain (uri)					
validityInterval	notBefore	notAfter					
validityIntervalFloating	serviceReference (*1) (serviceDescription (anonimousStateService, uddi, wsdlAddress, wsdlComplete), serviceParameters)	duration					
validityIntervalDurationPattern		duration					
validityIntervalStartsNow	validityInterval - notBefore	validityInterval - notAfter	backwardTolerance	forwardTolerance			
validityTimeMetered	serviceReference (*1)	duration	quantum				
validityTimePeriodic	Start	duration	phase	period		periodCount	
exerciseLimit	serviceReference (*1)					count	
feeFlat	serviceReference (*1)	rate - currency	to (*2) (paymentService (serviceReference), aba (institution, account), any)			rate - amount	
feeMetered	phase	rate - currency	to (*2)	per	by	rate - amount	
feePerInterval	serviceReference (*1)	rate - currency	to (*2)	per		rate - amount	
feePerUse		rate - currency	to (*2)			rate - amount	
feePerUsePrePay	serviceReference (*1)	rate - currency	to (*2)			rate - amount	initialNumberOfUses

(*1) Same information as the first *serviceReference* field defined
(*2) Same information as the first *to* field defined

19.2 ER diagram for PAR

To store this content in a relational database we need 4 tables. The model is very similar to the License model, but we have substracted the data corresponding to the Issuer and Principal.

With this diagram, we do not pretend to show where or how are stored the resources or the RDD rights. Depending on the implementation tables can be related to that diagram accordingly.



ER diagram for PAR

PAR

Number of indexes: ?
Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
PARID	**PK**	**I**	C-Large Length	Not allowed	
Status			C-Large Length	Not allowed	
LicensingURL			C-Large Length	Allowed	
Inventory			C-Large Length	Allowed	
PARXML			C-Blob	Not allowed	
Internal			C-Boolean	Not allowed	

Column details

1. PARID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String representing the unique identifier for the PAR

2. Status
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String that contains the status of the PAR, e.g. verified

3. LicensingURL
Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String that contains the URL to acquire a license for the object

4. Inventory
Physical data type: LONGTEXT
Allow NULLs: Allowed

Notes: String that contains the variables defined in the r:license element of the PAR, that can be referenced throught this license

5. PARXML
Physical data type: BLOB
Allow NULLs: Not allowed
Notes: contains the XML MPEG-21 REL license of the PAR

6. Internal
Physical data type: BOOLEAN
Allow NULLs: Not allowed
Notes: Boolean identifying if the PAR is an internal PAR if set a true

GrantGroups

Number of indexes: ?
Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
PARID	**PK**	**I**	C-Large Length	Not allowed	
GrantGroupID	**PK**	**I**	C-Large Length	Not allowed	
forAll			C-Large Length	Allowed	

Column details

1. PARLID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String representing the unique identifier for the PAR

2. GrantGroupID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String containing the unique identifier of the grantGroup

3. forAll
Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String that contains variables whose scope is this entire grantGroup uniquely identified by the GrantGroupID

Grants

Number of indexes: ?
Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
PARID	**PK**	**I**	C-Large Length	Not allowed	
GrantGroupID	**PK**	**I**	C-Large Length	Not allowed	
GrantID	**PK**	**I**	C-Large Length	Not allowed	
RightName			C-Large Length	Not allowed	
ResourceType			C-Large Length	Not allowed	
AXOID		**I**	C-Large Length	Allowed	
GrGrID_Res	**FK**	**I**	C-Large Length	Allowed	
forAll			C-Large Length	Allowed	
ResType			Integer	Not allowed	
ResSubType			Integer	Not allowed	

Column details

1. PARID (PK)
Physical data type: LONGTEXT

Allow NULLs: Not allowed
Notes: String representing the unique identifier for the PAR

2. GrantGroupID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String containing the unique identifier of the grantGroup

3. GrantID (PK)
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String containing the unique identifier of the grant

4. RightName
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String that specifies the right granted

5. ResourceType
Physical data type: LONGTEXT
Allow NULLs: Not allowed
Notes: String that specifies the type of object against which the principal of this grant has the right to perform an action. If the resouceType is Resource, then the object against which the AXMEDIS user can exercise the right is an AXMEDIS object, and if the resourceType is GrantGroup then the object is a grant or granGroup, typically for distriubution licenses

6. AXOID
Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String containing ths unique identifier of the AXMEDIS object

7. GrGrID_Res (FK)
Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String containing ths unique identifier of the grantGroup that can be issued

8. forAll
Physical data type: LONGTEXT
Allow NULLs: Allowed
Notes: String that contains variables whose scope is the entire grant uniquely identified by the GrantID

9. ResType
Physical data type: INTEGER
Allow NULLs: Not Allowed
Notes: If ResourceType is “Resource” this field sets the type of the “reference” to the resource found in AXOID. 0 → Digital Item Item, 1→ Digital Item Reference

10. ResSubType
Physical data type: INTEGER
Allow NULLs: Not Allowed
Notes: If ResType is 0 (Digital Item Item) this field sets the type of the reference. 0(id), 1(uri), 2(type)

Conditions

Number of indexes: ?
Number of foreign keys: ?

Columns		idx	Data type	Allow NULLs	Value/Range
PARID	**PK**	**I**	C-Large Length	Not allowed	
GrantGroupID	**PK**	**I**	C-Large Length	Not allowed	
GrantID	**PK**	**I**	C-Large Length	Not allowed	
ConditionID	**PK**	**I**	C-Large Length	Not allowed	

ConditionType	**I**	C-Large Length	Not allowed
TValue1		C-Large Length	Allowed
TValue2		C-Large Length	Allowed
TValue3		C-Large Length	Allowed
TValue4		C-Large Length	Allowed
TValue5		C-Large Length	Allowed
NValue1		C-Float	Allowed
NValue2		C-Float	Allowed
NValue3		C-Float	Allowed
NValue4		C-Float	Allowed
NValue5		C-Float	Allowed

Column details

1. AXLID (PK)

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String representing the unique identifier for the PAR

2. GrantGroupID (PK)

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String containing the unique identifier of the grantGroup

3. GrantID (PK)

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String containing the unique identifier of the grant

4. ConditionID

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String containing the unique identifier of the condition

5. ConditionType

Physical data type:	LONGTEXT
Allow NULLs:	Not allowed
Notes:	String representing the type of the condition. This field can have the values specificied in the ConditionType column of the Condition Table. For example, this field can take the values territory or validityInterval

6. TValue(1-5)

Physical data type:	LONGTEXT
Allow NULLs:	Allowed
Notes:	String that contains information related to the condition according to the ConditionType as defined in the Condition Table. For example, if the ConditionType is validityInterval, the TValue1 contains a String that represents the date at which the interval of time defined by this condition begins and the Tvalue2 contains a String that represents the date at which the interval of time defined by this condition ends

7. Nvalue(1-5)

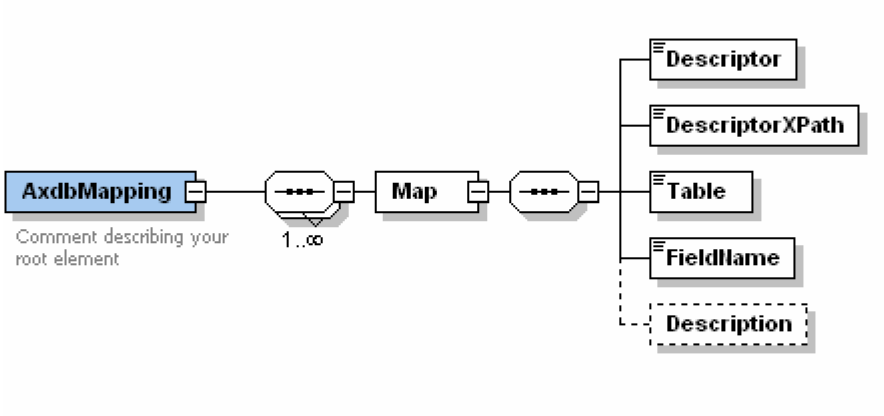
Physical data type:	FLOAT
Allow NULLs:	Allowed
Notes:	Numeric value that contains information related to the condition according to the ConditionType as defined in the Condition Table. For example, if the ConditionType is exerciseLimit, the NValue1 represents the limit on the number of times that certain exercises may occur

The relation between Tables and Classes is:

Table (ER)	Classes (UML) stored in the table
PAR	PAR
GrantGroups	GrantGroup
Grants	Grant, Right, Resource
Conditions	Condition (all types)

To represent all type of conditions, we have decided to store the data in one unique table with a set of “standard” fields. Each field of this table corresponds to an attribute of the condition depending on the condition type.

20 Formal description of format AXDBMapping (EXITECH)



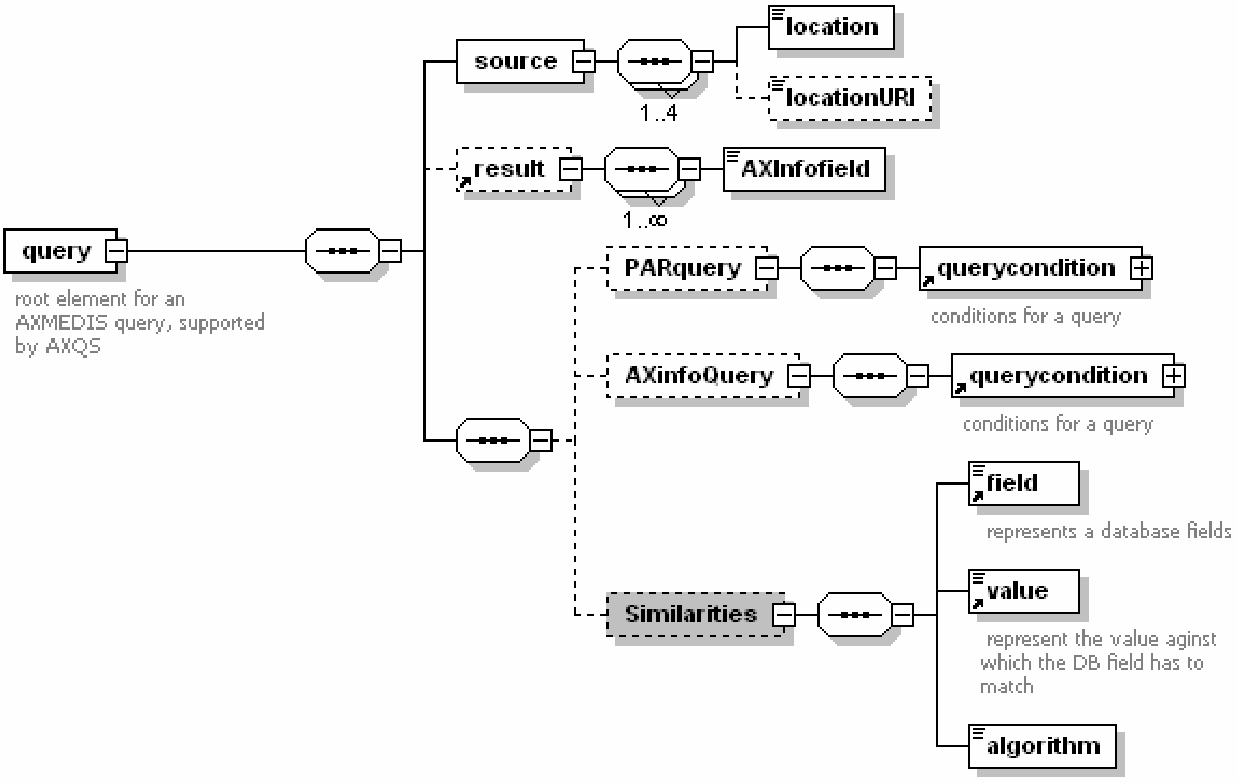
The textual version of the schema follows:

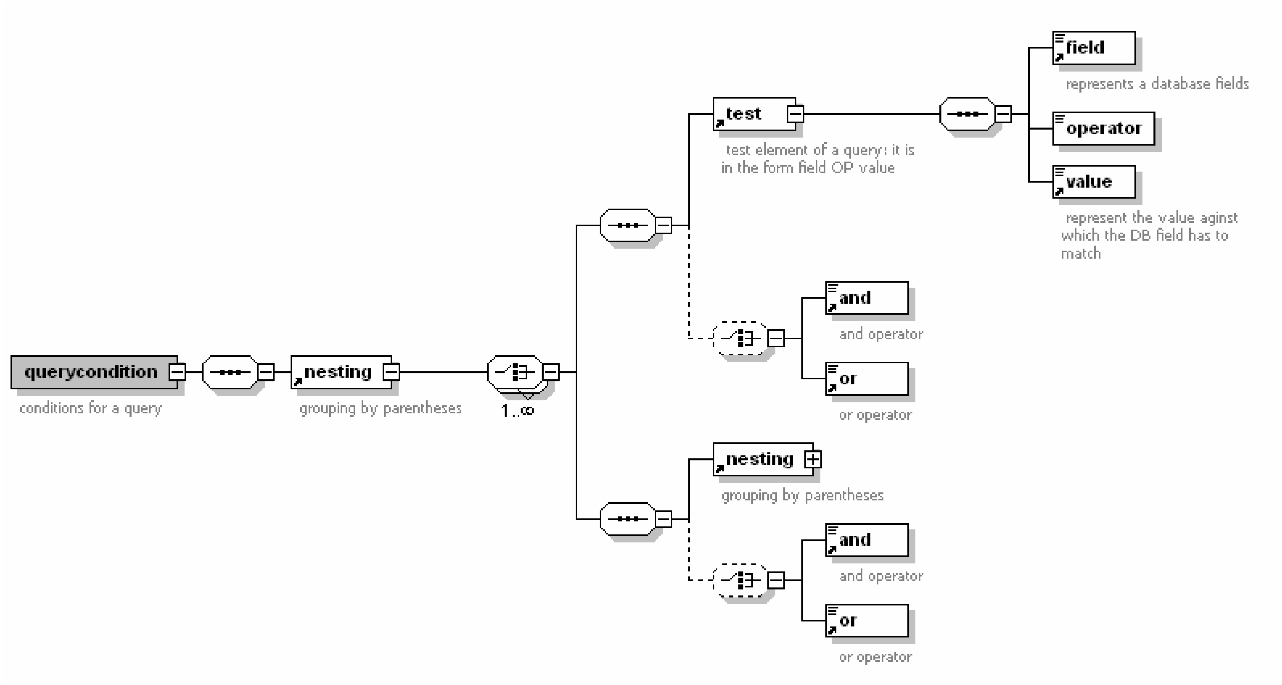
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
        <!-- version 1.3 (31/03/2006) -->
        <xs:element name="AxdbMapping">
                <xs:annotation>
                        <xs:documentation>Comment describing your root element</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                        <xs:sequence maxOccurs="unbounded">
                                <xs:element name="Map">
                                        <xs:complexType>
                                                <xs:sequence>
                                                        <xs:element name="Descriptor" type="xs:string"/>
                                                        <xs:element name="DescriptorXPath"
type="xs:string"/>
                                                        <xs:element name="Table" type="xs:string"/>
                                                        <xs:element name="FieldName" type="xs:string"/>
                                                        <xs:element name="Description" minOccurs="0"/>
                                                </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                        </xs:sequence>
                </xs:complexType>
        </xs:element>
</xs:schema>
```

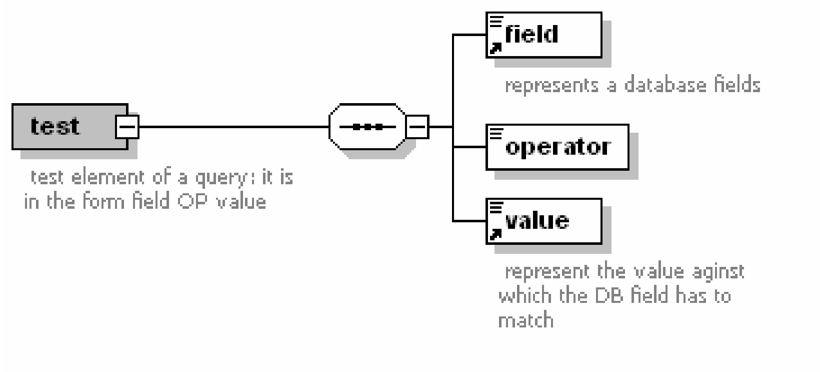
21 Formal description of format AXMEDIS Query (EXITECH)

```
<xs:element name="query">
        <xs:annotation>
            <xs:documentation>root element for an AXMEDIS query, supported by AXQS</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="source">
                    <xs:complexType>
                        <xs:sequence maxOccurs="4">
                            <xs:element name="location">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:enumeration
value="CRAWLER"/>

                                        <xs:enumeration
value="AXEPTOOL"/>

                                        <xs:enumeration
value="AXDB"/>

                                        <xs:enumeration value="QS"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="locationURI" type="xs:anyURI"
minOccurs="0"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element ref="result" minOccurs="0"/>
                <xs:sequence>
                    <xs:element name="PARquery" minOccurs="0">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="querycondition"/>
                            </xs:sequence>
                            <xs:attribute name="InternalPAR" type="xs:boolean"
use="optional"/>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="AXinfoQuery" minOccurs="0">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="querycondition"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="Similarities" minOccurs="0">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="field"/>
                                <xs:element ref="value"/>
                                <xs:element name="algorithm">
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:enumeration
value="REGULAREXPRESSION"/>

                                            <xs:enumeration
value=""/>

                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
```

```
                                </xs:sequence>
                                <xs:attribute name="name" type="xs:string" use="optional" default="QUERY"/>
                        </xs:complexType>
                </xs:element>
```


source
location
locationURI
1..4
result
AXInfofield
1..∞
query
root element for an AXMEDIS query, supported by AXQS
PARquery
querycondition
conditions for a query
AXinfoQuery
querycondition
conditions for a query
field
represents a database fields
Similarities
value
represent the value aginst which the DB field has to match
algorithm


querycondition
conditions for a query
nesting
grouping by parentheses
1..∞
test
test element of a query: it is in the form field OP value
field
represents a database fields
operator
value
represent the value aginst which the DB field has to match
and
and operator
or
or operator
nesting
grouping by parentheses
and
and operator
or
or operator


test
test element of a query: it is in the form field OP value
field
represents a database fields
operator
value
represent the value aginst which the DB field has to match

22 Formal description of format AXMEDIS Query Result (EXITECH)

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 sp1 U (http://www.xmlspy.com) by Fabrizio Fioravanti (Exitech) -
->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="queryresults">
        <xs:annotation>
            <xs:documentation>Version 1.7</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence minOccurs="0">
                <xs:element name="errorfields" minOccurs="0" maxOccurs="3">
                    <xs:complexType>
                        <xs:sequence maxOccurs="unbounded">
                            <xs:element name="fieldname"
type="xs:string"/>
                        </xs:sequence>
                        <xs:attribute name="source" use="required">
                            <xs:simpleType>
                                <xs:restriction base="xs:string">
                                    <xs:enumeration
value="AXEPTOOL"/>
                                    <xs:enumeration
value="AXDB"/>
                                    <xs:enumeration
value="CRAWLER"/>
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:attribute>
                    </xs:complexType>
                </xs:element>
                <xs:sequence minOccurs="0" maxOccurs="unbounded">
                    <xs:element name="AXObject">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:choice>
                                    <xs:element
name="AXEPTOOL">
                                        <xs:complexType>
                                            <xs:sequence>

    <xs:element name="peers">

    <xs:complexType>

    <xs:sequence maxOccurs="unbounded">

        <xs:element name="peerID" type="xs:string"/>
```

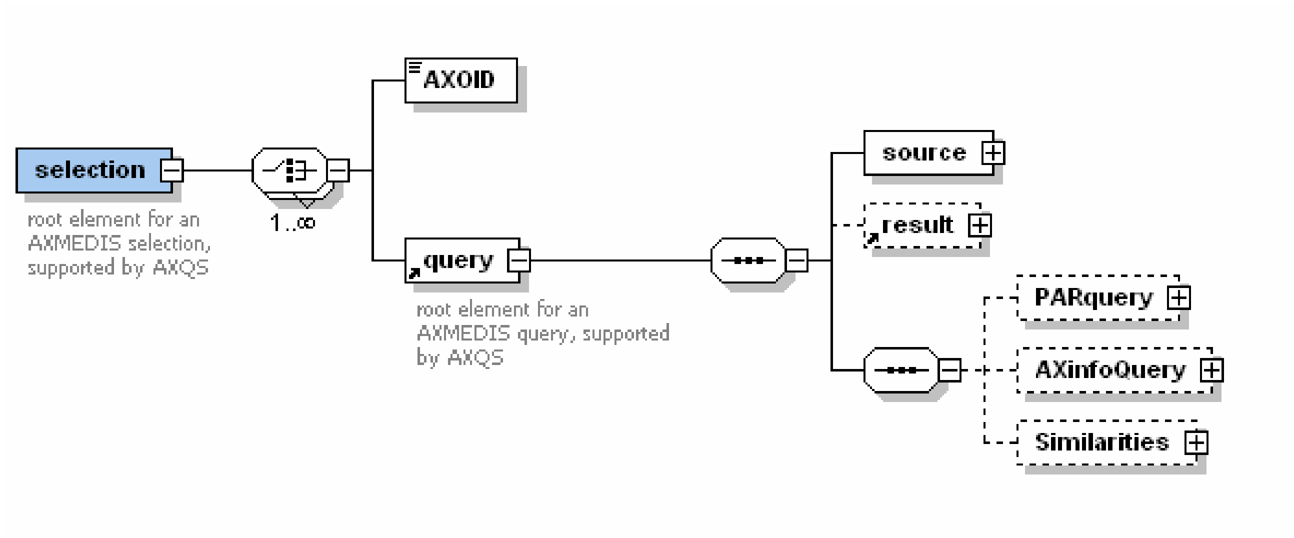
```
          </xs:sequence>

          </xs:complexType>

          </xs:element>

          <xs:element name="AXOID" type="xs:string"/>
                                                                                      </xs:sequence>
                                                                              </xs:complexType>
                                                                      </xs:element>
                                                                      <xs:element
name="CRAWLER">
                                                                              <xs:complexType>
                                                                                      <xs:sequence>

          <xs:element name="TAXOID" type="xs:string"/>
                                                                                      </xs:sequence>
                                                                              </xs:complexType>
                                                                      </xs:element>
                                                                      <xs:element name="AXDB">
                                                                              <xs:complexType>
                                                                                      <xs:sequence>

          <xs:element name="AXOID" type="xs:string"/>
                                                                                      </xs:sequence>
                                                                              </xs:complexType>
                                                                      </xs:element>
                                                                      <xs:element name="QS">
                                                                              <xs:complexType>
                                                                                      <xs:sequence>

          <xs:element name="AXOID" type="xs:string"/>
                                                                                      </xs:sequence>
                                                                              </xs:complexType>
                                                                      </xs:element>
                                                              </xs:choice>
                                                              <xs:element name="extrainfo"
minOccurs="0" maxOccurs="unbounded">
                                                                      <xs:complexType>
                                                                              <xs:sequence>
                                                                                      <xs:element
name="field" type="xs:string"/>
                                                                                      <xs:element
name="value" type="xs:string"/>
                                                                              </xs:sequence>
                                                                      </xs:complexType>
                                                              </xs:element>
                                                              <xs:element name="rootobjects"
minOccurs="0">
```

```
                                    <xs:complexType>
                                        <xs:sequence
maxOccurs="unbounded">
                                            <xs:element
name="rootobject" type="xs:string"/>
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
            </xs:sequence>
            <xs:attribute name="end" type="xs:boolean" use="required"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

23 Formal description of format AXMEDIS Selection (EXITECH)

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 sp1 U (http://www.xmlspy.com) by Fabrizio Fioravanti (Exitech) -
->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="nesting">
        <xs:annotation>
            <xs:documentation>Version 1.7</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:choice maxOccurs="unbounded">
                <xs:sequence>
                    <xs:element ref="test"/>
                    <xs:choice minOccurs="0">
                        <xs:element ref="and"/>
                        <xs:element ref="or"/>
                    </xs:choice>
                </xs:sequence>
                <xs:sequence>
                    <xs:element ref="nesting"/>
                    <xs:choice minOccurs="0">
                        <xs:element ref="and"/>
                        <xs:element ref="or"/>
                    </xs:choice>
                </xs:sequence>
            </xs:choice>
        </xs:complexType>
    </xs:element>
    <xs:element name="result">
        <xs:complexType>
            <xs:sequence maxOccurs="unbounded">
                <xs:element name="AXInfofield" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="query">
        <xs:annotation>
            <xs:documentation>root element for an AXMEDIS query, supported by
AXQS</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="source">
                    <xs:complexType>
                        <xs:sequence maxOccurs="4">
                            <xs:element name="location">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
```

```
                                        <xs:enumeration
value="CRAWLER"/>
                                        <xs:enumeration
value="AXEPTOOL"/>
                                        <xs:enumeration
value="AXDB"/>
                                        <xs:enumeration
value="QS"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="locationURI"
type="xs:anyURI" minOccurs="0"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element ref="result" minOccurs="0"/>
                <xs:sequence>
                    <xs:element name="PARquery" minOccurs="0">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="querycondition"/>
                            </xs:sequence>
                            <xs:attribute name="InternalPAR"
type="xs:boolean" use="optional"/>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="AXinfoQuery" minOccurs="0">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="querycondition"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="Similarities" minOccurs="0">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element ref="field"/>
                                <xs:element ref="value"/>
                                <xs:element name="algorithm">
                                    <xs:simpleType>
                                        <xs:restriction
base="xs:string">
                                            <xs:enumeration
value="REGULAREXPRESSION"/>
                                            <xs:enumeration
value=""/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
            </xs:sequence>
            <xs:attribute name="name" type="xs:string" use="optional"
default="QUERY"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="field" type="xs:string">
        <xs:annotation>
            <xs:documentation> represents a database fields</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="value" type="xs:anySimpleType">
        <xs:annotation>
            <xs:documentation> represent the value aginst which the DB field has to
match</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="test">
        <xs:annotation>
            <xs:documentation> test element of a query: it is in the form field OP
value</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="field"/>
                <xs:element name="operator">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:enumeration value="GT"/>
                            <xs:enumeration value="LT"/>
                            <xs:enumeration value="EQ"/>
                            <xs:enumeration value="GE"/>
                            <xs:enumeration value="LE"/>
                            <xs:enumeration value="NE"/>
                            <xs:enumeration value="STARTWITH"/>
                            <xs:enumeration value="ENDWITH"/>
                            <xs:enumeration value="CONTAINS"/>
                            <xs:enumeration value=""/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
                <xs:element ref="value"/>
            </xs:sequence>
            <xs:attribute name="NOT" type="xs:boolean" use="optional"
default="false"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="querycondition">
        <xs:annotation>
            <xs:documentation>conditions for a query</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="nesting"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="and" type="xs:token">
        <xs:annotation>
            <xs:documentation> and operator</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="or" type="xs:token">
        <xs:annotation>
            <xs:documentation> or operator</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:element name="selection">
        <xs:annotation>
            <xs:documentation>root element for an AXMEDIS selection, supported by
AXQS</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:choice maxOccurs="unbounded">
                <xs:element name="AXOID" type="xs:string"/>
                <xs:element ref="query"/>
            </xs:choice>
            <xs:attribute name="name" type="xs:string" use="required"/>
            <xs:attribute name="timestamp" type="xs:dateTime" use="required"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

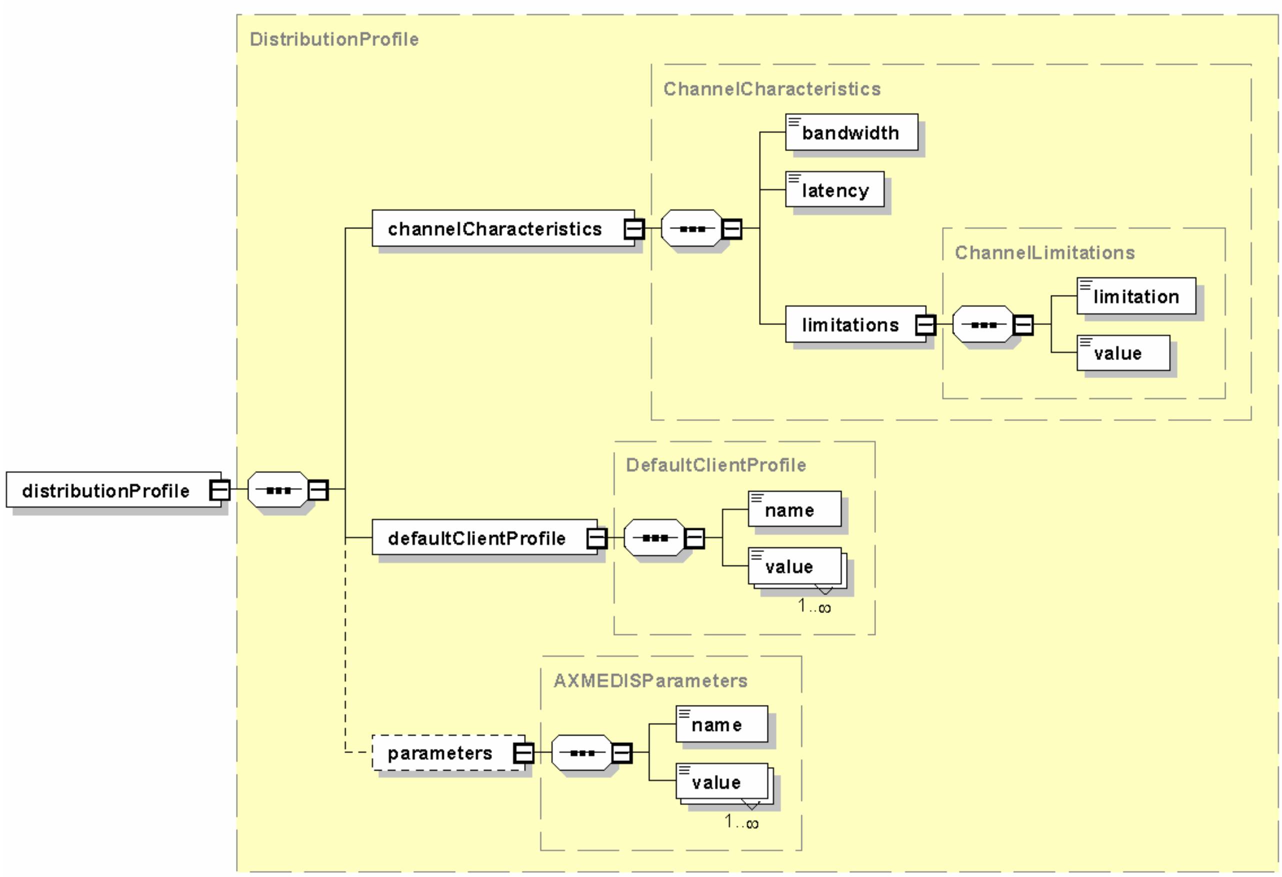

selection
root element for an AXMEDIS selection, supported by AXQS
1..∞
AXOID
query
root element for an AXMEDIS query, supported by AXQS
source
result
PARquery
AXinfoQuery
Similarities

24 Formal description of format Simple Query (FHGIGD)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="SimpleQuery">
        <xs:annotation>
            <xs:documentation>Simple query language</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence maxOccurs="unbounded">
                <xs:element name="FieldNameValuePair">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="FieldName">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:enumeration value="Title"/>
                                        <xs:enumeration value="Artist"/>
                                        <xs:enumeration value="Subject"/>
                                        <xs:enumeration value="Keywords"/>
                                        <xs:enumeration value="Description"/>
                                        <xs:enumeration value="Media Type"/>
                                        <xs:enumeration value="Year"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="Operator">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:enumeration value="GT"/>
                                        <xs:enumeration value="LT"/>
                                        <xs:enumeration value="EQ"/>
                                        <xs:enumeration value="GE"/>
                                        <xs:enumeration value="LE"/>
                                        <xs:enumeration value="NE"/>
                                        <xs:enumeration value="STARTWITH"/>
                                        <xs:enumeration value="ENDWITH"/>
                                        <xs:enumeration value="CONTAINS"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>
                            <xs:element name="Value" type="xs:string"/>
                        </xs:sequence>
                        <xs:attribute name="NOT" type="xs:boolean" use="optional" default="false"/>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="combination" type="andOr" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:simpleType name="andOr">
        <xs:restriction base="xs:string">
            <xs:enumeration value="AND"/>
            <xs:enumeration value="OR"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

25 Formal description of format Distribution Profile (FHGIGD)

```
<?xml version="1.0" encoding="UTF-8"?>
<!--definition for AXMEDIS distribution profile -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <!-- definition of simple type elements -->
    <xs:simpleType name="ContentType">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <!-- definition of complex type elements -->

    <!-- channel limitations -->
    <xs:complexType name="ChannelLimitations">
        <xs:sequence>
            <xs:element name="limitation" type="xs:string"/>
            <xs:element name="value" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>

    <!-- channel characteristics -->
    <xs:complexType name="ChannelCharacteristics">
        <xs:sequence>
            <xs:element name="bandwidth" type="xs:string"/>
            <xs:element name="latency" type="xs:string"/>
            <xs:element name="limitations" type="ChannelLimitations"/>
        </xs:sequence>
    </xs:complexType>

    <!-- default client profile -->
    <xs:complexType name="DefaultClientProfile">
        <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="value" type="xs:string" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <!-- AXMEDIS Parameter -->
    <xs:complexType name="AXMEDISParameters">
        <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="value" type="xs:string" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <!-- definition of the client profile -->
    <xs:complexType name="DistributionProfile">
        <xs:sequence>
            <xs:element name="channelCharacteristics" type="ChannelCharacteristics"/>
            <xs:element name="defaultClientProfile" type="DefaultClientProfile"/>
            <xs:element name="parameters" type="AXMEDISParameters" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <!-- finally ... the element -->
    <xs:element name="distributionProfile" type="DistributionProfile"/>
</xs:schema>
```


DistributionProfile
ChannelCharacteristics
bandwidth
latency
channelCharacteristics
ChannelLimitations
limitation
limitations
value
distributionProfile
DefaultClientProfile
name
defaultClientProfile
value
1..∞
AXMEDISParameters
name
parameters
value
1..∞

26 Formal description of communication protocol for Saver WebService (EXITECH)

Saver WebService	
Method	***Description***
commit_sync	This method allows a synchronous commit of an object provided in a URI by the webservice client. The response will arrive when the operation will be completed
commit_async	This method allows an asynchronous commit of an object provided in a URI by the webservice client. The response will arrive when the operation will be accepted and the result will be communicated to a Listener specified below

Saver

WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Saver"
targetNamespace="http://www.axmedis.org/saver.wsdl"
xmlns:tns="http://www.axmedis.org/saver.wsdl"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ax="urn:ax"
xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>

<schema targetNamespace="urn:ax"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ax"
 xmlns="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="unqualified"
 attributeFormDefault="unqualified">
 <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
 <complexType name="sync-result">
  <sequence>
    <element name="result" type="xsd:boolean" minOccurs="1" maxOccurs="1"/>
    <element name="version" type="xsd:int" minOccurs="1" maxOccurs="1"/>
    <element name="errormsg" type="xsd:string" minOccurs="0" maxOccurs="1" nillable="true"/>
    <element name="errorcode" type="xsd:int" minOccurs="1" maxOccurs="1"/>
  </sequence>
 </complexType>
 <!-- operation request element -->
 <element name="URI" type="xsd:anyURI"/>
 <!-- operation request element -->
 <element name="User" type="xsd:string"/>
 <!-- operation request element -->
 <element name="Password" type="xsd:string"/>
 <!-- operation response element -->
 <element name="return" type="ax:sync-result"/>
 <!-- operation request element -->
 <element name="CommitListenerService" type="xsd:anyURI"/>
 <!-- operation request element -->
 <element name="ListenerID" type="xsd:string"/>
 <!-- operation response element -->
 <element name="result" type="xsd:boolean"/>
</schema>

</types>

<message name="commit-syncRequest">
<part name="URI" element="ax:URI"/>
<part name="User" element="ax:User"/>
<part name="Password" element="ax:Password"/>
</message>

<message name="getSyncResult">
<part name="return" element="ax:return"/>
</message>

<message name="commit-asyncRequest">
<part name="URI" element="ax:URI"/>
<part name="User" element="ax:User"/>
<part name="Password" element="ax:Password"/>
<part name="CommitListenerService" element="ax:CommitListenerService"/>
<part name="ListenerID" element="ax:ListenerID"/>
</message>

<message name="commit-asyncResponse">
<part name="result" element="ax:result"/>
</message>

<portType name="SaverPortType">
<operation name="commit-sync">
 <documentation>Service definition of function ax:commit_sync</documentation>
 <input message="tns:commit-syncRequest"/>
 <output message="tns:getSyncResult"/>
</operation>
<operation name="commit-async">
 <documentation>Service definition of function ax:commit_async</documentation>
 <input message="tns:commit-asyncRequest"/>
 <output message="tns:commit-asyncResponse"/>
</operation>
</portType>

<binding name="Saver" type="tns:SaverPortType">
<SOAP:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="commit-sync">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
<operation name="commit-async">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
```

```
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
</binding>

<service name="Saver">
<documentation>gSOAP 2.7.0e generated service definition</documentation>
<port name="Saver" binding="tns:Saver">
 <SOAP:address location="http://www.axmedis.org/saver.cgi"/>
</port>
</service>

</definitions>
```

Saver	
Method	commit_sync
Description	Allow to synchronously commit an axmedis object in the AXDB
Input parameters	**xsd:anyURI URI**: URI where the axmedis object can be found **xsd:string User**: credential for commit **xsd:string Password**: credential for commit
Output parameters	A **complexType** with the following fields: **xsd:boolean result**: result of the operation **xsd:int version**: version assigned to the committed object **xsd:string errormsg**: in case of result==false this is the place for an error message **xsd:int errorcode**: in case of result==false this is the place for an error code

Saver	
Method	commit_async
Description	Allow to asynchronously commit an axmedis object in the AXDB
Input parameters	**xsd:anyURI URI**: URI where the axmedis object can be found **xsd:string User**: credential for commit **xsd:string Password**: credential for commit **xsd:anyURI CommitListenerService**: URI of the web service where a message will be send with the results of operation (a getSyncResult tag will be sent) **xsd:string ListenerID**: ID of the request set by the requester in order to collect bcak the results in a correct manner
Output parameters	**xsd:boolean result:** boolean results of the operation that means only that the service has put the request in a list and that will communicate later the result of the operation to the listener specified in input parameters

27 Formal description of communication protocol for CommitListener WebService (EXITECH)

CommitListener	
WSDL	<?xml version="1.0" encoding="UTF-8"?> <definitions name="CommitListener" targetNamespace="http://www.someone.org/listener.wsdl" xmlns:tns="http://www.someone.org/listener.wsdl" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ax="urn:ns/ax.xsd" xmlns:ns="urn:ns" xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/" xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"> <types> <schema targetNamespace="urn:ns/ax.xsd" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ax="urn:ns/ax.xsd" xmlns:ns="urn:ns" xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="unqualified" attributeFormDefault="unqualified"> <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/> <complexType name="sync-result"> <sequence> <element name="result" type="xsd:boolean" minOccurs="1" maxOccurs="1"/> <element name="version" type="xsd:int" minOccurs="1" maxOccurs="1"/> <element name="errormsg" type="xsd:string" minOccurs="0" maxOccurs="1" nillable="true"/> <element name="errorcode" type="xsd:int" minOccurs="1" maxOccurs="1"/> </sequence> </complexType> <complexType name="getSyncResult"> <sequence> <element name="return" type="ax:sync-result" minOccurs="1" maxOccurs="1"/> </sequence> </complexType> </schema> <schema targetNamespace="urn:ns" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ax="urn:ns/ax.xsd" xmlns:ns="urn:ns" xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="unqualified" attributeFormDefault="unqualified"> <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/> <!-- operation request element --> <element name="result" type="ax:getSyncResult"/> <!-- operation request element --> <element name="ListenerID" type="xsd:string"/> <!-- operation response element -->

	<element name="r" type="xsd:boolean"/> </schema> </types> <message name="commit-ListenerRequest"> <part name="result" element="ns:result"/> <part name="ListenerID" element="ns:ListenerID"/> </message> <message name="commit-ListenerResponse"> <part name="r" element="ns:r"/> </message> <portType name="CommitListenerPortType"> <operation name="commit-Listener"> <documentation>Service definition of function ns:commit_Listener</documentation> <input message="tns:commit-ListenerRequest"/> <output message="tns:commit-ListenerResponse"/> </operation> </portType> <binding name="CommitListener" type="tns:CommitListenerPortType"> <SOAP:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/> <operation name="commit-Listener"> <SOAP:operation style="rpc" soapAction=""/> <input> <SOAP:body use="literal" namespace="urn:ns"/> </input> <output> <SOAP:body use="literal" namespace="urn:ns"/> </output> </operation> </binding> <service name="CommitListener"> <documentation>gSOAP 2.7.0e generated service definition</documentation> <port name="CommitListener" binding="tns:CommitListener"> <SOAP:address location="http://www.someone.org/listener.cgi"/> </port> </service> </definitions>
Method	`commit_Listener`
Description	Listener web service that receive the result of the async commit
Input parameters	A **complexType** with the following fields: **xsd:boolean result**: result of the operation **xsd:int version**: version assigned to the committed object **xsd:string errormsg**: in case of result==false this is the place for an error message **xsd:int errorcode**: in case of result==false this is the place for an error code **xsd:string ListenerID**: ID of the request set by the requester in order to collect bcak the results in a correct manner
Output parameters	**xsd:boolean r**: true or false depending on the result of the operation

28 Formal description of communication protocol for Loader WebService (EXITECH)

Loader

WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Loader"
 targetNamespace="http://www.axmedis.org/loader.wsdl"
 xmlns:tns="http://www.axmedis.org/loader.wsdl"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ax"
 xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
 xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
 xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/"
 xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>

 <schema targetNamespace="urn:ax"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ax="urn:ax"
  xmlns="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="unqualified"
  attributeFormDefault="unqualified">
  <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
  <complexType name="checkout-result">
   <sequence>
     <element name="result" type="xsd:boolean" minOccurs="1" maxOccurs="1"/>
     <element name="downloadURI" type="xsd:anyURI" minOccurs="0" maxOccurs="1" nillable="true"/>
     <element name="errormsg" type="xsd:string" minOccurs="0" maxOccurs="1" nillable="true"/>
     <element name="errorcode" type="xsd:int" minOccurs="1" maxOccurs="1"/>
   </sequence>
  </complexType>
  <!-- operation request element -->
  <element name="AXOID" type="xsd:string"/>
  <!-- operation request element -->
  <element name="version" type="xsd:int"/>
  <!-- operation request element -->
  <element name="User" type="xsd:string"/>
  <!-- operation request element -->
  <element name="Password" type="xsd:string"/>
  <!-- operation response element -->
  <element name="return" type="ax:checkout-result"/>
  <!-- operation request element -->
  <element name="CheckoutListenerService" type="xsd:anyURI"/>
  <!-- operation request element -->
  <element name="ListenerID" type="xsd:string"/>
  <!-- operation response element -->
  <element name="result" type="xsd:boolean"/>
 </schema>

</types>

<message name="checkout-syncRequest">
 <part name="AXOID" element="ax:AXOID"/>
 <part name="version" element="ax:version"/>
 <part name="User" element="ax:User"/>
 <part name="Password" element="ax:Password"/>
</message>

<message name="getcheckoutResult">
 <part name="return" element="ax:return"/>
</message>

<message name="checkout-asyncRequest">
 <part name="AXOID" element="ax:AXOID"/>
 <part name="version" element="ax:version"/>
 <part name="User" element="ax:User"/>
 <part name="Password" element="ax:Password"/>
 <part name="CheckoutListenerService" element="ax:CheckoutListenerService"/>
 <part name="ListenerID" element="ax:ListenerID"/>
</message>

<message name="checkout-asyncResponse">
 <part name="result" element="ax:result"/>
</message>

<portType name="LoaderPortType">
 <operation name="checkout-sync">
  <documentation>Service definition of function ax:checkout_sync</documentation>
  <input message="tns:checkout-syncRequest"/>
  <output message="tns:getcheckoutResult"/>
 </operation>
 <operation name="checkout-async">
  <documentation>Service definition of function ax:checkout_async</documentation>
  <input message="tns:checkout-asyncRequest"/>
  <output message="tns:checkout-asyncResponse"/>
 </operation>
</portType>

<binding name="Loader" type="tns:LoaderPortType">
 <SOAP:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
 <operation name="checkout-sync">
  <SOAP:operation style="rpc" soapAction=""/>
  <input>
   <SOAP:body use="literal" namespace="urn:ax"/>
  </input>
  <output>
   <SOAP:body use="literal" namespace="urn:ax"/>
  </output>
 </operation>
 <operation name="checkout-async">
  <SOAP:operation style="rpc" soapAction=""/>
  <input>
   <SOAP:body use="literal" namespace="urn:ax"/>
  </input>
  <output>
   <SOAP:body use="literal" namespace="urn:ax"/>
  </output>
 </operation>
</binding>

<service name="Loader">
 <documentation>gSOAP 2.7.0e generated service definition</documentation>
 <port name="Loader" binding="tns:Loader">
  <SOAP:address location="http://www.axmedis.org/loader.cgi"/>
 </port>
</service>

</definitions>
```

Loader	
Method	checkout_sync
Description	Methods that allows to synchronously get a named version of an axmedis object or the last version of it.
Input parameters	**xsd:string AXOID**: AXOID of the object to be loaded **xsd:int version:** desired version for the object, if -1, the last version wil be returned **xsd:string User**: credential for commit **xsd:string Password**: credential for commit
Output parameters	A **complexType** named ax:getcheckoutResultwith the following fields: **xsd:boolean result**: result of the operation **xsd:anyURI downloadURI**: URI where the object can be downloaded **xsd:string errormsg**: in case of result==false this is the place for an error message **xsd:int errorcode**: in case of result==false this is the place for an error code

Loader	
Method	checkuot_async
Description	Methods that allows to asynchronously get a named version of an axmedis object or the last version of it.
Input parameters	**xsd:string AXOID**: AXOID of the object to be loaded **xsd:int version:** desired version for the object, if -1, the last version wil be returned **xsd:string User**: credential for commit **xsd:string Password**: credential for commit **xsd:anyURI CommitListenerService**: URI of the web service where a message will be send with the results of operation (a getSyncResult tag will be sent) **xsd:string ListenerID**: ID of the request set by the requester in order to collect bcak the results in a correct manner
Output parameters	**xsd:boolean result**: boolean results of the operation that means only that the service has put the request in a list and that will communicate later the result of the operation to the listener specified in input parameters

29 Formal description of communication protocol for checkoutListener Web Service (EXITECH)

<table>
<tr><th colspan="2">CheckoutListener</th></tr>
<tr><td>WSDL</td><td><pre>
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://www.someone.org/listener.wsdl" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ax="urn:ns/ax.xsd"
xmlns:ns="urn:ns" xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/" name="CkeckoutListener"
targetNamespace="http://www.someone.org/listener.wsdl">
<types>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:ns/ax.xsd" elementFormDefault="qualified"
attributeFormDefault="qualified">
<import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
<complexType name="checkout-result">
<sequence>
<element name="result" type="xsd:boolean" minOccurs="1" maxOccurs="1"/>
<element name="downloadURI" type="xsd:anyURI" minOccurs="0"
maxOccurs="1" nillable="true"/>
<element name="errormsg" type="xsd:string" minOccurs="0" maxOccurs="1"
nillable="true"/>
<element name="errorcode" type="xsd:int" minOccurs="1" maxOccurs="1"/>
</sequence>
</complexType>
<complexType name="co">
<sequence>
<element name="return" type="ax:checkout-result" minOccurs="1"
maxOccurs="1"/>
</sequence>
</complexType>
</schema>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:ns" elementFormDefault="qualified"
attributeFormDefault="qualified">
<import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
<!-- operation request element -->
<element name="checkout-Listener">
<complexType>
<sequence>
<element name="result" type="ax:co" minOccurs="1" maxOccurs="1"/>
<element name="ListenerID" type="xsd:string" minOccurs="0"
maxOccurs="1" nillable="true"/>
</sequence>
</complexType>
</element>
<!-- operation response element -->
<element name="checkout-ListenerResponse">
<complexType>
<sequence>
<element name="r" type="xsd:boolean" minOccurs="1" maxOccurs="1"/>
</sequence>
</complexType>
</element>
</pre></td></tr>
</table>

```
</schema>
</types>
<message name="checkout-ListenerRequest">
<part name="parameters" element="ns:checkout-Listener"/>
</message>
<message name="checkout-ListenerResponse">
<part name="parameters" element="ns:checkout-ListenerResponse"/>
</message>
<portType name="CkeckoutListenerPortType">
<operation name="checkout-Listener">
<documentation>Service definition of function
ns__checkout_Listener</documentation>
<input message="tns:checkout-ListenerRequest"/>
<output message="tns:checkout-ListenerResponse"/>
</operation>
</portType>
<binding name="CkeckoutListener" type="tns:CkeckoutListenerPortType">
<SOAP:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="checkout-Listener">
<SOAP:operation soapAction=""/>
<input>
<SOAP:body use="literal"/>
</input>
<output>
<SOAP:body use="literal"/>
</output>
</operation>
</binding>
<wsdl:service xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" name="loaderListener">
<wsdl:port name="CkeckoutListenerPortTypePort"
binding="tns:CkeckoutListener">
<address xmlns="http://schemas.xmlsoap.org/wsdl/soap/" location="PUT
ACTUAL ADDRESS HERE"/>
</wsdl:port>
</wsdl:service>
</definitions>
```

CheckoutListener	
Method	`ckecout_Listener`
Description	Listener that receives the results of a checkout process of an axmedis object
Input parameters	A **complexType** with name **result** with the following fields: **xsd:boolean result**: result of the operation **xsd:anyURI URI**: URI where the object is available **xsd:string errormsg**: in case of result==false this is the place for an error message **xsd:int errorcode**: in case of result==false this is the place for an error code **xsd:string ListenerID**: ID of the request set by the requester in order to collect bcak the results in a correct manner
Output parameters	**xsd:boolean r**: true or false depending on the result of the operation

30 Formal description of communication protocol for Publication_Support Web Service (EXITECH)

Publication_Support

WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Publication_support"
targetNamespace="http://www.axmedis.org/pub_support.wsdl"
xmlns:tns="http://www.axmedis.org/pub_support.wsdl"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ax="urn:ax"
xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>

<schema targetNamespace="urn:ax"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ax"
 xmlns="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="unqualified"
 attributeFormDefault="unqualified">
 <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
 <complexType name="result">
  <sequence>
    <element name="RetCode" type="xsd:int" minOccurs="1" maxOccurs="1"/>
    <element name="NumberOfResult" type="xsd:int" minOccurs="1" maxOccurs="1"/>
    <element name="res" type="xsd:string" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
 </complexType>
 <!-- operation request element -->
 <element name="date" type="xsd:string"/>
 <!-- operation request element -->
 <element name="time" type="xsd:string"/>
 <!-- operation response element -->
 <element name="return" type="ax:result"/>
</schema>

</types>

<message name="getModifiedObjectRequest">
<part name="date" element="ax:date"/>
<part name="time" element="ax:time"/>
</message>

<message name="getResult">
<part name="return" element="ax:return"/>
</message>

<portType name="Publication_supportPortType">
<operation name="getModifiedObject">
 <documentation>Service definition of function ax__getModifiedObject</documentation>
 <input message="tns:getModifiedObjectRequest"/>
 <output message="tns:getResult"/>
</operation>
</portType>
```

```
<binding name="Publication_support" type="tns:Publication_supportPortType">
<SOAP:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="getModifiedObject">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
</binding>

<service name="Publication_support">
<documentation>gSOAP 2.7.0e generated service definition</documentation>
<port name="Publication_support" binding="tns:Publication_support">
 <SOAP:address location="http://www.axmedis.org/pub_support.cgi"/>
</port>
</service>

</definitions>
```

Publication _Support	
Method	getModifiedObject
Description	Returns the list of AXOID that have been inserted/updated after a certain timestamp
Input parameters	**xsd:string date**: date after which the object has been inserted/updated. It is assumed that the date is in YYYY/MM/DD format **xsd:string time**: time after which the object has been inserted/updated. It is assumed that the time is in HH:MM:SS format related to GMT0
Output parameters	A complex type that has the following components: **xsd:int RetCode**: return code, where o means OK and greater than 0 corresponds to an error **xsd:int NumberOfResult**: number or results returned in the following sequence **unbounded sequence of xsd:string res**: a sequence (0 or more elements) of AXOID

31 Formal description of communication protocol for User _Support Web Service (EXITECH)

User _Support

WSDL

```
<schema targetNamespace="urn:ax"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ax"
 xmlns="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="unqualified"
 attributeFormDefault="unqualified">
 <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
 <!-- operation request element -->
 <element name="userName" type="xsd:string"/>
 <!-- operation request element -->
 <element name="userPassword" type="xsd:string"/>
 <!-- operation request element -->
 <element name="operation" type="xsd:int"/>
 <!-- operation response element -->
 <element name="Result" type="xsd:int"/>
</schema>

</types>

<message name="authenticate-userRequest">
<part name="userName" element="ax:userName"/>
<part name="userPassword" element="ax:userPassword"/>
<part name="operation" element="ax:operation"/>
</message>

<message name="authenticate-userResponse">
<part name="Result" element="ax:Result"/>
</message>

<portType name="User_supportPortType">
<operation name="authenticate-user">
 <documentation>Service definition of function ax__authenticate_user</documentation>
 <input message="tns:authenticate-userRequest"/>
 <output message="tns:authenticate-userResponse"/>
</operation>
</portType>

<binding name="User_support" type="tns:User_supportPortType">
<SOAP:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="authenticate-user">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
</binding>

<service name="User_support">
<documentation>gSOAP 2.7.0e generated service definition</documentation>
<port name="User_support" binding="tns:User_support">
 <SOAP:address location="http://www.axmedis.org/user_support.cgi"/>
</port>
</service>

</definitions>
```

Method	authenticateUser
Description	Returns an error code with respect to the user authentication procedure
Input parameters	**xsd:string userName**: userName to be checked **xsd:string userPassword**: password to be checked **xsd:int operationCode**: operation code to be verified (optional, or -1 means only authentication)
Output parameters	**xsd:int RetCode**: return code, where 0 means OK and greater than 0 corresponds to an error

32 Formal description of communication protocol for P2PHub_Support Web Service (EXITECH

P2PHub_Support

WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="P2PHub_support"
targetNamespace="http://www.axmedis.org/p2phub.wsdl"
xmlns:tns="http://www.axmedis.org/p2phub.wsdl"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ax="urn:ax"
xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>

<schema targetNamespace="urn:ax"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ax"
 xmlns="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="unqualified"
 attributeFormDefault="unqualified">
 <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
 <complexType name="details">
  <sequence>
    <element name="RetCode" type="xsd:int" minOccurs="1" maxOccurs="1"/>
    <element name="Axoid" type="xsd:string" minOccurs="0" maxOccurs="1" nillable="true"/>
    <element name="Version" type="xsd:int" minOccurs="1" maxOccurs="1"/>
    <element name="ObjectLocation" type="xsd:anyURI" minOccurs="0" maxOccurs="1" nillable="true"/>
  </sequence>
 </complexType>
 <!-- operation request element -->
 <element name="P2PID" type="xsd:string"/>
 <!-- operation request element -->
 <element name="Axoid" type="xsd:string"/>
 <!-- operation request element -->
 <element name="Version" type="xsd:int"/>
 <!-- operation request element -->
 <element name="ObjectLocation" type="xsd:anyURI"/>
 <!-- operation response element -->
 <element name="Result" type="xsd:int"/>
 <!-- operation response element -->
 <element name="P2PID" type="ax:details"/>
</schema>

</types>

<message name="add-P2PIDRequest">
<part name="P2PID" element="ax:P2PID"/>
<part name="Axoid" element="ax:Axoid"/>
<part name="Version" element="ax:Version"/>
<part name="ObjectLocation" element="ax:ObjectLocation"/>
</message>

<message name="add-P2PIDResponse">
<part name="Result" element="ax:Result"/>
```

```
</message>

<message name="update-P2PIDRequest">
<part name="P2PID" element="ax:P2PID"/>
<part name="Axoid" element="ax:Axoid"/>
<part name="Version" element="ax:Version"/>
<part name="ObjectLocation" element="ax:ObjectLocation"/>
</message>

<message name="update-P2PIDResponse">
<part name="Result" element="ax:Result"/>
</message>

<message name="del-P2PIDRequest">
<part name="P2PID" element="ax:P2PID"/>
</message>

<message name="del-P2PIDResponse">
<part name="Result" element="ax:Result"/>
</message>

<message name="get-P2PID-DetailsRequest">
<part name="P2PID" element="ax:P2PID"/>
</message>

<message name="P2PDetails">
<part name="P2PID" element="ax:P2PID"/>
</message>

<portType name="P2PHub_supportPortType">
<operation name="add-P2PID">
 <documentation>Service definition of function ax__add_P2PID</documentation>
 <input message="tns:add-P2PIDRequest"/>
 <output message="tns:add-P2PIDResponse"/>
</operation>
<operation name="update-P2PID">
 <documentation>Service definition of function ax__update_P2PID</documentation>
 <input message="tns:update-P2PIDRequest"/>
 <output message="tns:update-P2PIDResponse"/>
</operation>
<operation name="del-P2PID">
 <documentation>Service definition of function ax__del_P2PID</documentation>
 <input message="tns:del-P2PIDRequest"/>
 <output message="tns:del-P2PIDResponse"/>
</operation>
<operation name="get-P2PID-Details">
 <documentation>Service definition of function ax__get_P2PID_Details</documentation>
 <input message="tns:get-P2PID-DetailsRequest"/>
 <output message="tns:P2PDetails"/>
</operation>
</portType>

<binding name="P2PHub_support" type="tns:P2PHub_supportPortType">
<SOAP:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="add-P2PID">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
<operation name="update-P2PID">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
<operation name="del-P2PID">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
<operation name="get-P2PID-Details">
 <SOAP:operation style="rpc" soapAction=""/>
 <input>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </input>
 <output>
  <SOAP:body use="literal" namespace="urn:ax"/>
 </output>
</operation>
</binding>

<service name="P2PHub_support">
<documentation>gSOAP 2.7.0e generated service definition</documentation>
<port name="P2PHub_support" binding="tns:P2PHub_support">
 <SOAP:address location="http://www.axmedis.org/p2phub.cgi"/>
</port>
</service>

</definitions>
```

P2PHub_Support	
Method	add_P2PID
Description	This methods allows to add a record on the P2PHub table
Input parameters	xsd:string P2PID: the unique id of the record and primaty key xsd:string Axoid: the AXOID of the original object xsd:int Version: the version of the original object xsd:anyURI ObjectLocation: the location on the P2P network of the original object
Output parameters	xsd:int Result: result of the operation (0 means OK, other value will be decoded in error codes)

P2PHub_Support	
Method	update_P2PID
Description	This methods allows to update an existing record on the P2PHub table
Input parameters	xsd:string P2PID: the unique id of the record and primaty key xsd:string Axoid: the AXOID of the original object xsd:int Version: the version of the original object xsd:anyURI ObjectLocation: the location on the P2P network of the original object
Output parameters	xsd:int Result: result of the operation (0 means OK, other value will be decoded in error codes)

P2PHub_Support	
Method	del_P2PID
Description	This methods allows to remove an existing record on the P2PHub table
Input	xsd:string P2PID: the unique id of the record and primaty key

parameters	
Output parameters	xsd:int Result: result of the operation (0 means OK, other value will be decoded in error codes)

P2PHub_Support	
Method	get_P2PID_Details
Description	This methods allows to get AXOID, Version and URI from the P2PID
Input parameters	xsd:string P2PID: the unique id of the record and primaty key
Output parameters	A complex type that contains the following data: xsd:int Result: result of the operation (0 means OK, other value will be decoded in error codes) xsd:string Axoid: the AXOID of the original object xsd:int Version: the version of the original object xsd:anyURI ObjectLocation: the location on the P2P network of the original object

33 Formal description of communication protocol for Query_Support Web Service (EXITECH)

Query_Support

WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Query_Support"
targetNamespace="http://www.axmedis.org/query_support.wsdl"
xmlns:tns="http://www.axmedis.org/query_support.wsdl"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ax="urn:ax"
xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>

<schema targetNamespace="urn:ax"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ax"
 xmlns="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 attributeFormDefault="qualified">
 <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
 <complexType name="q">
  <sequence>
    <element name="user" type="xsd:string" minOccurs="1" maxOccurs="1"/>
    <element name="pwd" type="xsd:string" minOccurs="1" maxOccurs="1"/>
    <element name="query" type="xsd:string" minOccurs="1" maxOccurs="1"/>
  </sequence>
 </complexType>
 <complexType name="query">
  <sequence>
    <element name="send" type="ax:q" minOccurs="1" maxOccurs="1"/>
  </sequence>
 </complexType>
 <complexType name="qr">
  <sequence>
    <element name="RetCode" type="xsd:int" minOccurs="1" maxOccurs="1"/>
    <element name="XMLResult" type="xsd:string" minOccurs="1" maxOccurs="1"/>
  </sequence>
 </complexType>
 <!-- operation request element -->
 <element name="Make-Query-Sync">
  <complexType>
   <sequence>
    <element name="queryparm" type="ax:query" minOccurs="1" maxOccurs="1"/>
   </sequence>
  </complexType>
 </element>
 <!-- operation response element -->
 <element name="queryresults">
  <complexType>
```

```
    <sequence>
     <element name="return" type="ax:qr" minOccurs="1" maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation request element -->
  <element name="Make-Query-ASync">
   <complexType>
    <sequence>
     <element name="queryparm" type="ax:query" minOccurs="1" maxOccurs="1"/>
     <element name="QueryresultListenerService" type="xsd:anyURI" minOccurs="1" maxOccurs="1"/>
     <element name="ListenerID" type="xsd:string" minOccurs="1" maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="Make-Query-ASyncResponse">
   <complexType>
    <sequence>
     <element name="result" type="xsd:boolean" minOccurs="1" maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
 </schema>

</types>

<message name="Make-Query-SyncRequest">
<part name="parameters" element="ax:Make-Query-Sync"/>
</message>

<message name="queryresults">
<part name="parameters" element="ax:queryresults"/>
</message>

<message name="Make-Query-ASyncRequest">
<part name="parameters" element="ax:Make-Query-ASync"/>
</message>

<message name="Make-Query-ASyncResponse">
<part name="parameters" element="ax:Make-Query-ASyncResponse"/>
</message>

<portType name="Query_SupportPortType">
<operation name="Make-Query-Sync">
 <documentation>Service definition of function ax__Make_Query_Sync</documentation>
 <input message="tns:Make-Query-SyncRequest"/>
 <output message="tns:queryresults"/>
</operation>
<operation name="Make-Query-ASync">
 <documentation>Service definition of function ax__Make_Query_ASync</documentation>
 <input message="tns:Make-Query-ASyncRequest"/>
 <output message="tns:Make-Query-ASyncResponse"/>
</operation>
</portType>

<binding name="Query_Support" type="tns:Query_SupportPortType">
<SOAP:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="Make-Query-Sync">
 <SOAP:operation soapAction=""/>
 <input>
  <SOAP:body use="literal"/>
 </input>
 <output>
  <SOAP:body use="literal"/>
 </output>
</operation>
<operation name="Make-Query-ASync">
 <SOAP:operation soapAction=""/>
 <input>
  <SOAP:body use="literal"/>
 </input>
 <output>
  <SOAP:body use="literal"/>
 </output>
</operation>
</binding>

<service name="Query_Support">
<documentation>gSOAP 2.7.0e generated service definition</documentation>
<port name="Query_Support" binding="tns:Query_Support">
 <SOAP:address location="http://www.axmedis.org/query_support.cgi"/>
</port>
</service>

</definitions>
```

Query_Support	
Method	Make_Query_Sync
Description	Method that allows to synchronously issue a query and get the corresponding result.
Input parameters	ax:query, that is a complex type composed by a list of: xsd:string user, user for authentication xsd:string pwd, password for authentication xsd:string query, query to be performed
Output parameters	ax :queryresults, that is a complex type composed by a list of : xsd:int RetCode, that is the return code of the query, 0 if OK, other if an error occurs; xsd:string XMLResult, that is the result of the query in the format specified by the query result schema defined in this document

Query_Support	
Method	Make_Query_ASync
Description	Method that allows to asynchronously issue a query and get the corresponding result on a listener specified by the user.
Input parameters	ax:query , that is a complex type composed by a list of: xsd:string user, user for authentication xsd:string pwd, password for authentication xsd:string query, query to be performed xsd:anyURI QueryresultListenerService, that is the URI of the listener to which results have to be provided xsd:string ListenerID, listenerID that identifis on the listener the ID of the request
Output	xsd:boolean result that is true of false depending on the fact that the operation has been issued or

parameters	not.

34 Formal description of communication protocol for Query Support Listener Web Service (EXITECH)

QueryResultListener

WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="QueryResultListener"
targetNamespace="http://www.someone.org/querylistener.wsdl"
xmlns:tns="http://www.someone.org/querylistener.wsdl"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ax="urn:ns/ax.xsd"
xmlns:ns="urn:ns"
xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/">

<types>

<schema targetNamespace="urn:ns/ax.xsd"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ns/ax.xsd"
 xmlns:ns="urn:ns"
 xmlns="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 attributeFormDefault="qualified">
 <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
 <complexType name="qr">
  <sequence>
    <element name="RetCode" type="xsd:int" minOccurs="1" maxOccurs="1"/>
    <element name="XMLResult" type="xsd:string" minOccurs="1" maxOccurs="1"/>
  </sequence>
 </complexType>
 <complexType name="queryresults">
  <sequence>
    <element name="return" type="ax:qr" minOccurs="1" maxOccurs="1"/>
  </sequence>
 </complexType>
</schema>

<schema targetNamespace="urn:ns"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ax="urn:ns/ax.xsd"
 xmlns:ns="urn:ns"
 xmlns="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 attributeFormDefault="qualified">
 <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
 <!-- operation request element -->
 <element name="QueryResult-Listener">
  <complexType>
   <sequence>
    <element name="result" type="ax:queryresults" minOccurs="1" maxOccurs="1"/>
    <element name="ListenerID" type="xsd:string" minOccurs="1" maxOccurs="1" />
   </sequence>
  </complexType>
 </element>
 <!-- operation response element -->
 <element name="QueryResult-ListenerResponse">
  <complexType>
   <sequence>
    <element name="r" type="xsd:boolean" minOccurs="1" maxOccurs="1"/>
   </sequence>
  </complexType>
 </element>
</schema>

</types>

<message name="QueryResult-ListenerRequest">
<part name="parameters" element="ns:QueryResult-Listener"/>
</message>

<message name="QueryResult-ListenerResponse">
<part name="parameters" element="ns:QueryResult-ListenerResponse"/>
</message>

<portType name="QueryResultListenerPortType">
<operation name="QueryResult-Listener">
 <documentation>Service definition of function ns__QueryResult_Listener</documentation>
 <input message="tns:QueryResult-ListenerRequest"/>
 <output message="tns:QueryResult-ListenerResponse"/>
</operation>
</portType>

<binding name="QueryResultListener" type="tns:QueryResultListenerPortType">
<SOAP:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<operation name="QueryResult-Listener">
 <SOAP:operation soapAction=""/>
 <input>
  <SOAP:body use="literal"/>
 </input>
 <output>
  <SOAP:body use="literal"/>
 </output>
</operation>
</binding>

<service name="QueryResultListener">
<documentation>Exitech generated WSDL for AXMEDIS</documentation>
<port name="QueryResultListener" binding="tns:QueryResultListener">
 <SOAP:address location="http://www.someone.org/querylistener.cgi"/>
</port>
</service>

</definitions>
```

Method	`QueryResult_Listener`
Description	This service will wait for the results generated by the query results and will consume them in asynchronous manner differentiating results
Input parameters	ax:queryresult , that is a complex type composed by a list of: xsd:int RetCode, this return code will be 0 in the case a list of results is provided, greater than zero in the case we have an error, lower than 0 if this is the last set of results for the query. xsd:string XMLResult, the results if the query in the predefined format. xsd:string ListenerID, listenerID that identifis on the listener the ID of the request
Output parameters	xsd:boolean r, true if the result has been processed, false in all the other cases.

35 Formal description of communication protocol for Selection Archive Web Service (EXITECH)

Selection_Archive

WSDL

```
<?xml version="1.0" encoding="UTF-8"?><definitions
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://www.axmedis.org/selection_archive.wsdl" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ax="urn:ax"
xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/" name="Selection_Archive"
targetNamespace="http://www.axmedis.org/selection_archive.wsdl">

<types>

 <schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:ax" elementFormDefault="qualified"
attributeFormDefault="qualified">
  <import namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
  <complexType name="SelectionDetail">
   <sequence>
     <element name="Id" type="xsd:string" minOccurs="1" maxOccurs="1"/>
     <element name="Name" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="Timestamp" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
   </sequence>
  </complexType>
  <complexType name="SL">
   <sequence>
     <element name="RetCode" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
     <element name="NumberOfSelection" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
     <element name="Selection" type="ax:SelectionDetail" minOccurs="0"
maxOccurs="unbounded"/>
   </sequence>
  </complexType>
  <!-- operation request element -->
  <element name="ListUserSelection">
   <complexType>
    <sequence>
     <element name="user" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="SelectionList">
   <complexType>
    <sequence>
     <element name="selectionresult" type="ax:SL" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation request element -->
  <element name="ListEntitledSelection">
   <complexType>
    <sequence>
     <element name="user" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation request element -->
  <element name="LoadSelection">
   <complexType>
    <sequence>
     <element name="user" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="SelectionID" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="LoadSelectionResponse">
   <complexType>
    <sequence>
     <element name="Selection" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation request element -->
  <element name="SaveSelection">
   <complexType>
    <sequence>
     <element name="user" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="Selection" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="SaveSelectionResponse">
   <complexType>
    <sequence>
     <element name="result" type="xsd:boolean" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation request element -->
  <element name="DeleteSelection">
   <complexType>
    <sequence>
```

```
     <element name="user" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="SelectionID" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="DeleteSelectionResponse">
   <complexType>
    <sequence>
     <element name="result" type="xsd:boolean" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation request element -->
  <element name="ActualizeSelection-sync">
   <complexType>
    <sequence>
     <element name="user" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="selection" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="ActualizeSelection-syncResponse">
   <complexType>
    <sequence>
     <element name="actualizedSelection" type="xsd:string"
minOccurs="1" maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation request element -->
  <element name="ActualizeSelection-async">
   <complexType>
    <sequence>
     <element name="User" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="Password" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="selection" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
     <element name="ActualizeListenerService" type="xsd:anyURI"
minOccurs="1" maxOccurs="1"/>
     <element name="ListenerID" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="ActualizeSelection-asyncResponse">
   <complexType>
    <sequence>
```

```
     <element name="result" type="xsd:boolean" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
 </schema>

</types>

<message name="ListUserSelectionRequest">
 <part name="parameters" element="ax:ListUserSelection"/>
</message>

<message name="SelectionList">
 <part name="parameters" element="ax:SelectionList"/>
</message>

<message name="ListEntitledSelectionRequest">
 <part name="parameters" element="ax:ListEntitledSelection"/>
</message>

<message name="LoadSelectionRequest">
 <part name="parameters" element="ax:LoadSelection"/>
</message>

<message name="LoadSelectionResponse">
 <part name="parameters" element="ax:LoadSelectionResponse"/>
</message>

<message name="SaveSelectionRequest">
 <part name="parameters" element="ax:SaveSelection"/>
</message>

<message name="SaveSelectionResponse">
 <part name="parameters" element="ax:SaveSelectionResponse"/>
</message>

<message name="DeleteSelectionRequest">
 <part name="parameters" element="ax:DeleteSelection"/>
</message>

<message name="DeleteSelectionResponse">
 <part name="parameters" element="ax:DeleteSelectionResponse"/>
</message>

<message name="ActualizeSelection-syncRequest">
 <part name="parameters" element="ax:ActualizeSelection-sync"/>
</message>

<message name="ActualizeSelection-syncResponse">
 <part name="parameters" element="ax:ActualizeSelection-syncResponse"/>
</message>

<message name="ActualizeSelection-asyncRequest">
 <part name="parameters" element="ax:ActualizeSelection-async"/>
</message>

<message name="ActualizeSelection-asyncResponse">
 <part name="parameters" element="ax:ActualizeSelection-
asyncResponse"/>
</message>
```

```
<portType name="Selection_ArchivePortType">
 <operation name="ListUserSelection">
  <documentation>Service definition of function
ax__ListUserSelection</documentation>
  <input message="tns:ListUserSelectionRequest"/>
  <output message="tns:SelectionList"/>
 </operation>
 <operation name="ListEntitledSelection">
  <documentation>Service definition of function
ax__ListEntitledSelection</documentation>
  <input message="tns:ListEntitledSelectionRequest"/>
  <output message="tns:SelectionList"/>
 </operation>
 <operation name="LoadSelection">
  <documentation>Service definition of function
ax__LoadSelection</documentation>
  <input message="tns:LoadSelectionRequest"/>
  <output message="tns:LoadSelectionResponse"/>
 </operation>
 <operation name="SaveSelection">
  <documentation>Service definition of function
ax__SaveSelection</documentation>
  <input message="tns:SaveSelectionRequest"/>
  <output message="tns:SaveSelectionResponse"/>
 </operation>
 <operation name="DeleteSelection">
  <documentation>Service definition of function
ax__DeleteSelection</documentation>
  <input message="tns:DeleteSelectionRequest"/>
  <output message="tns:DeleteSelectionResponse"/>
 </operation>
 <operation name="ActualizeSelection-sync">
  <documentation>Service definition of function
ax__ActualizeSelection_sync</documentation>
  <input message="tns:ActualizeSelection-syncRequest"/>
  <output message="tns:ActualizeSelection-syncResponse"/>
 </operation>
 <operation name="ActualizeSelection-async">
  <documentation>Service definition of function
ax__ActualizeSelection_async</documentation>
  <input message="tns:ActualizeSelection-asyncRequest"/>
  <output message="tns:ActualizeSelection-asyncResponse"/>
 </operation>
</portType>

<binding name="Selection_Archive" type="tns:Selection_ArchivePortType">
 <SOAP:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
 <operation name="ListUserSelection">
  <SOAP:operation soapAction=""/>
  <input>
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
 <operation name="ListEntitledSelection">
  <SOAP:operation soapAction=""/>
  <input>
```

```
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
 <operation name="LoadSelection">
  <SOAP:operation soapAction=""/>
  <input>
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
 <operation name="SaveSelection">
  <SOAP:operation soapAction=""/>
  <input>
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
 <operation name="DeleteSelection">
  <SOAP:operation soapAction=""/>
  <input>
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
 <operation name="ActualizeSelection-sync">
  <SOAP:operation soapAction=""/>
  <input>
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
 <operation name="ActualizeSelection-async">
  <SOAP:operation soapAction=""/>
  <input>
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
</binding>

<service name="Selection_Archive">
 <documentation>gSOAP 2.7.0e generated service
definition</documentation>
 <port name="Selection_Archive" binding="tns:Selection_Archive">
  <soap:address location="http://localhost:8080/SelectionWS/sa"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"/>
 </port>
```

	`</service>`
	`</definitions>`

Selection_Archive	
Method	DeleteSelection
Description	This method allows to delete a selection that is present in the archive
Input parameters	xsd:string user: user name for authentication and authorization xsd:string pwd: password for authentication and authorization xsd:string SelectionID: ID of the selection to be deleted
Output parameters	xsd:boolean result: true or false depending on the fact that the selection has been deleted or not.

Selection_Archive	
Method	SaveSelection
Description	This method allow to save a selection created by the user
Input parameters	xsd:string user: user name for authentication and authorization xsd:string pwd: password for authentication and authorization xsd:string Selection: the selection in XML format that have to be stored
Output parameters	xsd:boolean result: true of false depending on the success of the save operation

Selection_Archive	
Method	LoadSelection
Description	this methods allows to load a selection from the archive
Input parameters	xsd:string user: user name for authentication and authorization xsd:string pwd: password for authentication and authorization xsd:string SelectionID: ID of the selection to be loaded
Output parameters	xsd:string Selection: the selection in XML format that have been loaded

Selection_Archive	
Method	ListUserSelection
Description	This method returns the selection created by the user
Input parameters	xsd:string user: user name for authentication and authorization xsd:string pwd: password for authentication and authorization
Output parameters	ax:SelectionList Selections: list of selection that have been created by the user; this parameter is a sequence of: xsd:int RetCode, that is the return code, where 0 means operation success xsd:int NumberOfSelection, that is the number of selection returned a list of ax:SelectionDetail, where each element is defined by: xsd:string Id, that is the ID of the Selection xsd:string Name, that is the name of the selection xsd:string Timestamp, that is the timestamp of the selection

Selection_Archive	
Method	ListEntitledSelection
Description	This method returns the selection for which the user is entitled that are the selection created by the user and the selection for which has been entitled as the user become to a group
Input parameters	xsd:string user: user name for authentication and authorization

	xsd:string pwd: password for authentication and authorization
Output parameters	ax:SelectionList Selections: list of selection that have been created by the user; this parameter is a sequence of: xsd:int RetCode, that is the return code, where 0 means operation success xsd:int NumberOfSelection, that is the number of selection returned a list of ax:SelectionDetail, where each element is defined by: xsd:string Id, that is the ID of the Selection xsd:string Name, that is the name of the selection xsd:string Timestamp, that is the timestamp of the selection

Selection_Archive	
Method	ListUserSelection
Description	This method returns the selection created by the user
Input parameters	xsd:string user: user name for authentication and authorization xsd:string pwd: password for authentication and authorization
Output parameters	ax:SelectionList Selections: list of selection that have been created by the user; this parameter is a sequence of: xsd:int RetCode, that is the return code, where 0 means operation success xsd:int NumberOfSelection, that is the number of selection returned a list of ax:SelectionDetail, where each element is defined by: xsd:string Id, that is the ID of the Selection xsd:string Name, that is the name of the selection xsd:string Timestamp, that is the timestamp of the selection

[EXITECH] NEW … PAY ATTENTION … SERVICE CHANGED TO HAVE A MORE EFFECTIVE INTERFACE

Selection_Archive	
Method	ActualizeSelection_sync
Description	This methods return synchronously the list of AXOID that the selection represent in that moment. The list of AXOID is in the format of a selection as formalized in the selection definition
Input parameters	xsd:string user: username to the authorized to perform the operation xsd:string pwd: password of the user for authentication xsd:string selection: selection that have to be actualized
Output parameters	xsd::string actualized selction or null in the case of error. If some sources are not available only a partial selection will be returned.

[EXITECH] NEW … PAY ATTENTION … SERVICE CHANGED TO HAVE A MORE EFFECTIVE INTERFACE

Selection_Archive	
Method	ActualizeSelection_async
Description	This methods return asynchronously the list of AXOID that the selection represent in that moment. The list of AXOID is in the format of a selection as formalized in the selection definition and will be returned to a listener provided by the who uses this method. A possible listener is proposed.
Input parameters	xsd:string user: username to the authorized to perform the operation xsd:string pwd: password of the user for authentication xsd:string selection: selection that have to be actualized xsd:anyURI ActualizeListenerService: address of the listening web service xsd:string ListenerID: ID of the notification

Output parameters	xsd:Boolean result: true if the operation has been put in the queue of the operations to be performed

36 Formal description of communication protocol for Actualize Listener

ActualizeListener

WSDL

```
<?xml version="1.0" encoding="UTF-8"?><definitions
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://www.someone.org/actualizelistener.wsdl"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ns="urn:ns"
xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/"
xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/"
name="ActualizeListener"
targetNamespace="http://www.someone.org/actualizelistener.wsdl">

<types>

 <schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:ns" elementFormDefault="qualified"
attributeFormDefault="qualified">
  <import
namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
  <!-- operation request element -->
  <element name="Actualize-Listener">
   <complexType>
    <sequence>
     <element name="actualizedSelection" type="xsd:string"
minOccurs="1" maxOccurs="1"/>
     <element name="ListenerID" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
  <!-- operation response element -->
  <element name="Actualize-ListenerResponse">
   <complexType>
    <sequence>
     <element name="r" type="xsd:boolean" minOccurs="1"
maxOccurs="1"/>
    </sequence>
   </complexType>
  </element>
 </schema>

</types>

<message name="Actualize-ListenerRequest">
 <part name="parameters" element="ns:Actualize-Listener"/>
</message>

<message name="Actualize-ListenerResponse">
 <part name="parameters" element="ns:Actualize-
ListenerResponse"/>
</message>

<portType name="ActualizeListenerPortType">
 <operation name="Actualize-Listener">
  <documentation>Service definition of function
```

```
ns__Actualize_Listener</documentation>
  <input message="tns:Actualize-ListenerRequest"/>
  <output message="tns:Actualize-ListenerResponse"/>
 </operation>
</portType>

<binding name="ActualizeListener"
type="tns:ActualizeListenerPortType">
 <SOAP:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
 <operation name="Actualize-Listener">
  <SOAP:operation soapAction=""/>
  <input>
   <SOAP:body use="literal"/>
  </input>
  <output>
   <SOAP:body use="literal"/>
  </output>
 </operation>
</binding>

<service name="ActualizeListener">
 <documentation>gSOAP 2.7.0e generated service
definition</documentation>
 <port name="ActualizeListener" binding="tns:ActualizeListener">
  <soap:address
location="http://localhost:8080/SelectionWS/listener"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"/>
 </port>
</service>

</definitions>
```

Method	QueryResult_Listener
Description	This method is called to notify the actualized result of a selection
Input parameters	A complex type created by: xsd_string:selection actualized selection xsd:string ListenerID: id of the operation
Output parameters	xsd:boolean r: Boolean ack of the receipt of resposnse

37 Formal description of communication protocol for Locking web service

LockUnlockWS	
WSDL	<?xml version="1.0" encoding="UTF-8"?><definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://www.exitech.it/axmedis/LockUnlockWS" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:ax="http://www.exitech.it/axmedis/LockUnlock" xmlns:SOAP="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:MIME="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:DIME="http://schemas.xmlsoap.org/ws/2002/04/dime/wsdl/" xmlns:WSDL="http://schemas.xmlsoap.org/wsdl/" name="LockUnlock" targetNamespace="http://www.exitech.it/axmedis/LockUnlockWS">

```
<types>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.exitech.it/axmedis/LockUnlock"
elementFormDefault="qualified"
attributeFormDefault="qualified">
<import
namespace="http://schemas.xmlsoap.org/soap/encoding/"/>
<!-- operation request element -->
<element name="lockObject">
<complexType>
<sequence>
<element name="userId" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
<element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
<element name="axoid" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
<element name="version" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
</sequence>
</complexType>
</element>
<!-- operation response element -->
<element name="lockObjectResponse">
<complexType>
<sequence>
<element name="Result" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
</sequence>
</complexType>
</element>
<!-- operation request element -->
<element name="unlockObject">
<complexType>
<sequence>
<element name="userId" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
<element name="pwd" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
<element name="axoid" type="xsd:string" minOccurs="1"
maxOccurs="1"/>
<element name="version" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
</sequence>
</complexType>
</element>
<!-- operation response element -->
<element name="unlockObjectResponse">
<complexType>
<sequence>
<element name="Result" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
</sequence>
</complexType>
</element>
</schema>
</types>
<message name="lockObjectRequest">
<part name="parameters" element="ax:lockObject"/>
</message>
<message name="lockObjectResponse">
```

	<part name="parameters" element="ax:lockObjectResponse"/> </message> <message name="unlockObjectRequest"> <part name="parameters" element="ax:unlockObject"/> </message> <message name="unlockObjectResponse"> <part name="parameters" element="ax:unlockObjectResponse"/> </message> <portType name="LockUnlockPortType"> <operation name="lockObject"> <documentation>Service definition of function ax__lockObject</documentation> <input message="tns:lockObjectRequest"/> <output message="tns:lockObjectResponse"/> </operation> <operation name="unlockObject"> <documentation>Service definition of function ax__unlockObject</documentation> <input message="tns:unlockObjectRequest"/> <output message="tns:unlockObjectResponse"/> </operation> </portType> <binding name="LockUnlock" type="tns:LockUnlockPortType"> <SOAP:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/> <operation name="lockObject"> <SOAP:operation soapAction=""/> <input> <SOAP:body use="literal"/> </input> <output> <SOAP:body use="literal"/> </output> </operation> <operation name="unlockObject"> <SOAP:operation soapAction=""/> <input> <SOAP:body use="literal"/> </input> <output> <SOAP:body use="literal"/> </output> </operation> </binding> <wsdl:service xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" name="LockUnlockWS"> <wsdl:port name="LockUnlockPortTypePort" binding="tns:LockUnlock"> <soap:address location="http://localhost:8080/LockUnlockWS/lockunlock"/> </wsdl:port> </wsdl:service> </definitions>
Method	lockObject
Description	This method is called to issue a lock on a version of an AXMEDIS object
Input parameters	xsd:string userId: user that locks the object xsd:string pwd: password to authenticate the user

	xsd:string axoid: AXOID of the object to be locked xsd:int version: version of the object to be locked
Output parameters	xsd:int boolean Result: Error code for the operation
Method	unlockObject
Description	This method is called to release a lock on a version of an AXMEDIS object. Only who has issued the lock or an administrator can release the lock
Input parameters	xsd:string userId: user that locks the object xsd:string pwd: password to authenticate the user xsd:string axoid: AXOID of the object to be locked xsd:int version: version of the object to be locked
Output parameters	xsd:int boolean Result: Error code for the operation

38 Bibliography

AXMEDIS Deliverable DE 3.1.2 Part E AXFW-Spec-(Database-gathering)
AXMEDIS Deliverable DE 3.1.2.2.9 AXFW-Spec-(Database-gathering) first update of DE 3.1.2