Automating Production of Cross Media Content for Multi-channel Distribution

www.AXMEDIS.org

DE9.1.5
Integrated Prototype of CMS integration and feedback

Version: 1.1
Date: 22/07/2008 14:25:00
Responsible: EXITECH (ff@exitech.it)

Project Number: IST-2-511299
Project Title: AXMEDIS
Deliverable Type: Public
Visible to User Groups: Yes
Visible to Affiliated: Yes
Visible to Public: Yes

Deliverable Number: DE9.1.5
Contractual Date of Delivery: see annex I
Work-Package contributing to the Deliverable: WP 9.1.2 and sun WPs
Nature of the Deliverable: Prototype and Report
Author(s): EXITECH, DSI, SEJER, AFI, ANSC, ILABS, EUTELSAT, TISCALI, XIM, TEO, ELION, VRS, SDAE

Abstract:
This deliverable is related to the integrated prototype of WP9.1.2 that has been finalized apart from some residual optimization, some maintenance and future change not foreseen at the moment by producing a real integrated prototype cooperating with CAMART, AXDB and therefore also with AXCS.
This deliverable is an update of DE 9.1.4.

Keyword List:
Administrative data, AXCS, AXDB, CAMART

Document responsible

name: Fabrizio Fioravanti
Email address: ff@exitech.it
Affiliation acronym: EXITECH

Table of Contents

1 Executive Summary and Report Scope

This deliverable is the accompanying document for the Prototype having the name common to the deliverable. This deliverable contains all the details related to the prototype from the user requirements and use case to the specification and manuals plus a forecast for validation phase.
This deliverable is an update of DE 9.1.4 and therefore contains all the information of 9.1.4 updated and revised where needed.

2 Introduction

This deliverable is related to the mock up of WP9.1.2 that has been finalized apart from some residual optimization, some maintenance and future change not foreseen at the moment by producing a real integrated prototype cooperating with CAMART, AXDB and therefore also with AXCS.

This integrated prototype is able to perform the following tasks:

- Getting logs coming from AXCS web service
- Organizing such logs in the internal database of the factory
- Generating internal XML format in polling mode
- Generating the XML format provided by all of the partners according to the specification, and publishing at a predefined time frequency the resulting XML in an ftp directory
- Generating also in polling mode an XML format that is transformed according to the profiled XSLT.
- Generating top-bottom ten on demand for statistics.
- Authentication
- Availability of a web service for gathering statistics instead of the GUI only
- Availability of a Web service to remotely control AII completely in order to be able to automate log collection

3 User Requirements

The requirements that are reported here are taken from DE 2.1.1.2.1.

3.1 Accounting Manager and Reporting Tools

Accounting managing and reporting tool should collect information from AXMEDIS Certifier and Supervisor and Administrative Information Integrator and log these data in Account/public log database.

11.5.1) DRM modelling, modelling licensing. Study Rights information models
11.5.2) Modelling database for licensing and transaction tracking
11.5.3) Communicating with the AXMEDIS Certifier and Supervisor to get specific information related to the transactions performed on the objects of a given content provider or aggregator.
11.5.4) Storing into the AXMEDIS database the transactions, matching who has done the action on what;
11.5.5) Listing clients of the provider, with the history of their transactions, etc.
11.5.6) Listing objects for which the user has authorization
11.5.7) Listing all distributors and higher level
11.5.8) Generate any kind of report among those for which data are present and accessible to the user who requested the report.
11.5.9) Report and statistic analysis data generation in order to empower content owner and distributors to improve, and possibly automate the process of building statistically based promotions and personalised offer at least as detailed hereafter:

- Generate a list of the most used/acquired contents
- Generate a list of the most used/acquired contents per user
- Generate a list of the most used/acquired contents per distributor
- Generate a list of the most used/acquired contents per category
- Generate a list of the most used/acquired contents per user and category
- Generate a list of the most used/acquired contents per distributor and category
- Generate a list of the most used/acquired contents per distributor, user and category
- Generate a list of the least used/acquired contents
- Generate a list of the least used/acquired contents per user
- Generate a list of the least used/acquired contents per distributor
- Generate a list of the least used/acquired contents per category
- Generate a list of the least used/acquired contents per user and category
- Generate a list of the least used/acquired contents per distributor and category
- Generate a list of the leats used/acquired contents per distributor, user and category

11.5.10) Statistic analysis of the content usage, very useful for tuning the service and the structure of the ready to use proposed objects. Among these statistics it would be advisable to have the following ones: (should be addressed by external tools since statistical data are not stored in AXDB)

- Content sorted by usage rate
- Content used sorted by category
- Content used sorted by category and usage rate
- Top 10 used Content sorted by category
- Top 20 used Content sorted by category
- Bottom 10 used Content sorted by category
- Bottom 20 used Content sorted by category

11.5.11) Compute statistics about the access, utilisation, distribution etc. of the AXMEDIS objects based on the event reports previously generated. Among these statistics it would be advisable to have the following ones: (should be addressed by external tools simce statistical data are not stored in AXDB)

- Content sorted by access rate
- Content accessed sorted by category
- Content accessed sorted by category and access rate
- Top 10 accessed Content sorted by category
- Top 20 accessed Content sorted by category
- Bottom 10 accessed Content sorted by category

- Bottom 20 accessed Content sorted by category

3.2 Administrative Information Integrator

This is a set of tools for making available the administrative information received from the AXMEDIS certifier and supervisor and collected into the AXMEDIS database (managed by the Accounting Managing and Reporting tool) into the database of the Content Providers in their administrative form. For example, to bring administrative information into XAURA, HP CMS, XX CMS, etc. For this purpose, in WP 9.1 several Administrative Information Integrators will be realised. The idea is to find a common basis among them and to customise the application according to the needs and protocols to interact with the different CMSs.

The Administrative Information Integrator shall:

3.2.1) interface with different CMS technology;
3.2.2) store administrative information into the Content Provider database.
3.2.3) communicate with the AXDB to get administrative information related to a specific Content Provider.
3.2.4) guarantee privacy of sensitive data via protection mechanisms

4 Use Cases

In this section the use cases reported in DE 2.1.1.2.2 are listed and at the end a general use case with the last addition in terms of functionalities.

4.1 AXMEDIS Reporting and Statistics Web Service (DSI, EXITECH)

4.1.1 Object usage reporting for accounting purposes (EXITECH, DSI)

UCId	UC13.2.5.1
Use case	Object usage reporting for accounting purposes
Description	An Actor wants data about object usage for accounting purposes
Actors	Business users interested in accounting activities (e.g. Collecting Societies, Distributors, etc.), CAMART, AXCS
Assumptions	None
Steps	1 A business users interested in accounting activities sends a report request to the CAMART providing its own id, a timestamp and some other criteria 2 The service checks and validates the received data 3 The service collects information related to the requestor pertinent object usage from AXCS using the Reporting Web Service and sends it back to the requester
Post-conditions	The Actor has the report
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	CAMART is the only tool dealing with Reporting Web Service

4.1.2 Object usage reporting for statistics purposes (EXITECH, DSI)

UCId	UC13.2.5.2
Use case	Object usage reporting for statistics purposes
Description	An Actor wants data about object usage for statistics purposes
Actors	Business users interested in statistic activities (e.g. Collecting Societies, etc.), CAMART, AXCS
Assumptions	None
Steps	1 A business users interested in statistic activities sends a report request to the CAMART providing its own id, a timestamp and some other criteria 2 The service checks and validates the received data 3 The service collects anonymous information about object usage from AXCS using the Statistcs Web Service and sends it back to the requester
Post-conditions	The Actor has the report
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	CAMART is the only tool dealing with Statistics Web Service

4.2 Accounting Manager and Reporting Tool (EXITECH)

4.2.1 List of all operations performed on an object

UCId	UC13.2.5.1
Use case	List of all operations performed on an object
Description	An Actor wants to know all operations performed on an object
Actors	Content provider, distributor, collecting society
Assumptions	None

Steps	1 An Actor sends the request for having the list of all operation of an object: the request contain the Object ID and the code of the operation to be performed, and time window related to the period 2 The service checks and validates the data received 3 The service collects and sends back information related to the object in the database
Post-conditions	The Actor has the list
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.2.2 List of all operations performed by a user

UCId	UC13.2.5.2
Use case	List of all operations performed by a user
Description	An Actor wants to know all operations performed by a user
Actors	Distributor
Assumptions	None
Steps	1 An Actor sends the request for having the list of all operation performed by an user: the request contain the User ID, the Actor ID and the code of the operation to be performed, and time window related to the period 2 The service checks and validates the data received 3 The service collects and sends back Action-Logs related to the user
Post-conditions	The Distributor has the list
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.2.3 Usage report about an object

UCId	UC13.2.5.3
Use case	Usage report about an object
Description	An Actor wants to know usage statistic about an object
Actors	Distributor, Collecting society, content owner, content provider
Assumptions	None
Steps	1 An Actor sends the request for having the statistic about an object usage: the request contain the Object ID, the Actor ID and the code of the operation to be performed, and time window related to the period 2 The service checks and validates the data received 3 The service collects and sends back information related to the object usage
Post-conditions	The Actor obtains the statistic
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.2.4 Usage report about a distributor

UCId	UC13.2.5.4
Use case	Usage report about a distributor
Description	An Actor wants to know usage statistic about a distributor
Actors	Distributor, Collecting society, content owner, content provider
Assumptions	None
Steps	1 The Actor sends a request to the service: the request contains the Distributor ID, the Actor ID and the operation to be performed, and time window related

	to the period 2 The service checks and validates the data received 3 The service collects and sends back information related to the distributor
Post-conditions	The Actor obtains the statistic
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.2.5 Usage report about a provider

UCId	UC13.2.5.5
Use case	Usage report about a provider
Description	An Actor wants to know usage statistic about a provider
Actors	Distributor, Collecting society, content owner, content provider
Assumptions	None
Steps	1 The Actor sends a request to the service: the request contains the Provider ID, the Actor ID and the operation to be performed, and time window related to the period 2 The service checks and validates the data received 3 The service collects and sends back information related to the provider
Post-conditions	The Actor obtains the statistic
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.2.6 List objects for which an administrative account can be requested

UCId	UC13.2.5.6
Use case	List objects which an administrative account can be requested
Description	An Actor wants to obtain and consult the list of objects for which an administrative account can be requested, e.g. all the objects for which the Actor has the eligibility to obtain an administrative report)
Actors	Content creators, distributors, collecting society, content providers
Assumptions	None
Steps	1 An Actor requests the list of objects for which an administrative account can be requested 2 The reporting tool searches in the corresponding database all the AXMEDIS objects for which the actor is eligible to ask a report. 3 The reporting tool lists all the results.
Post-conditions	None
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.2.7 Listing AXMEDIS clients of a distributor/channel

UCId	UC13.2.5.7
Use case	Listing all AXMEDIS clients
Description	An Actor wants to consult the AXMEDIS clients list that has been connected to the distributor or on a channel.
Actors	Distributors.
Assumptions	None
Steps	1 The Actor asks for the list of AXMEDIS clients that have been connected to a distributor of a channel

	2 The reporting tool searches in the corresponding database all the AXMEDIS clients satisfying point 1. 3 The reporting tool lists all the results.
Post-conditions	None
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.2.8 Listing distributors

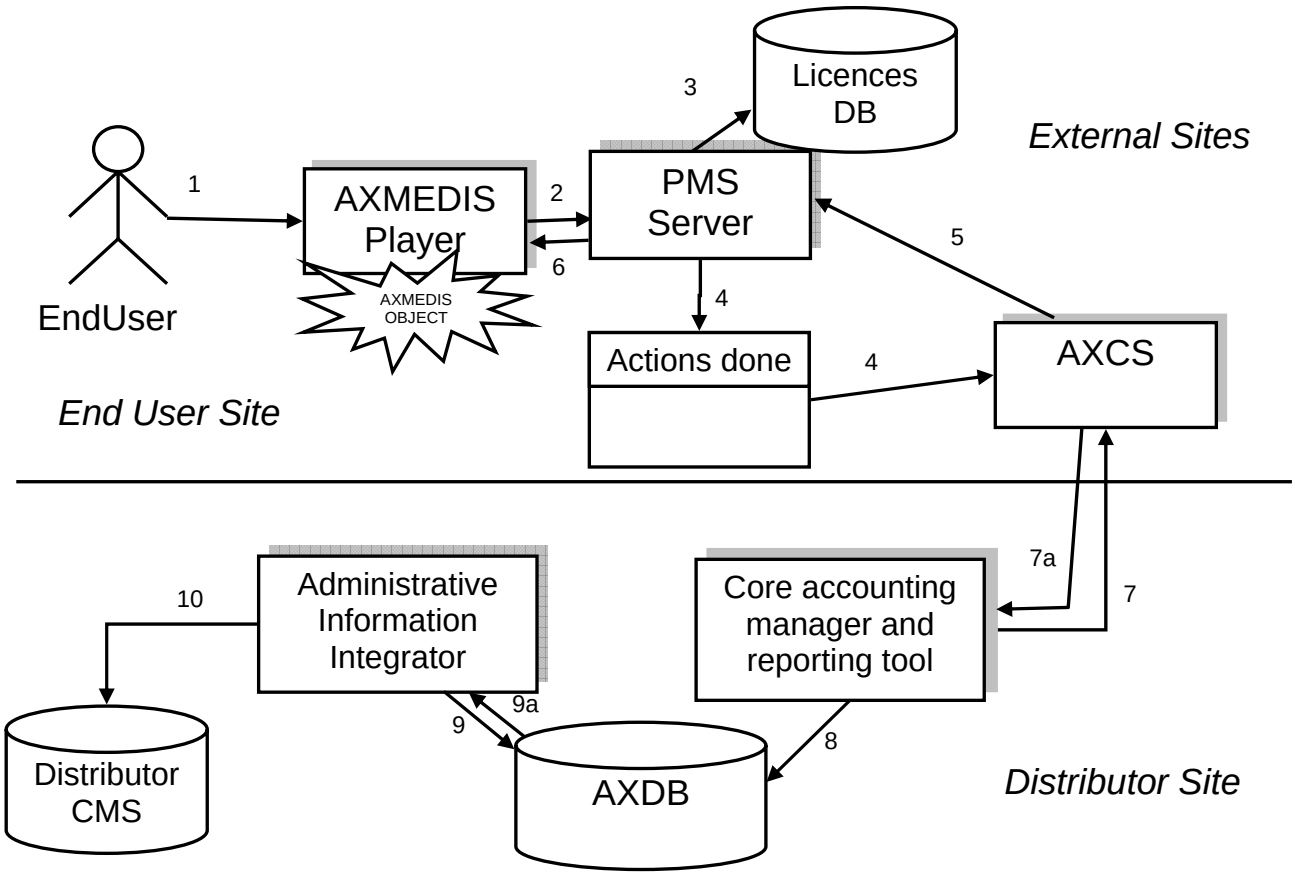
UCId	UC13.2.5.8
Use case	Listing all distributors of AXMEDIS objects, those that have redistributed its objects.
Description	An Actor wants to consult the distributors of AXMEDIS objects.
Actors	Content creators, Distributors, End users, Content Providers.
Assumptions	None
Steps	1 An actor asks for a list of distributors of AXMEDIS objects 2 The reporting tool searches in the corresponding database all the distributors of AXMEDIS objects 3 The reporting tool lists all the results.
Post-conditions	None
Variations	None
Asynchronous actions	None
Design suggestions	None
Issues	None

4.3 General Use Cases and scenarios

In this section the general use case from several different documents have been collected.:

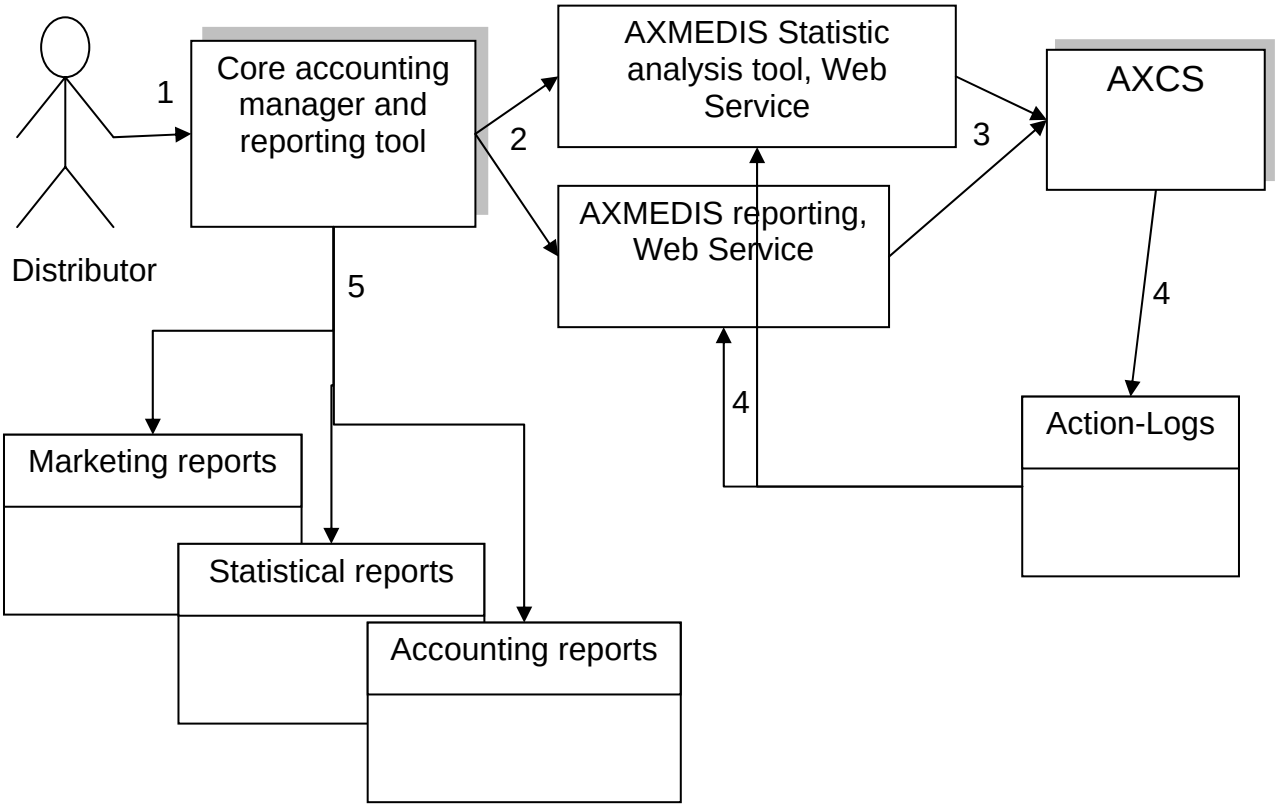
- Mapping of Administrative Information
- Distributor wants Administrative Information
- Creator or Collecting Society wants Administrative Information
- Administrative Information Integrator
- Administrative Information Integrator as seen from Collecting society perspective
- Relationships between CAMART and AII

4.3.1 Mapping of Administrative Information



1. End User requests to perform an action on an AXMEDIS Protected Object
2. AXMEDIS Player asks PMS to perform an Action (assuming client has been already certified)
3. PMS checks in the Licence DB if the Action is allowed (assuming OK)
4. PMS sends AXCS the action performed
5. AXCS returns the key to access the content (if necessary)
6. PMS grants the access to the content and possibly returns the key to the AXMEDIS Player
7. CAMART retrieves from AXCS the actions performed by all the End Users on objects distributed by the distributor
8. CAMART stores the transactions into the AXDB
9. Administrative Information Integrator gets transactions performed from the AXDB
10. Administrative information are mapped into the Distributor CMS

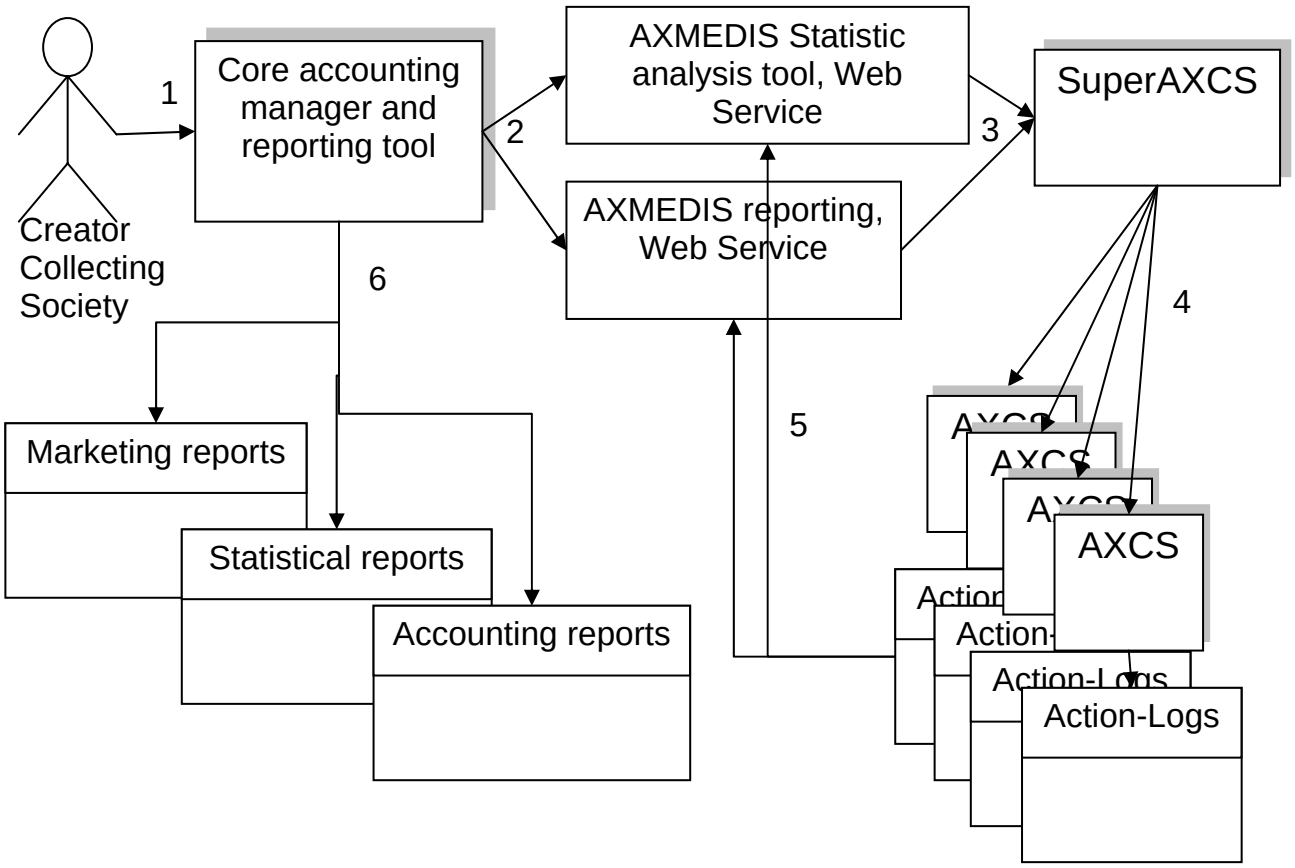
4.3.2 Distributor wants Administrative Information



Distributor, producer, collecting Site

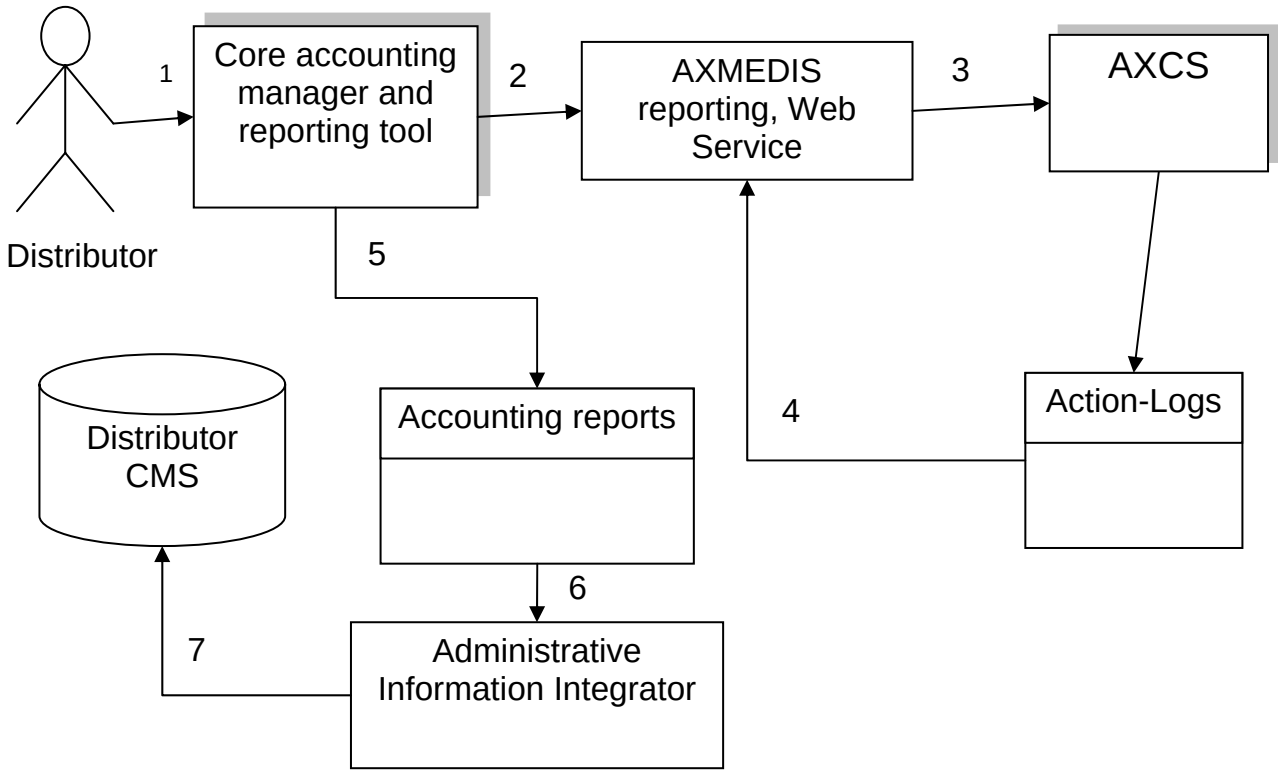
1. A Distributor wants to obtain information on actions performed on the objects he has rights for.
2. CAMART queries the correct tool for obtaining the Action-Logs in the correct form (anonymous or not, aggregated or not, etc)
3. AXMEDIS Statistic or reporting tools query AXCS
4. AXCS extracts the required Action-Logs and communicate them to the tools that perform actions to return results in the desired form
5. Different reports are generated on the basis of the information collected.

4.3.3 Creator or Collecting Society wants Administrative Information



1. An Actor, that is collecting society or creator, wants to recover information on actions performed on the objects he has rights.
2. Core accounting manager and reporting tool query the correct tool for obtaining the Action-Logs in the correct form (anonymous or not, aggregated or not, etc)
3. AXMEDIS Statistic or reporting tools query the SuperAXCS
4. SuperAXCS recover information from the different AXCSs
5. The different AXCSs extract the required Action-Logs and communicate them to the tools that perform actions to return results in the desired form
6. Different reports are generated on the basis of the information collected.

4.3.4 Administrative Information Integrator



1. A Distributor wants to recover information on actions performed on the objects he has rights.
2. CAMART queries the reporting web service to obtain the Action-Logs
3. AXMEDIS Statistic or reporting tools queries AXCS
4. AXCS extracts the required Action-Logs and communicate them to the reporting tool
5. Accounting report is generated.
6. Accounting report is passed to the Administrative Information Integrator
7. Data are loaded into the Distributor CMS

4.3.5 Administrative Information Integrator as seen from Collecting society perspective

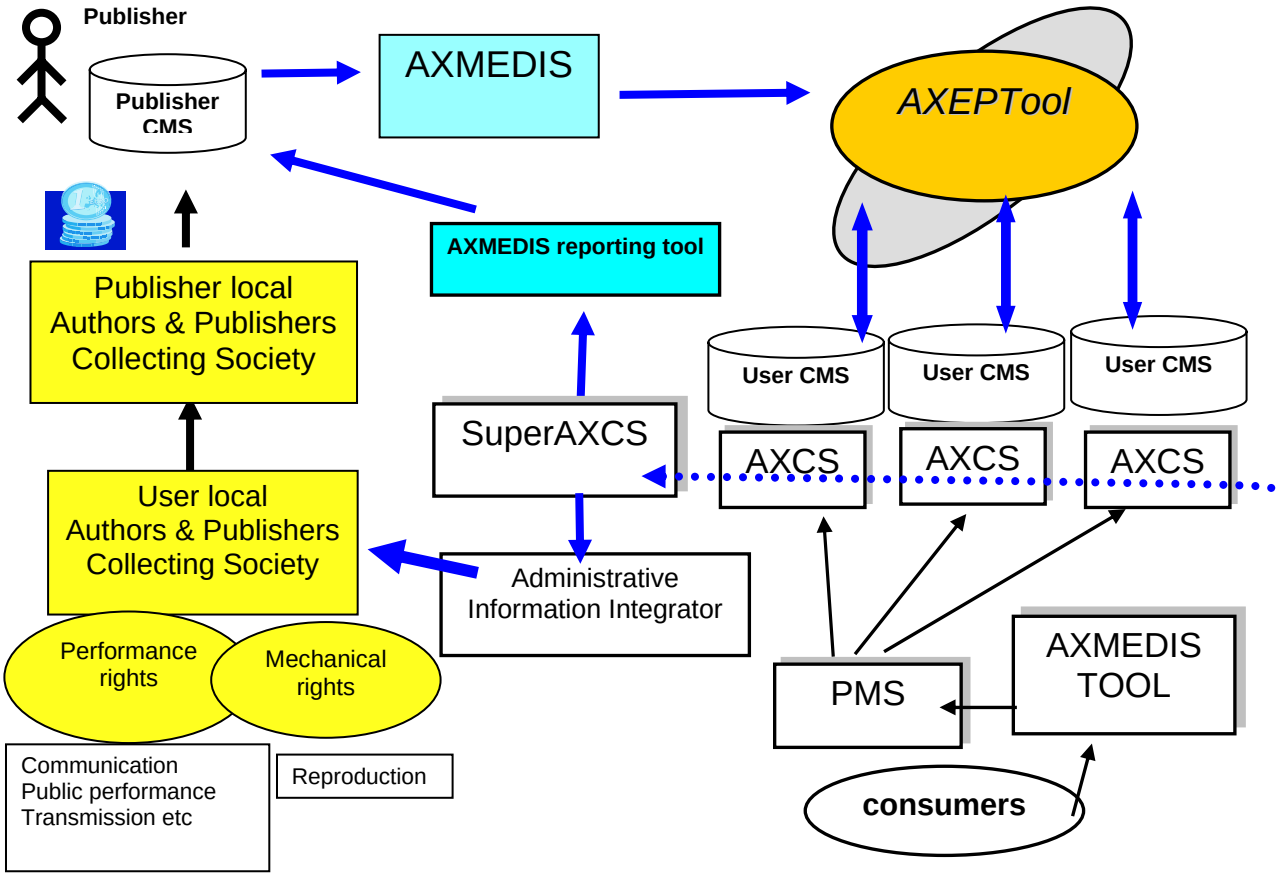
The following three scenarios describe how AXMEDIS provides a service to the Collecting Societies, supporting them in gathering reporting information on the use of protected objects so to enhance the management, ease the administration administration and enforce the rights they are vested in or represent.

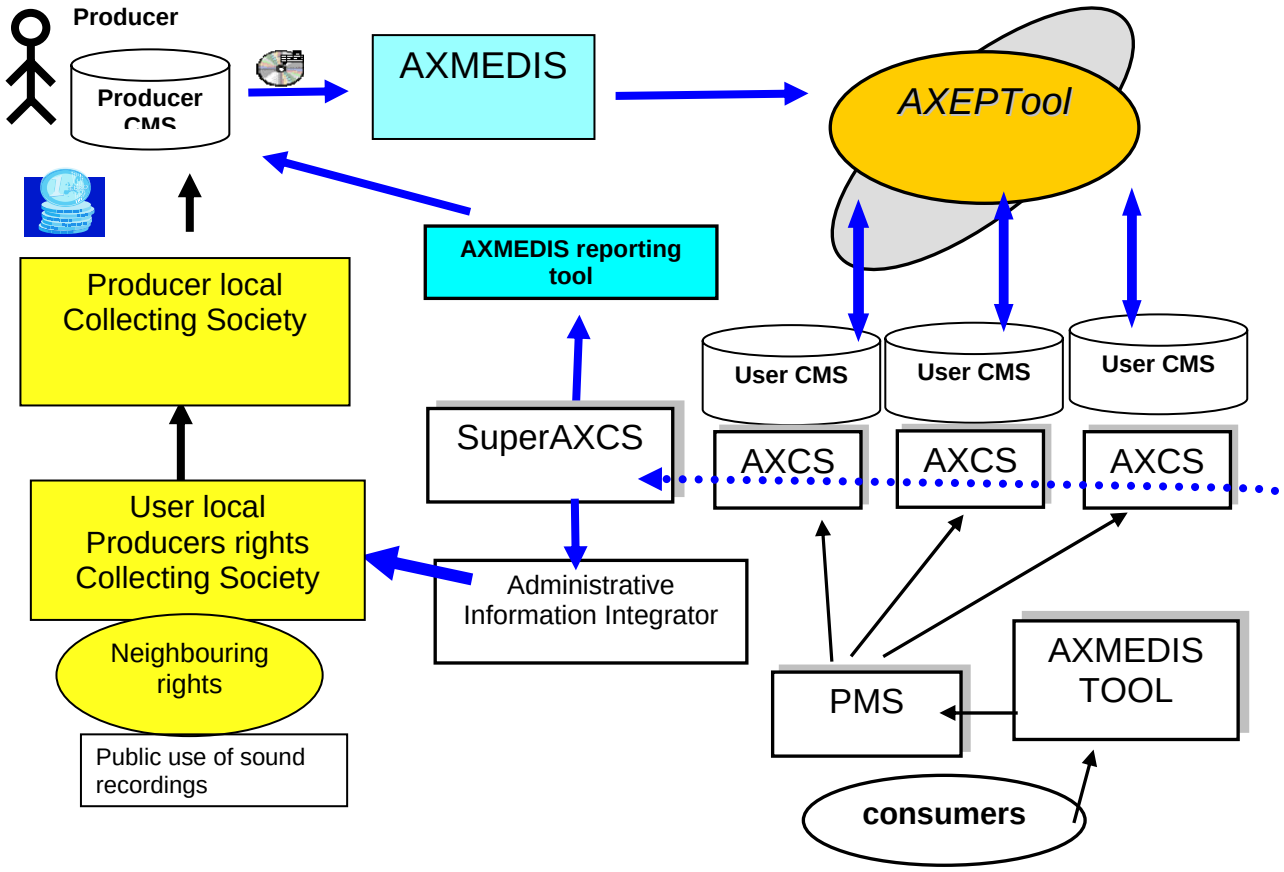
Collecting societies administer a wide range of rights on behalf of copyright owners for a wide range of uses and users. They collect and distribute to right owners royalty income and equitable remuneration in relation to the exercise of these rights. In addition to these core functions, there are many other functions carried out by all or many Collecting societies such as enforcement, monitoring and auditing activities, particularly important in view of the increasingly uses of copyrighted content in the AXMEDIS context.

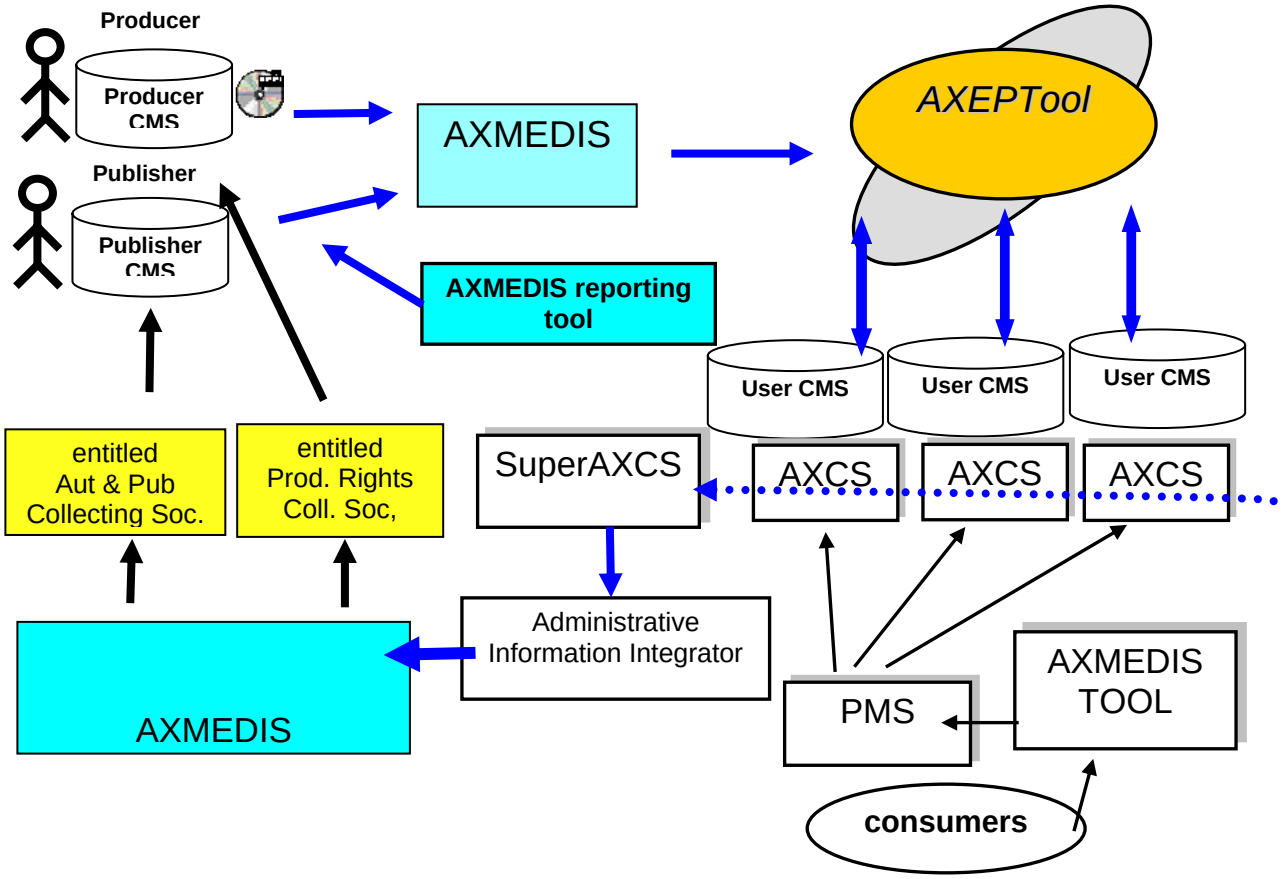
It has to be underline that these scenarios only refer to the use of music whose exploitation rights are granted to the original publisher and the producer. Their content, once governed as an AXMEDIS object, are ready to be exploited within the AXEPTOOL and distributed accordingly to the DRM and license terms provided. All actions (uses) or events performed on these AXMEDIS objects are recorded in the AXCS of each user and then reported, along with other relevant data, to the super AXCS. The super AXCS tool will then interact with the AXMEDIS reporting tool and with the Administrative Information Integrator that will respectively report relevant information into the right owners' database and into the database of the entitled collecting societies. It has to be underline that the link between AXMEDIS tools and the collecting societies should be implemented by taking into account new tools and network developed by collecting societies themselves such as the FastTrack project. This project aims at realizing a global interconnected network of databases on

musical and audiovisual works, rights owners, contracts and data on sound recording to support diary operations of the societies involved such as identification of works and distribution of royalties

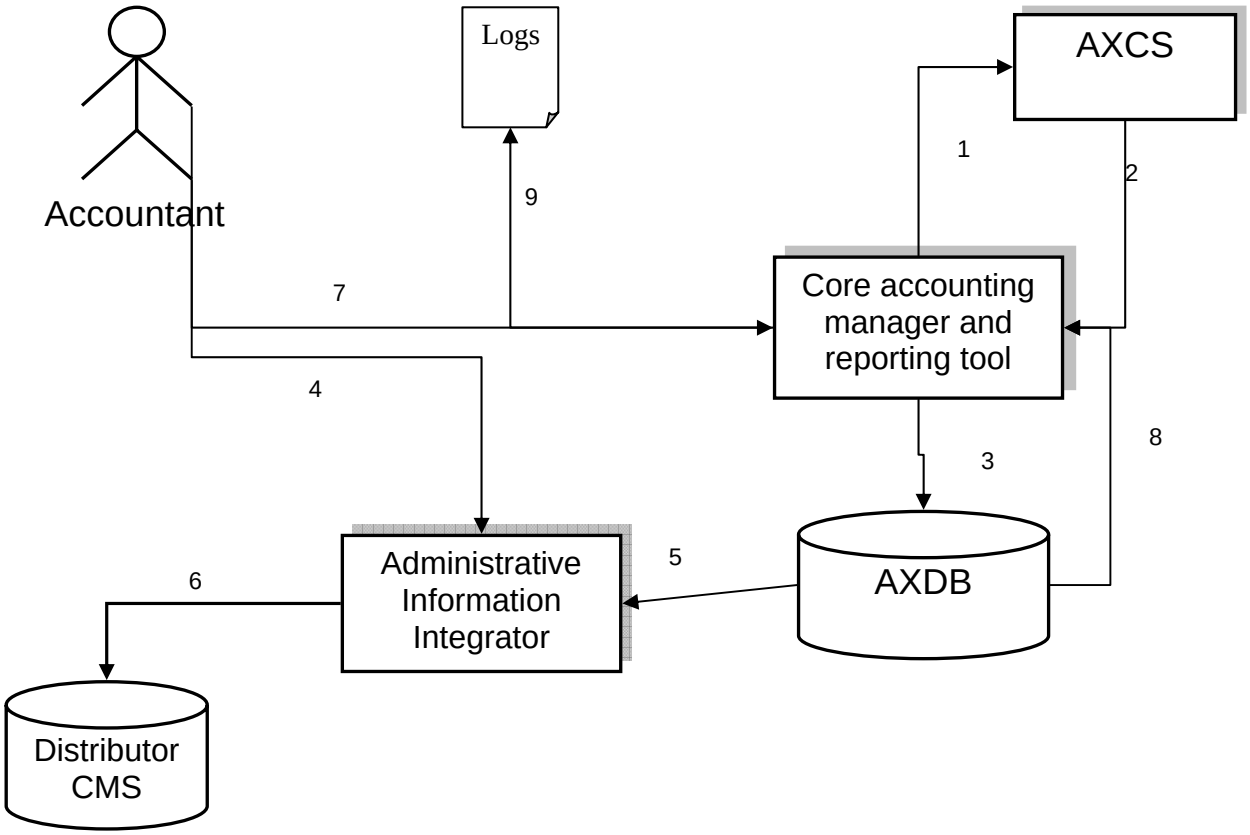
Independently of the ways and methodologies rights are granted to users (e.g. compulsory license, individual license) the Administrative Information integrator tool should provide Collecting societies with data needed to check, verify and monitor the use of the AXMEDIS objects in conformity with the rights granted by the relevant license and with information necessary to identify right owners including identification standard codes already developed (such as the ISRC) as well as those under development.
An AXMEDIS object will involve a multiplicity of rights owners (there could be many just in one musical work) and many different collecting societies (such as public performance rights societies, mechanical rights societies, producers rights societies, performers rights societies etc). Its multi distribution channels will allow multiple reproductions, transmissions and retransmissions until it reaches the end user/consumer.
Due to this issue and to the complexity of the different rules that govern the collection and distribution of royalties for the exploitation of multimedia contents and compounded objects in the digital environment, the control of the correct use of the rights granted and the consequent collection of the royalties due become a complex task and one of the main concern of right holders community.




Producer
Producer CMS
AXMEDIS
AXEPTool
AXMEDIS reporting tool
Producer local Collecting Society
User CMS
User CMS
User CMS
SuperAXCS
AXCS
AXCS
AXCS
User local Producers rights Collecting Society
Administrative Information Integrator
Neighbouring rights
Public use of sound recordings
PMS
AXMEDIS TOOL
consumers


Producer
Producer CMS
Publisher
Publisher CMS
AXMEDIS
AXEPTool
AXMEDIS reporting tool
User CMS
User CMS
User CMS
entitled Aut & Pub Collecting Soc.
entitled Prod. Rights Coll. Soc,
SuperAXCS
AXCS
AXCS
AXCS
Administrative Information Integrator
AXMEDIS
PMS
AXMEDIS TOOL
consumers

4.3.6 Relationships between CAMART and AII



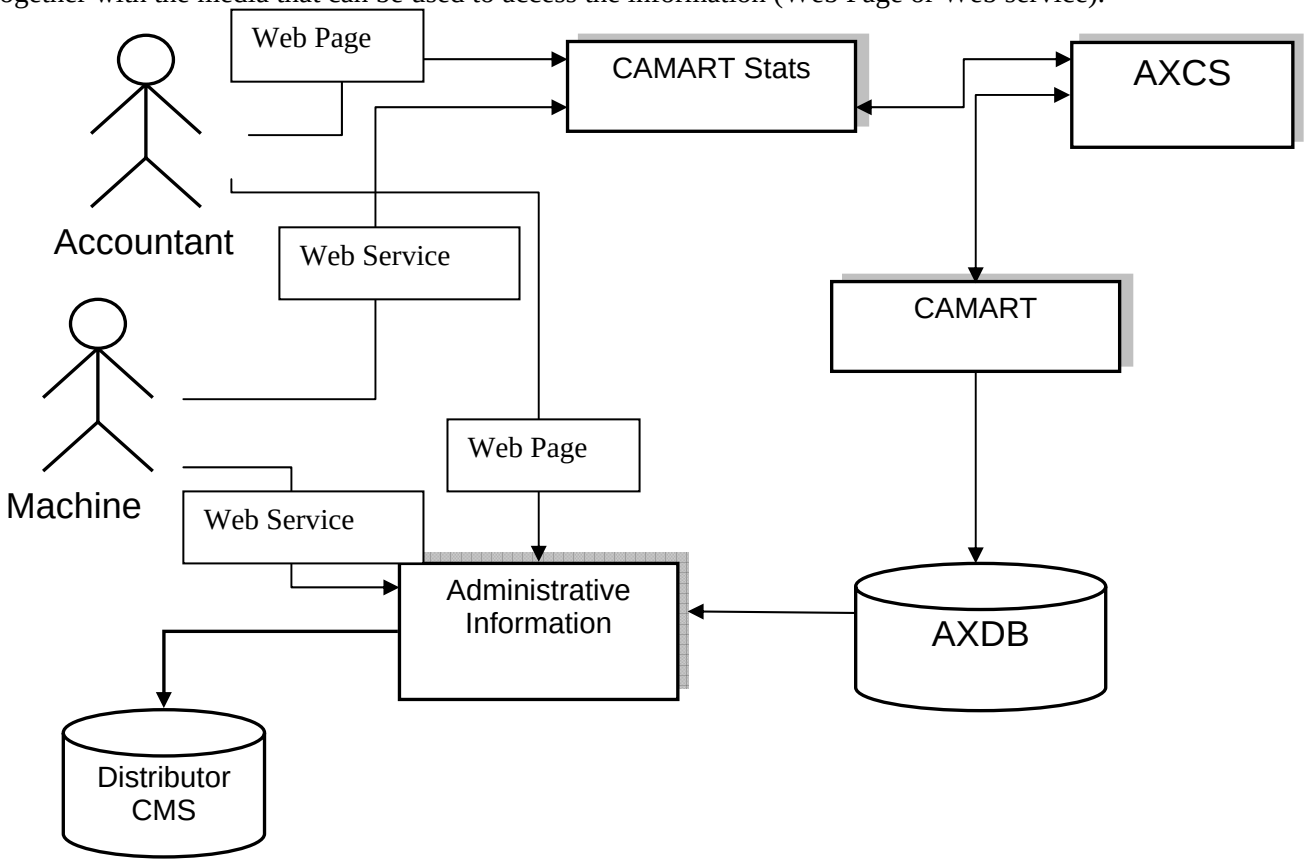
1. CAMART contact AXCS web service for having fresh logs
2. AXCS give back to CAMART the Logs
3. CAMART store logs on AXDB
4. Accountant configure AII to put export logs toward CMS
5. AII get logs from AXDB
6. AII export logs to CMS
7. Accountant can optionally query directly the CAMART for logs filtered by AXOID, AXUID, timestamp etc
8. In that case CAMART will get the log from AXDB
9. CAMART will generate a raw XML report of Logs

5 System Specification

In this section an update of what stated on the basis of the addition and improvement realized in the prototype in DE3-1-2-2-15 is reported.

5.1 General Architecture

In the following diagram the relationship among users and tools of the AXMEDIS platform is reported together with the media that can be used to access the information (Web Page or Web service).



5.2 Module or Executable Tool Core Accounting Manager and Reporting Tools (CAMART)

5.2.1 General Description of the CAMART Module

The role of Core Accounting manager and Reporting Tool (CAMART) is strictly bound with database for logs (provided by AXCS) since it has to collect information regarding the B2B activities and B2C actions. AXCS will not store forever its logs and therefore it is necessary for CAMART to gather time by time such logs and store locally in the AXMEDIS database. Such information will be collected on scheduled time interval and CAMART will act as a client of the AXCS Reporting Web Service.
AXMEDIS system is scalable and therefore we have to deal with the fact that some installation can have AXDB, AXCS and other supporting tools on different machines, while others can be less distributed due to a lesser need for speed or storage capacity.
The core accounting manager is a sort of Client side of the bridge between the AXDB and the AXCS databases in order to allow AXCS to be independent by the database. The server side in the AXCS is the Web Service: AXMEDIS Reporting Web Service. The CAMART can be interpreted as a part of the AXMEDIS Database Interface, since is the part of the system that allows writing data related to Action-Logs into the AXMEDIS DB.

5.2.2 Core Accounting Manager and Reporting Tools interface toward the user

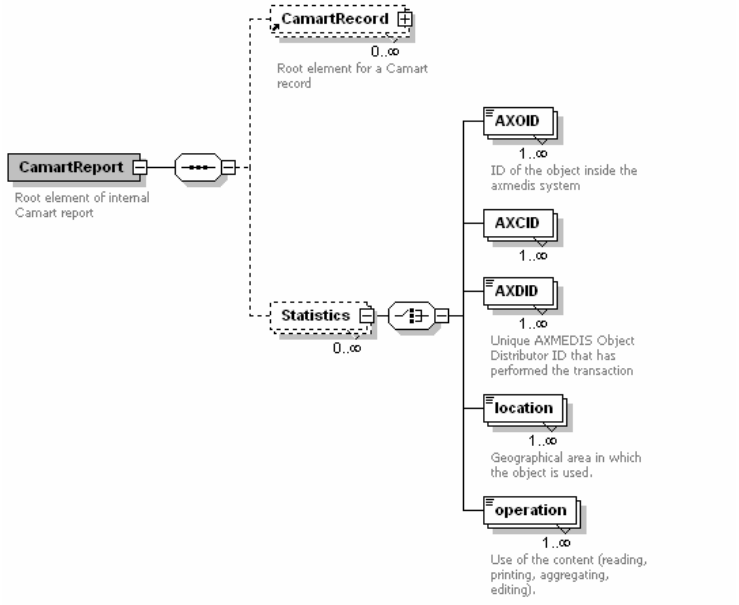
The actor working on the CAMART user interface can be any administrative and management user that has interest in making queries and browsing the information related to the usage of AXMEDIS objects. For example:

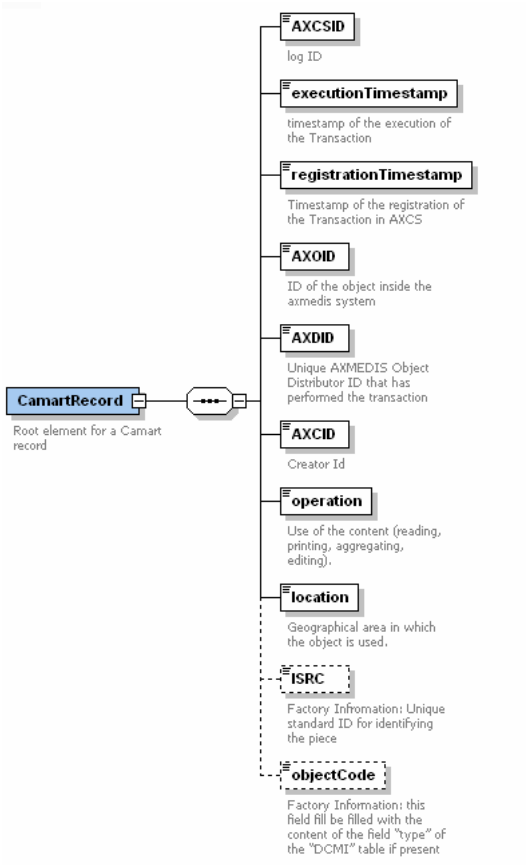
- A Distributor could be interested in seeing the list of Action Logs related to a given second distributor, integrator, etc.
- An Integrator could be interested in seeing the list of Action Logs related to a given AXMEDIS Object, etc.
- A Distributor could be interested in seeing the list of second level distributors that have exploited some specific AXMEDIS object, etc.
- A Distributor could be interested to see how many transactions have been registered on its AXMEDIS objects in the last two months.

These examples can be used by the Actor (account manager) to extract the information and move it into the Administrative Database of the CMS by means of the AXMEDIS tool called Administrative Information Integrator.

The general role of CAMART with respect to the end user that uses it is to provide a web interface for making reporting directly onto their systems. Reporting queries are always executed on the AXMEDIS Database Interface for collecting information that are contained in the local database that in turn had been gathered from the AXCS. In this way the actor sees only the data for which is authorized;

The user interface, is web based in order to cover the needs of interoperability, usability and maintainability that are expected from AXMEDIS in the large sense.
The format that is provided to the user is generated according to the XML schema reported below in graphical shape:





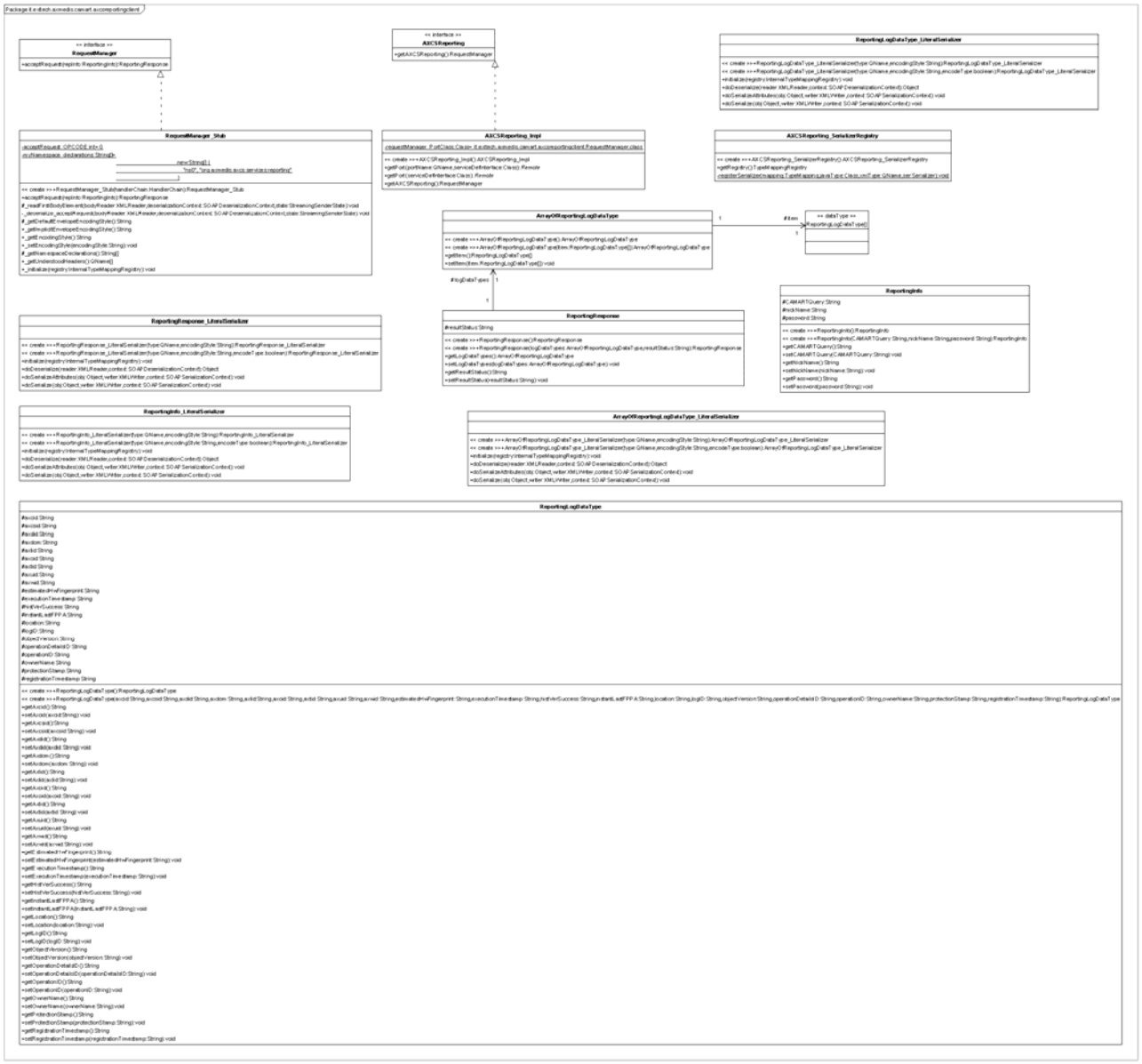
5.2.3 CAMART interface with AXCS

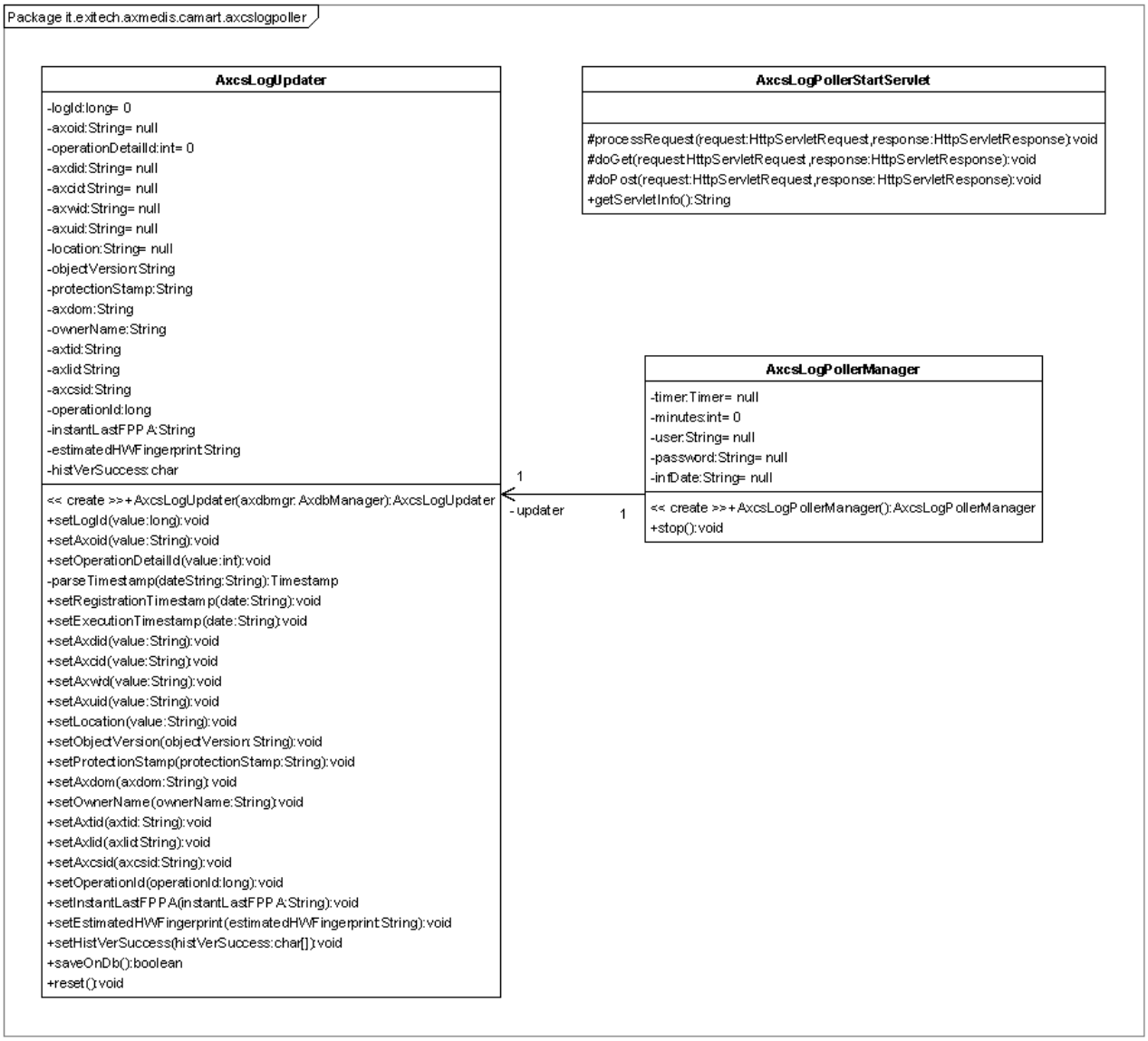
CAMART needs to get fresh logs from AXCS and therefore it is a client of the web service interface that AXCS will expose to have access to log information.
In this section it is defined the interface in order to specify in terms of WSDL this communication protocol that will be a synchronous polling from CAMART to AXCS.
AXCS will have to implement two web services and CAMART will be a client for them, reading at predefined time interval the logs from AXCS and storing them in the local AXDB related to Logs for the reporting part and reading on demand for the statistical part.
The reporting process is completely automatic and is managed by a CAMART daemon that will periodically extract from AXCS all the logs for which the user is entitled and store them in the AXDB.

The web services implemented by AXCS/SuperAXCS are described in details in DE 3.1.2.2.13 and therefore no need to explain them in this document.

5.2.4 CAMART Module Design in terms of Classes

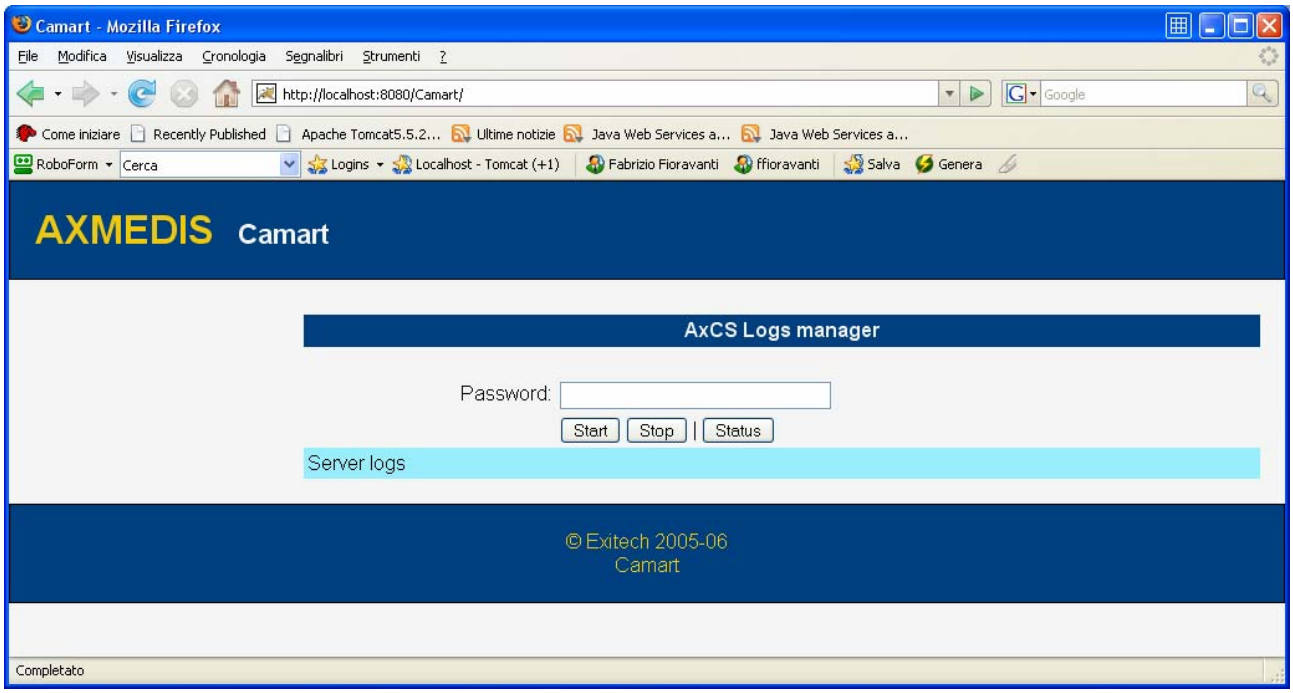
A detailed view of the module in terms of classes is reported in the following pictures:



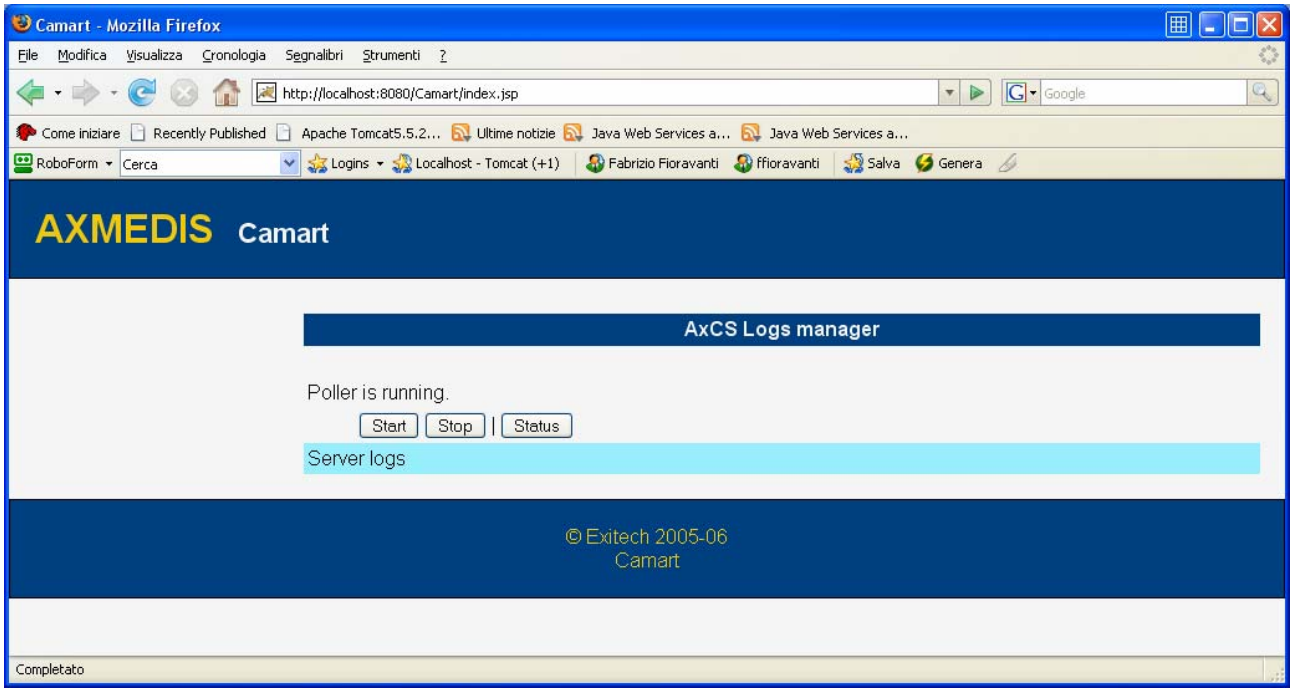


5.2.5 CAMART Prototype description

This module is comprised of a daemon that collects info from AXCS and stores logs locally on AXDB for future reuse from AII.
The interface of this daemon is very simple and is reported below:



After the authentication we have the daemon interface



The prototype now automatically starts when the service starts and can be stopped on demand.

This thread runs in background and collects locally the remote logs provided by AXCS, showing also information on the logs inserted in the last poll with the amount of logs that were already present.

5.2.6 Technical and Installation information

To deploy the Camart Service, a PC with Java2 1.5 Runtime environment and Apache Tomcat 5.5.20 is required. The service will be distributed as a WAR file to be deployed in %TOMCAT_ROOT%/webapps.

5.2.7 Integration and compilation issues

Since Java and Apache Tomcat technologies, combined with W3C standard protocol (HTTP) and mark-up language (XHTML), were used, there is not any known issue related to interoperability and integration in different context.

5.2.8 Configuration Parameters

Config parameter	Possible values
Camart.war\WEB-INF\classes\axdb.properties	
axdbUrl: URI of the axdb for the jdbc	jdbc:mysql://localhost/axdbv4?jdbcCompliantTruncation=false
axdbUser: user for the connection to the AXDB. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	axdbuser (usual user)
axdbPwd: password for the axdbUser for accessing database. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	mkzamk (usual pwd)
axdbMapping: file that contains the mapping for AXDB query engines **(not used in this project)**	C\:\\Inetpub\\wwwroot\\axmedis\\metadata-mapper.xml (usually it is there)
axdbDataSource: connection pool to be used with Tomcat (usually not to be changed unless you change also contex.xml). Used only if axdbManagerClass =it.exitech.axmedis.axdb.DataSourceAxdbManager	jdbc/listenerResource
axdbManagerClass: single ton class to be used for connecting to the AXDB. If it.exitech.axmedis.axdb.DataSourceAxdbManager is used please change also Camart.war/META-INF/context.xml	it.exitech.axmedis.axdb.DataSourceAxdbManager (recommended) it.exitech.axmedis.axdb.MysqlAxdbManager (for debug only)
initialConnections: number of connections in the pool. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	10 (suggested value)
increment: number of connection incremented when the pool is exhausted. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	5 (suggested value)
Camart.war\WEB-INF\context.xml	
file that controls the DataSource to be created in Tomcat. Please change the parameters according to your database configuration	<?xml version="1.0" encoding="UTF-8"?> <Context docBase="Camart.war" path="/Camart"> <Resource auth="Container" driverClassName="org.gjt.mm.mysql.Driver" maxActive="8" maxIdle="4" name="jdbc/listenerResource" password="mkzamk" type="javax.sql.DataSource" url="jdbc:mysql://localhost/axdbv4?jdbcCompliantTruncation=false" username="axdbuser"/> </Context>
Camart.war\WEB-INF\classes\axcs.properties	
logpollingtimeoutmins: time interval in minutes between two downloads from AXCS webservice	2
loguser: user entitled to download logs	mario
logpassword: password of the user entitled to download logs	xxxxx
endpoint: endpoint for the statistic WS **(not used in this project)**	http://localhost:8080/AXCSStatistics/services/Statistics
axcslogendpoint: endpoint of the AXCS Reporting Web Service	http://flauto.dsi.unifi.it:8080/AXCSReporting/services/Reporting

5.2.9 Errors reported and that may occur

Error code	Description and rationales
AccountException	Happens if user has not the rights to access the service, or if the HTTP session was terminated by inactivity time-out. AIIServlet should gently-fail upon this.

5.3 Administrative Information Integrator (AII)

5.3.1 General Description of the Module

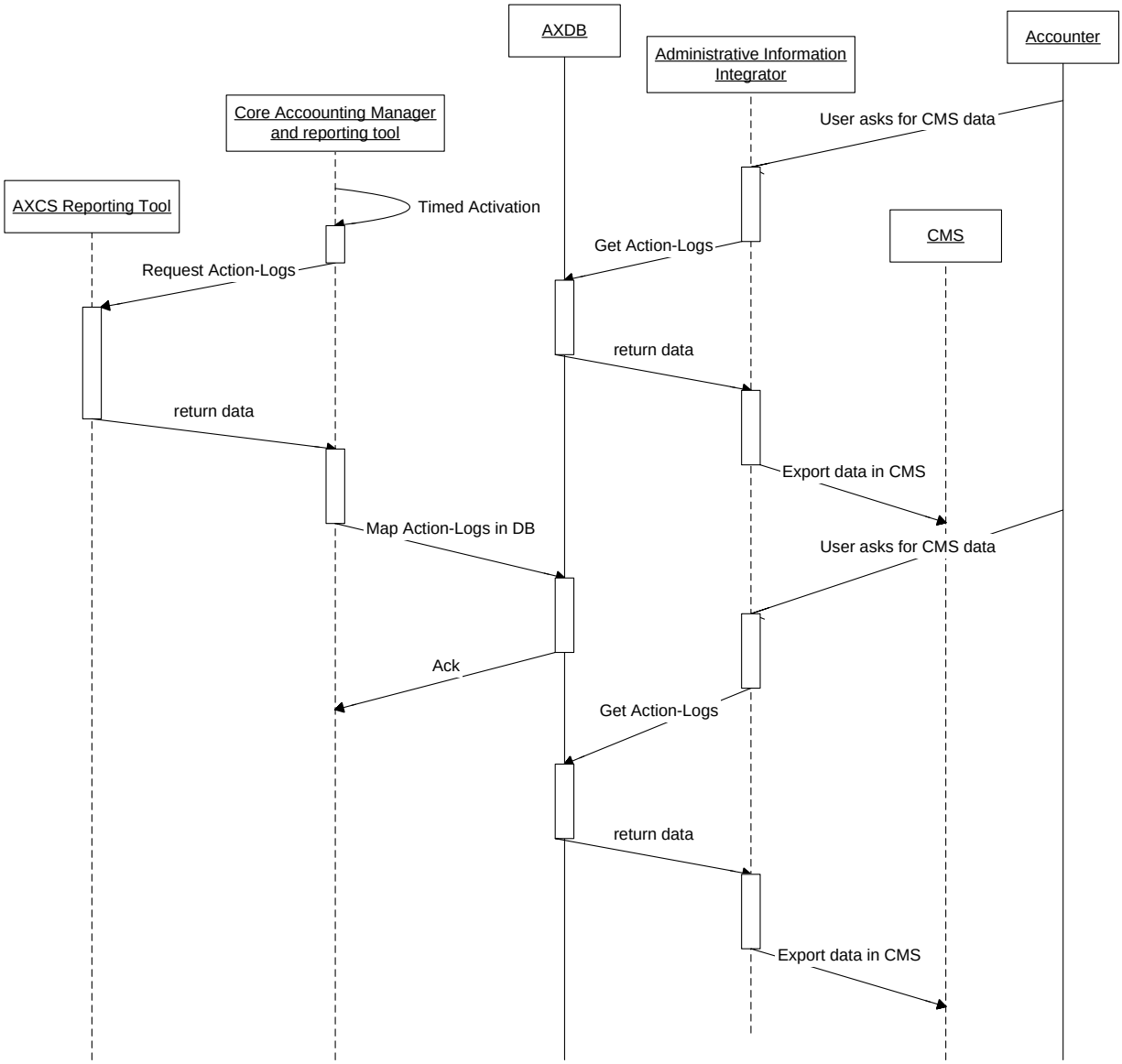
Administrative Information integrator is a critical part of the AXMEDIS system since it is the real bridge between the AXMEDIS world and the world of company's CMS and CRM for taking in account administrative and legal aspects (such reclaim for payment not done and so on).
This component has also a double face since it can operate in a dual manner: used for polling information from AXMEDIS system when needed by distributor for example, or used for pushing information in the CMS as soon as they are available for example in the case of collecting societies.
The AII can also completely remotely managed by a Web Service that offers all the functionalities guaranteed by the web application.
The operating mode is determined by accounting people during the installation/configuration of the system when it will be established whose fields have to be exported from the DB to the CMS and the frequency of exporting. When a frequency is set, the Administrative Information Integrator will work in push mode, pushing information in the CMS import area, otherwise it operates in polling mode by starting the update in the CMS by a link to a web page.

The principal specification arises, as always, from requirements. Administrative Information Integrator has to:

- ***interface with different CMS technologies***: this means that administrative information integrator has to export its reports in a portable format such as an XML that will be defined in the detailed specification. The generated XML can then be parsed and transformed by the way of standard mechanisms such as XSL transformation;
- ***store administrative information into the Content Provider database***: this operation will be possible in two ways, or by preparing a file to be put in the import area on the content provider database or by interfacing with the services offered by the CMS of the Content provider by standard mechanisms such as web services or other remote interface available to the Administrative Information Integrator;
- ***communicate with the AXDB to get administrative information related to a specific Content Provider***: this is the minimum security requirement that have to be established in order to be sure to distribute information to entitled persons only.
- ***guarantee privacy of sensitive data via protection mechanisms***: apart from what has already been stated, if a network connection is necessary to transfer data and the CMS of the Content Provider permits a connection on a secure channel, then the data will be sent encrypted.

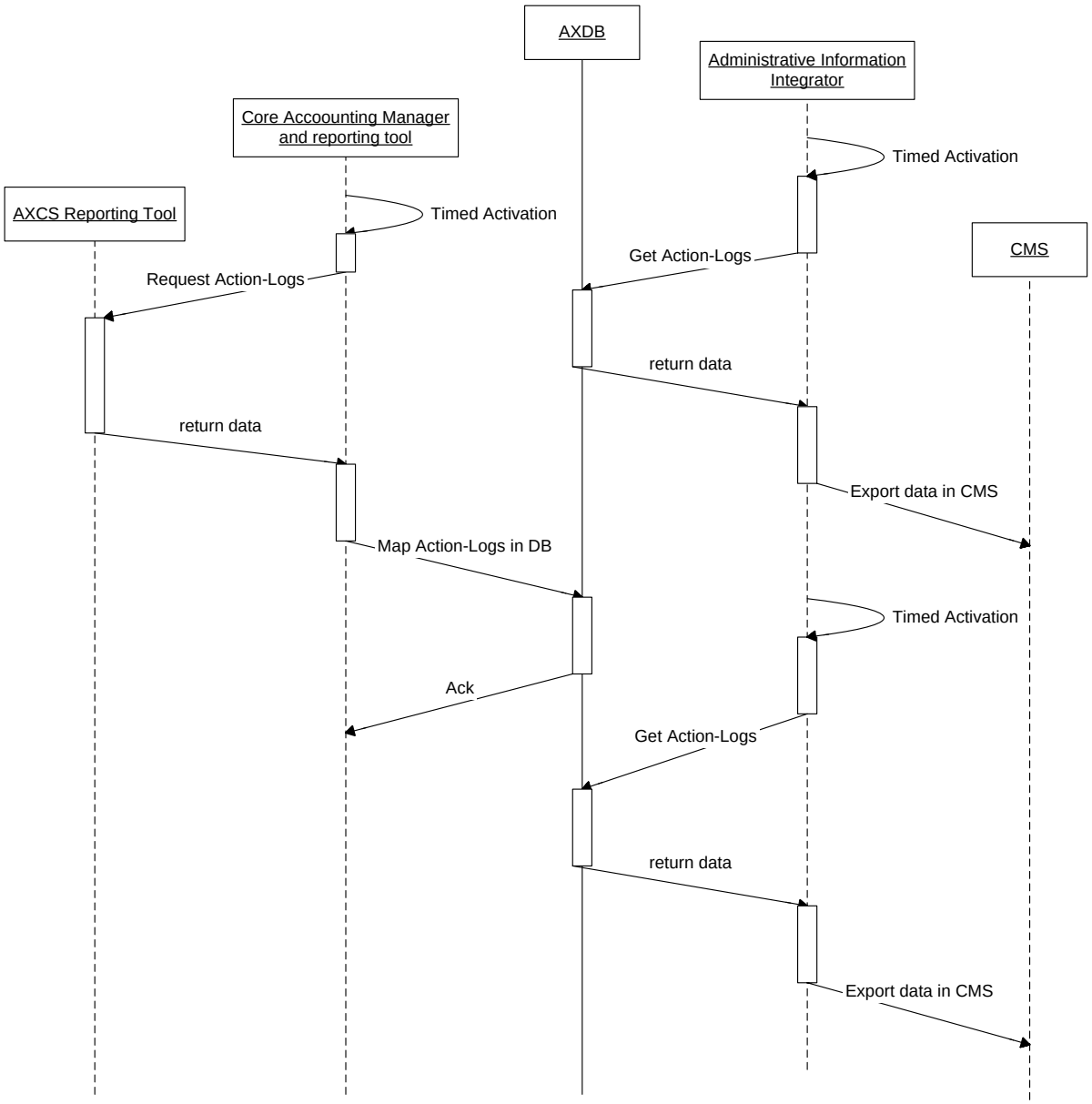
5.3.2 Administrative Information Integrator in polling mode

This is the operational mode when an automatic update of the data in the CMS is not set. In this mode the Accountant connects to a web page and issue a simple command to the Administrative Information Integrator that visualizes in the web page or exports in the CMS Import area the information that have not been already exported according to the format defined during the installation and configuration process.
The process that happens in the Accounting Area when the Administrative Information Integrator is in polling mode can be modeled by the following diagram.

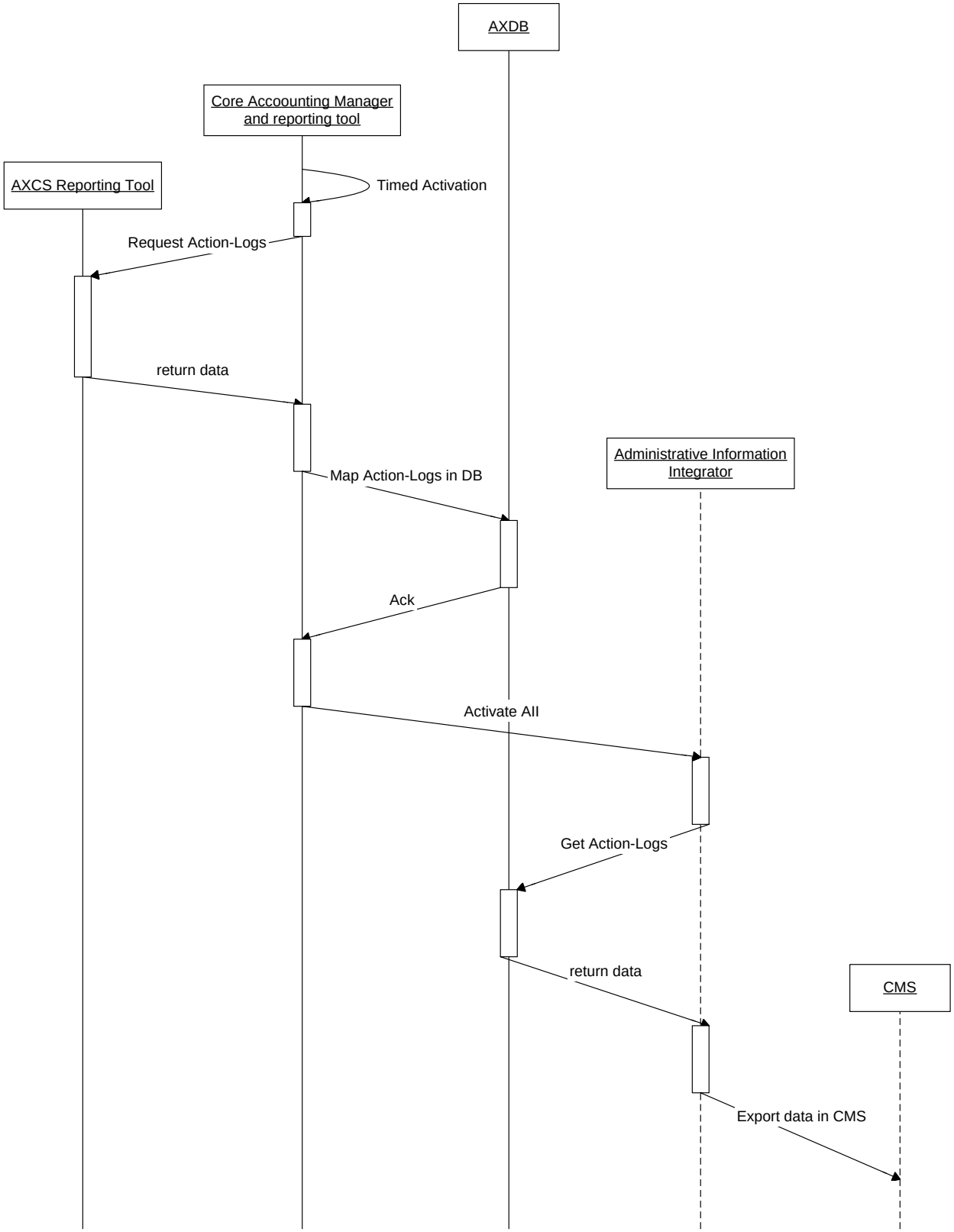


5.3.3 Administrative Information Integrator in push mode

All the times that during the configuration of the Administrative Information Integrator the timed insertion of administrative information has been selected, the Administrative Information Integrator is enabled to put in the import area of the CMS all the information that the CMS accountant has selected. The updating of the CMS should be also synchronized by a trigger sent by Core Accounting Manager and reporting tool.
In the following diagram the timed activation is reported, by which the Administrative Information Integrator and the CAMART operates asynchronously on a timed basis.



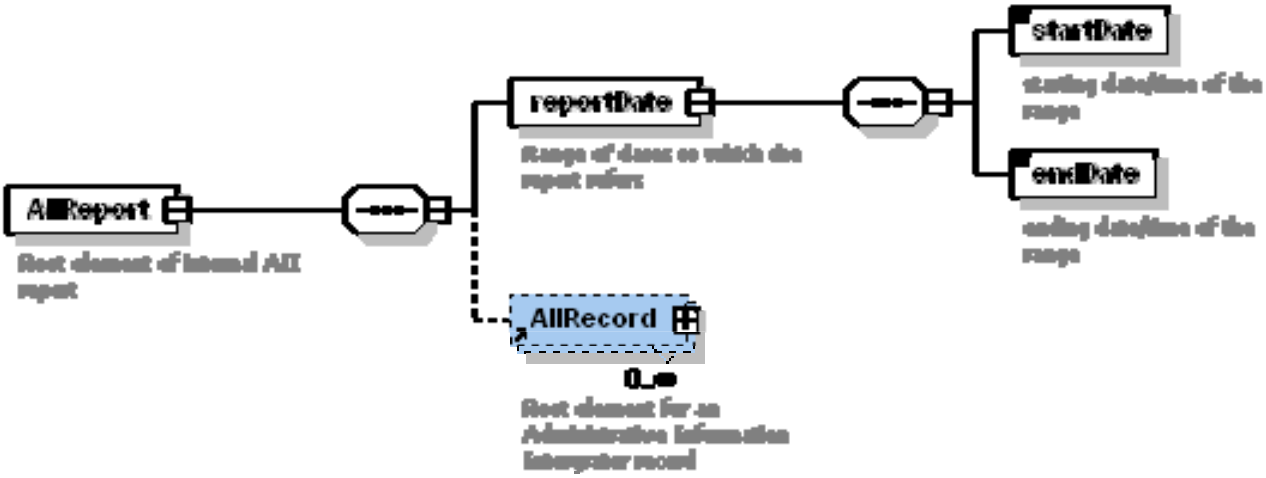
In the following diagram the timed activation is reported, by which the Administrative Information Integrator is triggered by the CAMART.

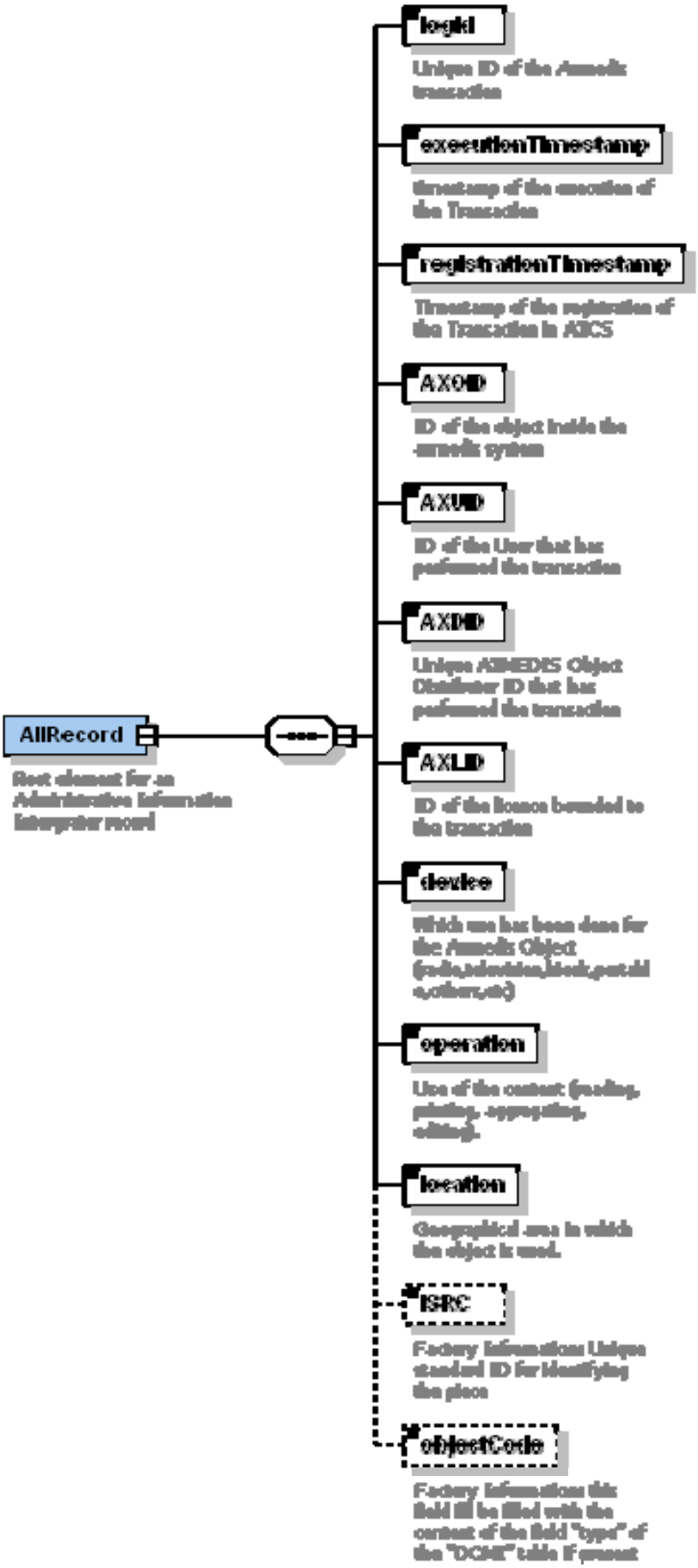

AXDB
Core Accoounting Manager and reporting tool
AXCS Reporting Tool
Timed Activation
Request Action-Logs
return data
Administrative Information Integrator
Map Action-Logs in DB
Ack
Activate AII
Get Action-Logs
return data
CMS
Export data in CMS

5.3.4 Administrative Information Integrator and CAMART integration

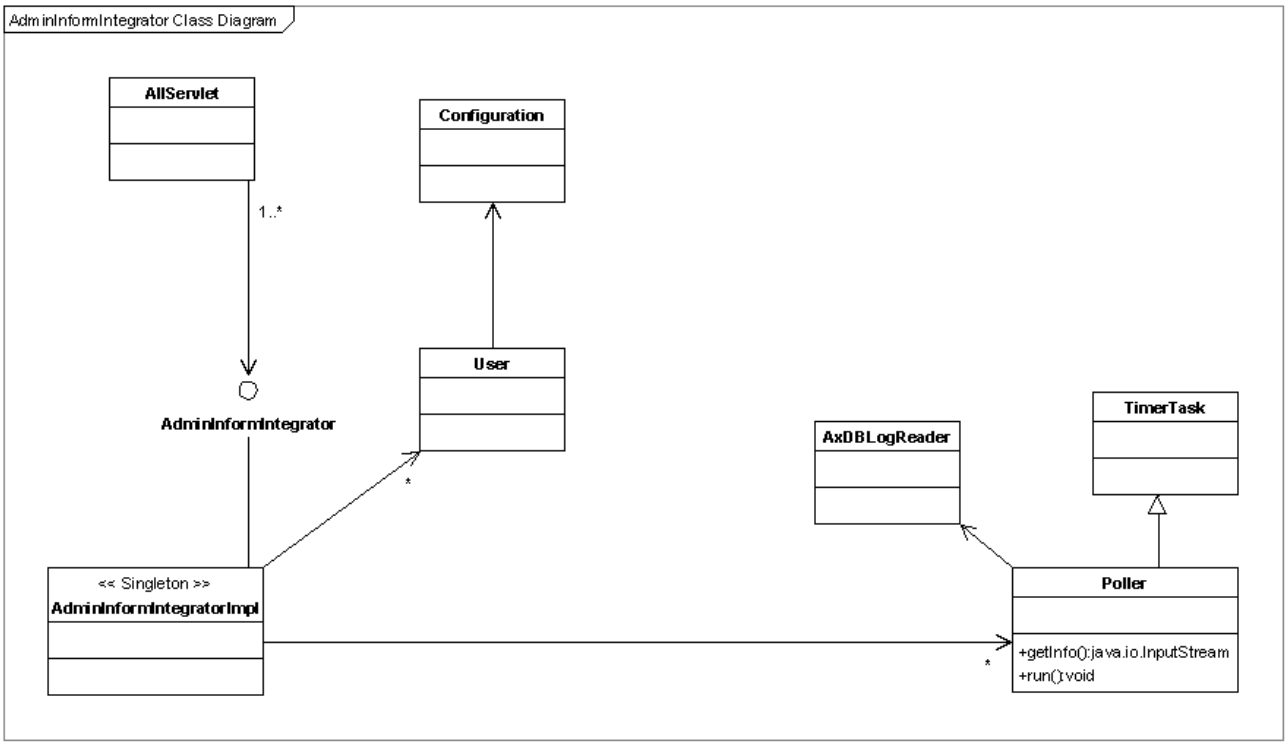
It is necessary to point out how CAMART and AII are related in order to have the right tool doing the right work.
CAMART is the only interface toward the AXCS, while AII will read the Logs that are already in the AXDB; CAMART has the duty to put log on AXDB with a predefined scheduled period. CAMART will offer to the user an interface for getting logs that are on AXDB independently of the CMS exporting.

In this section an XML schema for the general format of the AII record file is defined.
For each partner, on the basis of the requirements shown in DE 9.1.1, an XSL has been created to be applied to the XML in order to obtain a document in the user target format. This process can be repeated for each new partner of affiliated that joins AXMEDIS if the general XML file is not a supported format.
In the XML file, the Business Information will not appear and the factory operation will be considered optional, since it is not known if they can be provided or not.
The general schema is reported below in graphical form, and in Section 8 in the xsd format..




logId
Unique ID of the Axmedis transaction
executionTimestamp
timestamp of the execution of the Transaction
registrationTimestamp
Timestamp of the registration of the Transaction in AXCS
AXOID
ID of the object inside the axmedis system
AXUID
ID of the User that has performed the transaction
AXDID
Unique AXMEDIS Object Distributor ID that has performed the transaction
AXLID
ID of the licence bounded to the transaction
device
Which one has been done for the Axmedis Object (radio,television,block,portal s,others,etc)
operation
Use of the content (reading, printing, copying, editing).
location
Geographical area in which the object is used.
ISRC
Factory Information Unique standard ID for identifying the piece
objectCode
Factory Information this field ill be filled with the content of the field "type" of the "DCMI" table if present
AllRecord
Root element for an Administrative Information Interpreter record

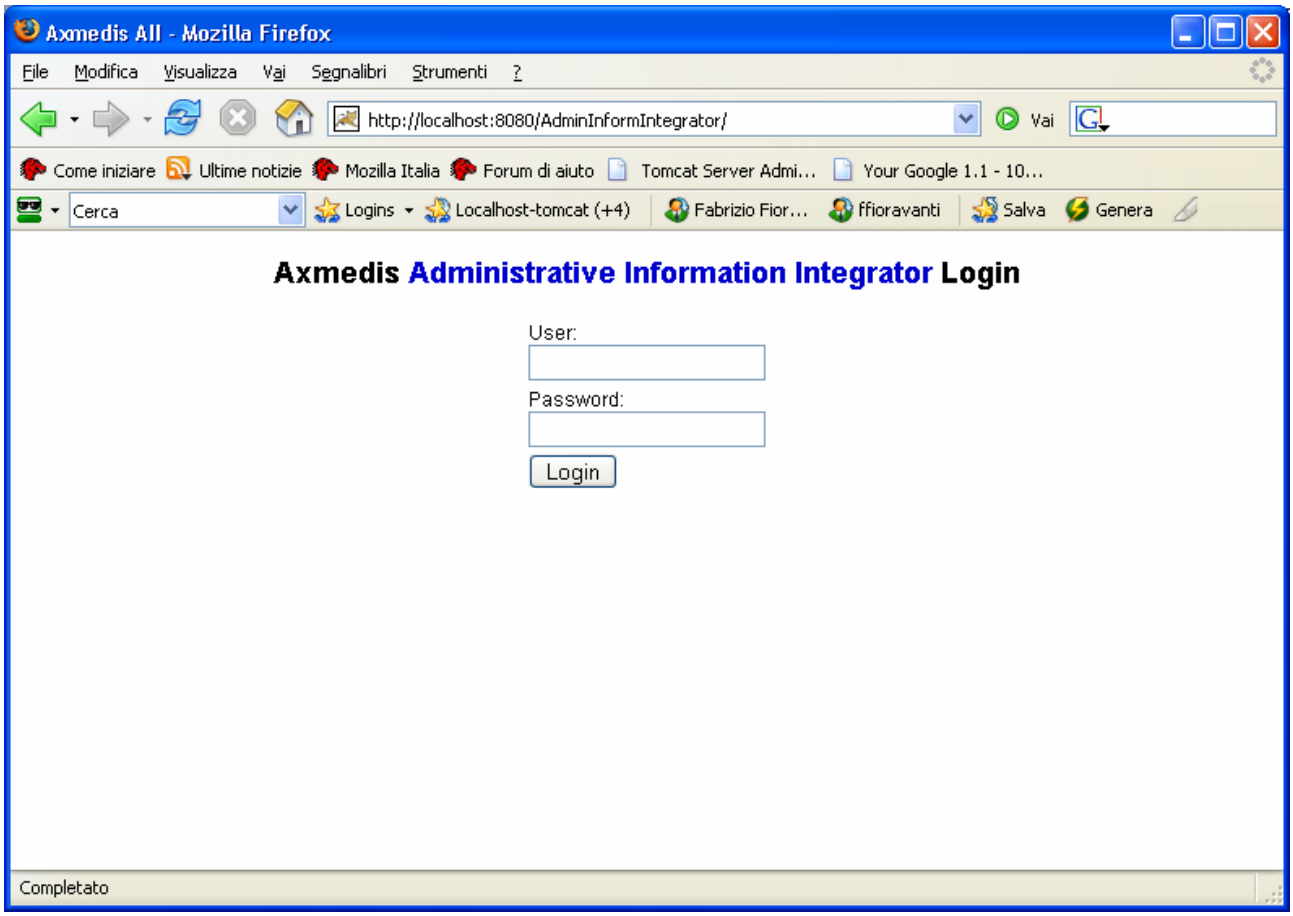
5.3.5 Module Design in terms of Classes



- AdminInformIntegrator: it is the main interface of the service, the business logic is implemented in AdminInformIntegratorImpl;
- AdminInformIntegratorImpl is the controller class of Administrative Information Integrator. It keeps track of the users and the various asynchronous Poller tasks;
- Poller is a class wrapping a polling task, which can be use synchronously or asynchronously. It works synchronously when Administrative Information Integrator is in polling mode, and asynchronously when the service is in pushing mode. In the last case the class is used as a TimerTask that polls data periodically and sends them back to controller class, which publish them at the exporting path selected by user;
- AxDBLogReader is the class in charge to read data from AXDB;
- User is the class that keeps track of all info about logged user, from user ID to configuration for exporting/formatting the AXDB results.

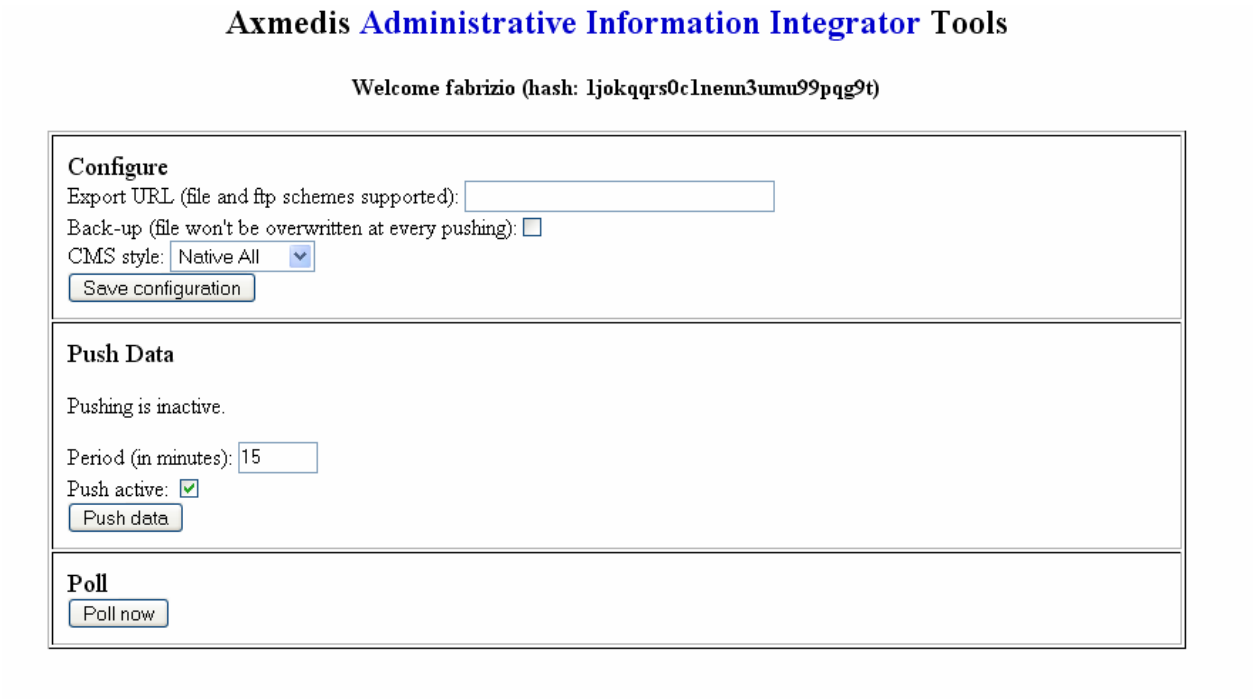
5.3.6 All Prototype User Interface description

Regarding the Administrative Information Integrator seen from the user perspective, the authentication screen is the first page that appears:



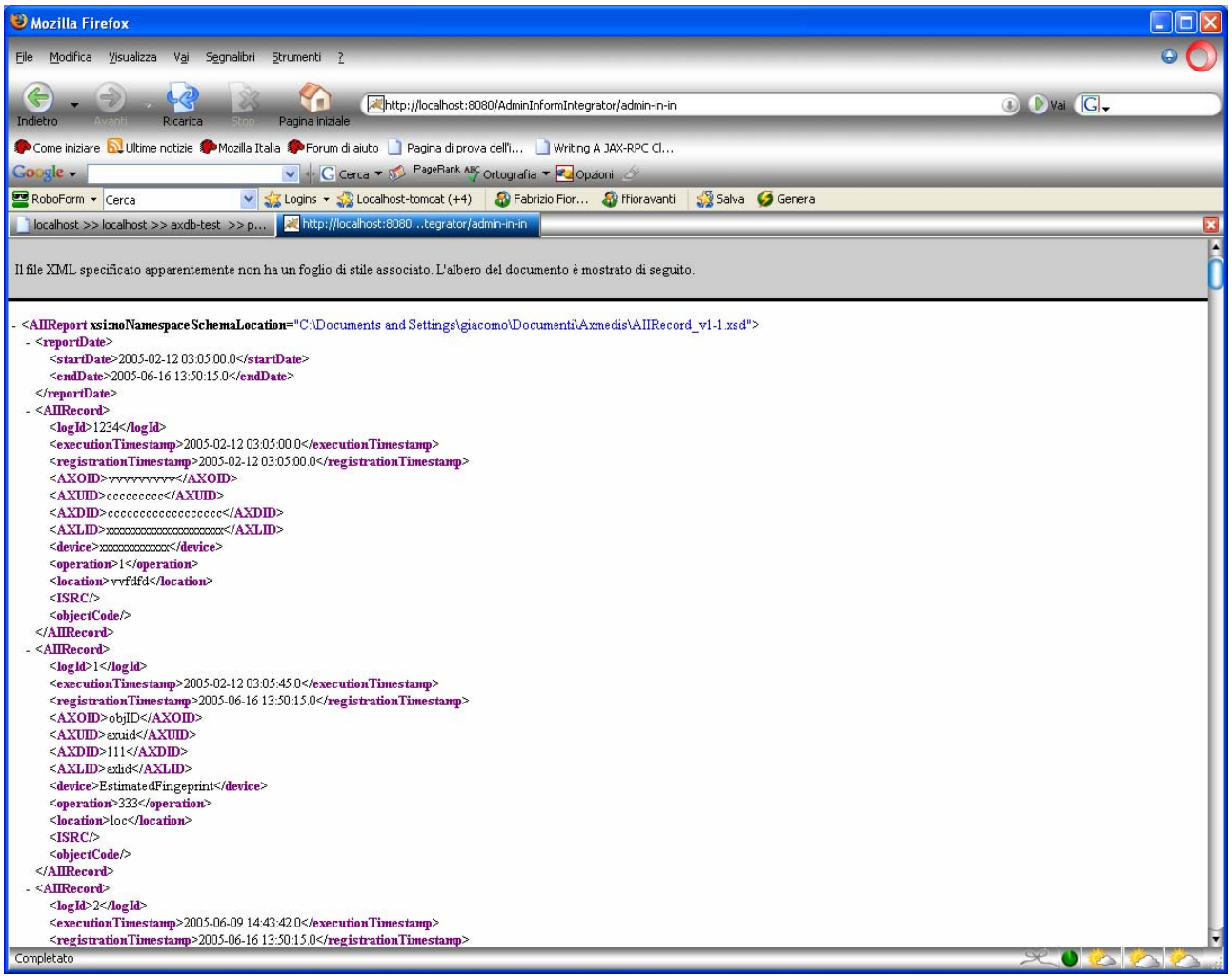
and the User after a login with authentication and authorization phase is bring to the following User Interface that allows to:

- Configure the AII for the logged user, giving him the chance to select a CMS style and the FTP URL where the logs will be transferred after each data pushing
- Configure the time intervals in minutes between two pushing of data and enabling or disabling pushing
- Poll raw data or formatted data in the case the user has already saved a configuration

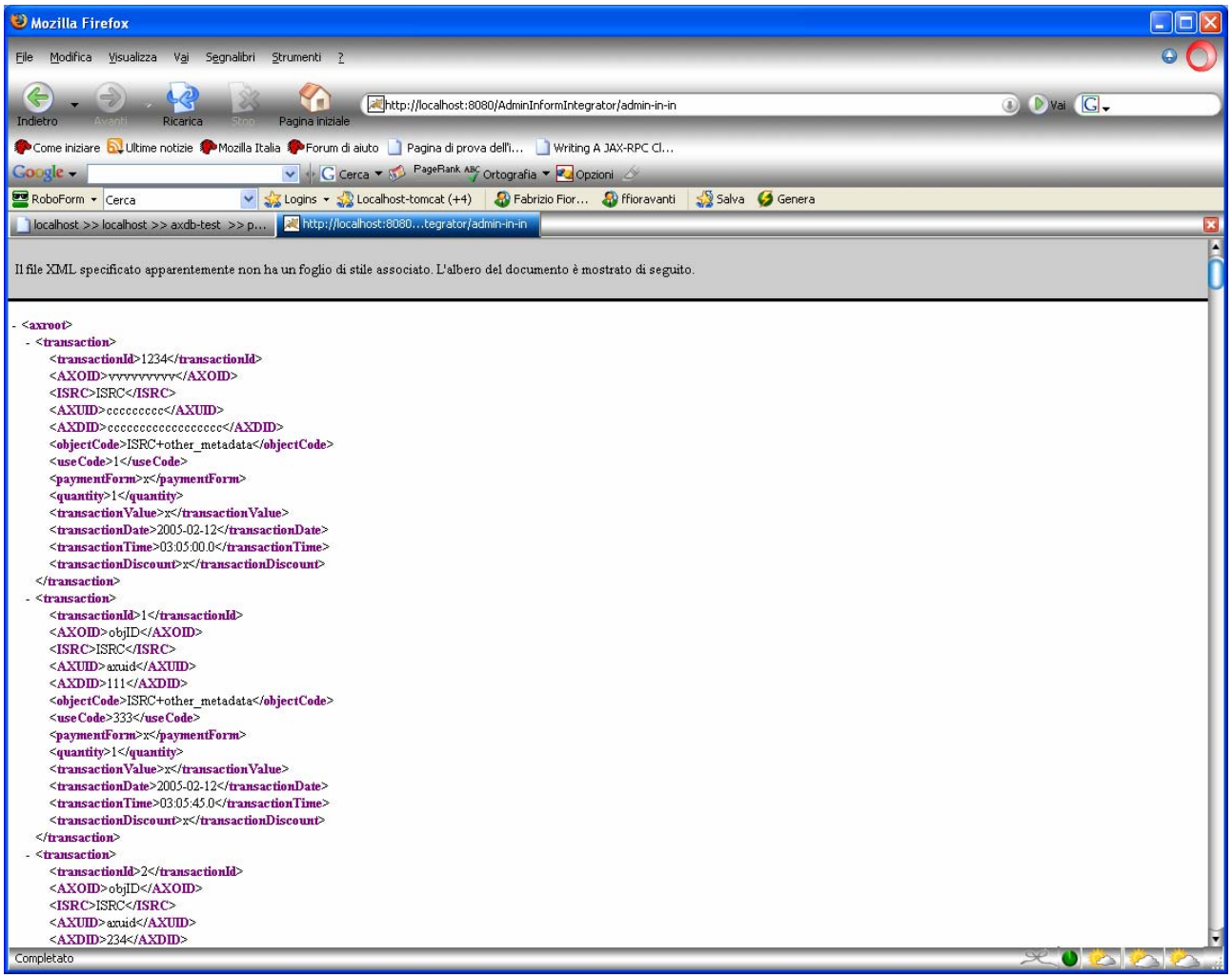


The system will push data in the configured ftp URL at the predefined interval in the format specified in DE 3.1.2.2.15 after getting them from the remote AXCS. In the case of polling the data in XML (formatted or unformatted) will be presented on the screen allowing the user to manually save the file for his own purposes and allowing him to use it immediately. The system tracks the time at which a log has been requested in order to avoid multiple gathering of the same data.

If the user has not chosen the configuration the general format is displayed, if the Poll now button is pressed:



While after the configuration of the format will appear:



5.3.7 AII Web Service Interface

All the AII operation shown above can be also performed by using a Web Service that allows a full control of the application behaviour allowing an automated configuration or polling without human interaction.

The WSDL is reported below:

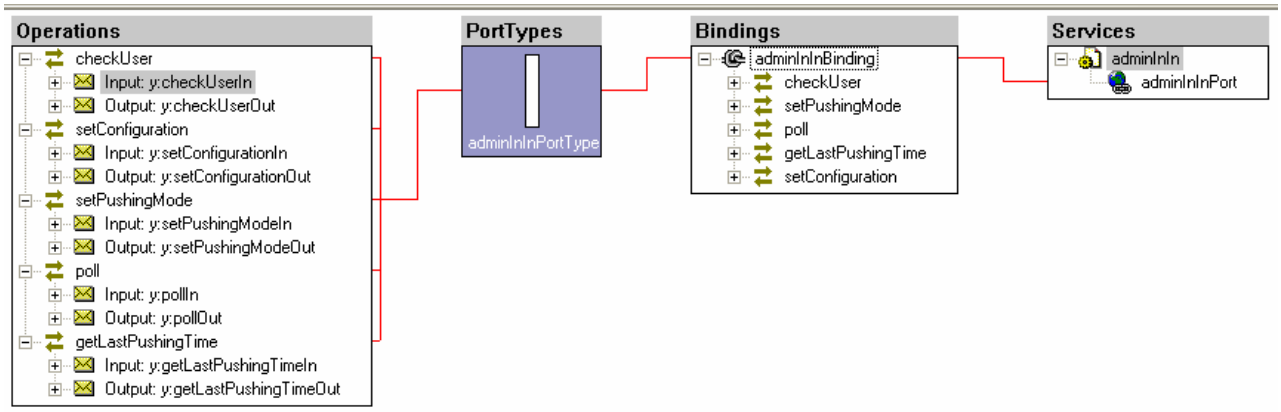
```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 U (http://www.altova.com) by Giacomo (WORK) -->
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:y="http://www.axmedis.org/adminInIn" xmlns:ns="http://www.axmedis.org/adminInIn/in-out.xsd"
targetNamespace="http://www.axmedis.org/adminInIn">
        <types>
                <xs:schema targetNamespace="http://www.axmedis.org/adminInIn/in-out.xsd">
                        <xs:element name="checkUserParams">
                                <xs:complexType>
                                        <xs:sequence>
                                                <xs:element type="xs:string" name="user"/>
                                                <xs:element type="xs:string" name="password"/>
                                        </xs:sequence>
                                </xs:complexType>
                        </xs:element>
                        <xs:element type="ns:userHashType" name="userHash"/>
                        <xs:element name="setConfigurationParams">
                                <xs:complexType>
                                        <xs:sequence>
```

```
                                        <xs:element ref="ns:userHash"/>
                                        <xs:element name="configuration">
                                                <xs:complexType>
                                                        <xs:sequence>
                                                                <xs:element name="cmsStyle">
                                                                        <xs:simpleType>
                                                                                <xs:restriction
base="xs:string">
                                                                                        <xs:enumeration
value="internal"/>
                                                                                        <xs:enumeration
value="AFI"/>
                                                                                        <xs:enumeration
value="ANSC"/>
                                                                                        <xs:enumeration
value="ILABS"/>
                                                                                        <xs:enumeration
value="EUTELSAT"/>
                                                                                        <xs:enumeration
value="TISCALI"/>
                                                                                        <xs:enumeration
value="SEJER"/>
                                                                                        <xs:enumeration
value="XIM"/>
                                                                                </xs:restriction>
                                                                        </xs:simpleType>
                                                                </xs:element>
                                                                <xs:element type="xs:anyURI" minOccurs="0"
name="exportingPath"/>
                                                                <xs:element type="xs:boolean" minOccurs="0"
name="backUp"/>
                                                        </xs:sequence>
                                                </xs:complexType>
                                        </xs:element>
                                </xs:sequence>
                        </xs:complexType>
                </xs:element>
                <xs:element type="xs:boolean" name="operationSuccessful"/>
                <xs:element name="setPushingModeParams">
                        <xs:complexType>
                                <xs:sequence>
                                        <xs:element ref="ns:userHash"/>
                                        <xs:element type="xs:boolean" name="enabled"/>
                                        <xs:element type="xs:int" name="period"/>
                                </xs:sequence>
                        </xs:complexType>
                </xs:element>
                <xs:element type="ns:userHashType" name="pollParam"/>
                <xs:element name="report">
                        <xs:complexType>
                                <xs:choice>
                                        <xs:element type="xs:string" name="csv"/>
                                        <xs:element type="xs:anyType" name="xml"/>
                                </xs:choice>
                        </xs:complexType>
                </xs:element>
                <xs:element type="ns:userHashType" name="getLastPushingTimeParam"/>
                <xs:element type="xs:int" name="minutes"/>
                <xs:simpleType name="userHashType">
                        <xs:restriction base="xs:string">
                                <xs:length value="32"/>
                        </xs:restriction>
                </xs:simpleType>
        </xs:schema>
    </types>
    <message name="checkUserIn">
        <part name="params" element="ns:checkUserParams"/>
    </message>
    <message name="checkUserOut">
        <part name="ret" element="ns:userHash"/>
    </message>
    <message name="setConfigurationIn">
        <part name="param" element="ns:setConfigurationParams"/>
    </message>
```

```
<message name="setConfigurationOut">
        <part name="ret" element="ns:operationSuccessful"/>
</message>
<message name="setPushingModeIn">
        <part name="param" element="ns:setPushingModeParams"/>
</message>
<message name="setPushingModeOut">
        <part name="ret" element="ns:operationSuccessful"/>
</message>
<message name="pollIn">
        <part name="param" element="ns:pollParam"/>
</message>
<message name="pollOut">
        <part name="ret" element="ns:report"/>
</message>
<message name="getLastPushingTimeIn">
        <part name="param" element="ns:getLastPushingTimeParam"/>
</message>
<message name="getLastPushingTimeOut">
        <part name="ret" element="ns:minutes"/>
</message>
<portType name="adminInInPortType">
        <operation name="checkUser">
                <input message="y:checkUserIn"/>
                <output message="y:checkUserOut"/>
        </operation>
        <operation name="setConfiguration">
                <input message="y:setConfigurationIn"/>
                <output message="y:setConfigurationOut"/>
        </operation>
        <operation name="setPushingMode">
                <input message="y:setPushingModeIn"/>
                <output message="y:setPushingModeOut"/>
        </operation>
        <operation name="poll">
                <input message="y:pollIn"/>
                <output message="y:pollOut"/>
        </operation>
        <operation name="getLastPushingTime">
                <input message="y:getLastPushingTimeIn"/>
                <output message="y:getLastPushingTimeOut"/>
        </operation>
</portType>
<binding name="adminInInBinding" type="y:adminInInPortType">
        <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="checkUser">
                <soap:operation soapAction="http://www.axmedis.org/aii/checkUser" style="document"/>
                <input>
                        <soap:body use="literal"/>
                </input>
                <output>
                        <soap:body use="literal"/>
                </output>
        </operation>
        <operation name="setPushingMode">
                <soap:operation soapAction="http://www.axmedis.org/aii/setPushingMode" style="document"/>
                <input>
                        <soap:body use="literal"/>
                </input>
                <output>
                        <soap:body use="literal"/>
                </output>
        </operation>
        <operation name="poll">
                <soap:operation soapAction="http://www.axmedis.org/aii/poll" style="document"/>
                <input>
                        <soap:body use="literal"/>
                </input>
                <output>
                        <soap:body use="literal"/>
                </output>
        </operation>
        <operation name="getLastPushingTime">
                <soap:operation soapAction="http://www.axmedis.org/aii/getLastPushingTime" style="document"/>
```

```
                        <input>
                                <soap:body use="literal"/>
                        </input>
                        <output>
                                <soap:body use="literal"/>
                        </output>
                </operation>
                <operation name="setConfiguration">
                        <soap:operation soapAction="http://www.axmedis.org/aii/setConfiguration" style="document"/>
                        <input>
                                <soap:body use="literal"/>
                        </input>
                        <output>
                                <soap:body use="literal"/>
                        </output>
                </operation>
        </binding>
        <service name="adminInIn">
                <port name="adminInInPort" binding="y:adminInInBinding">
                        <soap:address location="http://localhost:8080/AdminInformIntegrator/adminInIn"/>
                </port>
        </service>
</definitions>
```

The functionalities can be easily understood also by the means of the following pictures.



It should be noted that before doing any other operation, the checkUser operation has to be performed.

5.3.8 Technical and Installation information

To deploy the Administrative Information Integrator service a PC with Java2 1.5 Runtime environment and Apache Tomcat 5.5.20 is required. The service will be distributed as a WAR file to be deployed in %TOMCAT_ROOT%/webapps.

5.3.9 Draft User Manual

The configuration interface appears after the login that allows to map the user with the chosen configuration. If the user do not select any particular configuration, data will be returned in the neutral format specified in this document, while if the user selects one of the predefined CMS styles the data will be formatted accordingly.
Logs can be extracted in the current time instant with the “Poll Now” button or scheduled every n minutes by the Push section.
The Export URL in the first section is the ftp url where the data formatted according to the CMS style have to be put each time the pushing action is activated.

5.3.10 Integration and compilation issues

Since Java and Apache Tomcat technologies, combined with W3C standard protocol (HTTP) and mark-up language (XHTML), were used, there is not any known issue related to interoperability and integration in different context.

Config parameter	Possible values
Camart.war\WEB-INF\classes\axdb.properties	
axdbUrl: URI of the axdb for the jdbc	jdbc:mysql://localhost/axdbv4?jdbcCompliantTruncation=false
axdbUser: user for the connection to the AXDB. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	axdbuser (usual user)
axdbPwd: password for the axdbUser for accessing database. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	mkzamk (usual pwd)
axdbMapping: file that contains the mapping for AXDB query engines **(not used in this project)**	C\:\\Inetpub\\wwwroot\\axmedis\\metadata-mapper.xml (usually it is there)
axdbDataSource: connection pool to be used with Tomcat (usually not to be changed unless you change also contex.xml). Used only if axdbManagerClass =it.exitech.axmedis.axdb.DataSourceAxdbManager	jdbc/listenerResource
axdbManagerClass: single ton class to be used for connecting to the AXDB. If it.exitech.axmedis.axdb.DataSourceAxdbManager is used please change also Camart.war/META-INF/context.xml	it.exitech.axmedis.axdb.DataSourceAxdbManager (recommended) it.exitech.axmedis.axdb.MysqlAxdbManager (for debug only)
initialConnections: number of connections in the pool. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	10 (suggested value)
increment: number of connection incremented when the pool is exhausted. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	5 (suggested value)
Camart.war\WEB-INF\context.xml	
file that controls the DataSource to be created in Tomcat. Please change the parameters according to your database configuration	<?xml version="1.0" encoding="UTF-8"?> <Context docBase="Camart.war" path="/Camart"> <Resource auth="Container" driverClassName="org.gjt.mm.mysql.Driver" maxActive="8" maxIdle="4" name="jdbc/listenerResource" password="mkzamk" type="javax.sql.DataSource" url="jdbc:mysql://localhost/axdbv4?jdbcCompliantTruncation=false" username="axdbuser"/> </Context>
Camart.war\WEB-INF\classes\aii.properties	
maxPushRequests: initial value for the size of Hash Table of the push requests (resized dynamically)	an integer value, usually set to 32
maxConcurrentUsers: max number of concurrent user on the system	an integer value, usually between 10 and 1000

5.3.11 Errors reported and that may occur

Error code	Description and rationales
AccountException	Happens if user has not the rights to access the service, or if the HTTP session was terminated by inactivity time-out. AIIServlet should gently-fail upon this.

5.4 CAMART for Statistics

5.4.1 Introduction

The role of Core Accounting manager and Reporting Tool (CAMART) for Statistics is strictly bound with database for logs (provided by AXCS) since it has to gather information from AXMEDIS Certifier and Supervisor about Action Log and provide them to the user via web page or web service interface.

The statistics are also accessible also by a Web Service Interface whose WSDL is reported below:

```
<?xml version="1.0" encoding="UTF-8"?><definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:y="http://new.webservice.namespace"
xmlns:cam="http://www.exitech.it/CamartStats.xsd" targetNamespace="http://new.webservice.namespace">
         <types>
                  <xs:schema targetNamespace="http://www.exitech.it/CamartStats.xsd"
elementFormDefault="qualified" attributeFormDefault="unqualified">
                           <!-- version 1.3 (25-Jul-2006)
 JWSDP compatible -->
                           <xs:element name="CamartRecord">
                                    <xs:annotation>
                                             <xs:documentation>Root element for a Camart
record</xs:documentation>
                                    </xs:annotation>
                                    <xs:complexType>
                                             <xs:sequence>
                                                      <xs:element name="AXCSID" type="xs:string">
                                                               <xs:annotation>
                                                                        <xs:documentation>log
ID</xs:documentation>
                                                               </xs:annotation>
                                                      </xs:element>
                                                      <xs:element name="executionTimestamp"
type="xs:dateTime">
                                                               <xs:annotation>

         <xs:documentation>timestamp of the execution of the Transaction</xs:documentation>
                                                               </xs:annotation>
                                                      </xs:element>
                                                      <xs:element name="registrationTimestamp"
type="xs:dateTime">
                                                               <xs:annotation>

         <xs:documentation>Timestamp of the registration of the Transaction in
AXCS</xs:documentation>
                                                               </xs:annotation>
                                                      </xs:element>
                                                      <xs:element name="AXOID">
                                                               <xs:annotation>
                                                                        <xs:documentation>ID of
the object inside the axmedis system</xs:documentation>
                                                               </xs:annotation>
                                                               <xs:simpleType>
                                                                        <xs:restriction
base="xs:string">
                                                                                 <xs:maxLength
value="40"/>
                                                                                 <xs:whiteSpace
value="collapse"/>
                                                                        </xs:restriction>
                                                               </xs:simpleType>
                                                      </xs:element>
                                                      <xs:element name="AXDID">
                                                               <xs:annotation>
```

```
                                                                                <xs:documentation>Unique
AXMEDIS Object Distributor ID that has performed the transaction</xs:documentation>
                                                                        </xs:annotation>
                                                                        <xs:simpleType>
                                                                                <xs:restriction
base="xs:string">
                                                                                        <xs:maxLength
value="40"/>
                                                                                        <xs:whiteSpace
value="collapse"/>
                                                                                </xs:restriction>
                                                                        </xs:simpleType>
                                                                </xs:element>
                                                                <xs:element name="AXCID" type="xs:string">
                                                                        <xs:annotation>

          <xs:documentation>Creator Id</xs:documentation>
                                                                        </xs:annotation>
                                                                </xs:element>
                                                                <xs:element name="operation"
type="xs:string">
                                                                        <xs:annotation>
                                                                                <xs:documentation>Use of
the content (reading, printing, aggregating, editing).</xs:documentation>
                                                                        </xs:annotation>
                                                                </xs:element>
                                                                <xs:element name="location"
type="xs:string">
                                                                        <xs:annotation>

          <xs:documentation>Geographical area in which the object is used.</xs:documentation>
                                                                        </xs:annotation>
                                                                </xs:element>
                                                                <xs:element name="ISRC" type="xs:string"
minOccurs="0">
                                                                        <xs:annotation>

          <xs:documentation>Factory Infromation: Unique standard ID for identifying the
piece</xs:documentation>
                                                                        </xs:annotation>
                                                                </xs:element>
                                                                <xs:element name="objectCode"
type="xs:string" minOccurs="0">
                                                                        <xs:annotation>

          <xs:documentation>Factory Information: this field fill be filled with the content of the
field “type” of the “DCMI” table if present </xs:documentation>
                                                                        </xs:annotation>
                                                                </xs:element>
                                                        </xs:sequence>
                                                </xs:complexType>
                                        </xs:element>
                                        <xs:element name="CamartReport">
                                                <xs:annotation>
                                                        <xs:documentation>Root element of internal Camart
report</xs:documentation>
                                                </xs:annotation>
                                                <xs:complexType>
                                                        <xs:sequence>
                                                                <xs:element ref="cam:CamartRecord"
minOccurs="0" maxOccurs="unbounded"/>
                                                                <xs:element name="Statistics" minOccurs="0"
maxOccurs="unbounded">
                                                                        <xs:complexType>
                                                                                <xs:sequence>
                                                                                        <xs:element
name="StatsItem" maxOccurs="unbounded">

          <xs:complexType>

          <xs:simpleContent>

          <xs:extension base="xs:string">

          <xs:attribute name="count" type="xs:int" use="required"/>
```

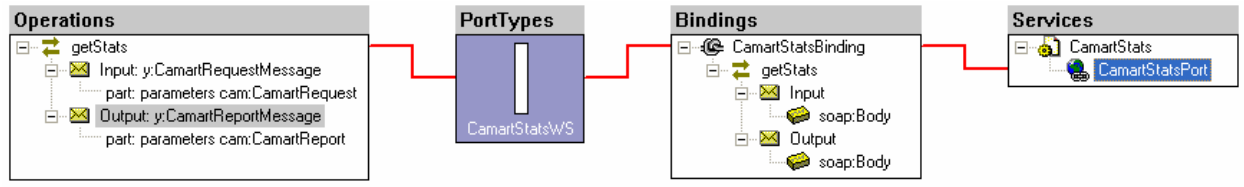
```
         </xs:extension>

         </xs:simpleContent>

         </xs:complexType>
                                                                                  </xs:element>
                                                                        </xs:sequence>
                                                                        <xs:attributeGroup
ref="cam:StatsAttribGroup"/>
                                                                </xs:complexType>
                                                        </xs:element>
                                                </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                                <xs:element name="CamartRequest">
                                        <xs:annotation>
                                                <xs:documentation>Root element of the request used
in WS</xs:documentation>
                                        </xs:annotation>
                                        <xs:complexType>
                                                <xs:sequence>
                                                        <xs:element name="authorization">
                                                                <xs:complexType>
                                                                        <xs:attribute
name="user" type="xs:string" use="required"/>
                                                                        <xs:attribute name="pw"
type="xs:string" use="required"/>
                                                                </xs:complexType>
                                                        </xs:element>
                                                        <xs:element name="date"
type="xs:dateTime"/>
                                                        <xs:element name="AXOID" type="xs:string"
minOccurs="0"/>
                                                        <xs:element name="AXCID" type="xs:string"
minOccurs="0"/>
                                                        <xs:element name="AXDID" type="xs:string"
minOccurs="0"/>
                                                        <xs:element name="AXDOM" type="xs:string"
minOccurs="0"/>
                                                        <xs:element name="AXWID" type="xs:string"
minOccurs="0"/>
                                                        <xs:element name="Chart" minOccurs="0"
maxOccurs="unbounded">
                                                                <xs:complexType>
                                                                        <xs:attributeGroup
ref="cam:StatsAttribGroup"/>
                                                                </xs:complexType>
                                                        </xs:element>
                                                </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                                <xs:attributeGroup name="StatsAttribGroup">
                                        <xs:attribute name="type" use="required">
                                                <xs:simpleType>
                                                        <xs:restriction base="xs:string">
                                                                <xs:enumeration
value="TopChart"/>
                                                                <xs:enumeration
value="BottomChart"/>
                                                        </xs:restriction>
                                                </xs:simpleType>
                                        </xs:attribute>
                                        <xs:attribute name="cardinality" type="xs:int"/>
                                        <xs:attribute name="restrict-by" use="required">
                                                <xs:simpleType>
                                                        <xs:restriction base="xs:string">
                                                                <xs:enumeration value="AXOID"/>
                                                                <xs:enumeration value="AXCID"/>
                                                                <xs:enumeration value="AXDID"/>
                                                                <xs:enumeration
value="location"/>
                                                                <xs:enumeration
value="operation"/>
                                                        </xs:restriction>
```

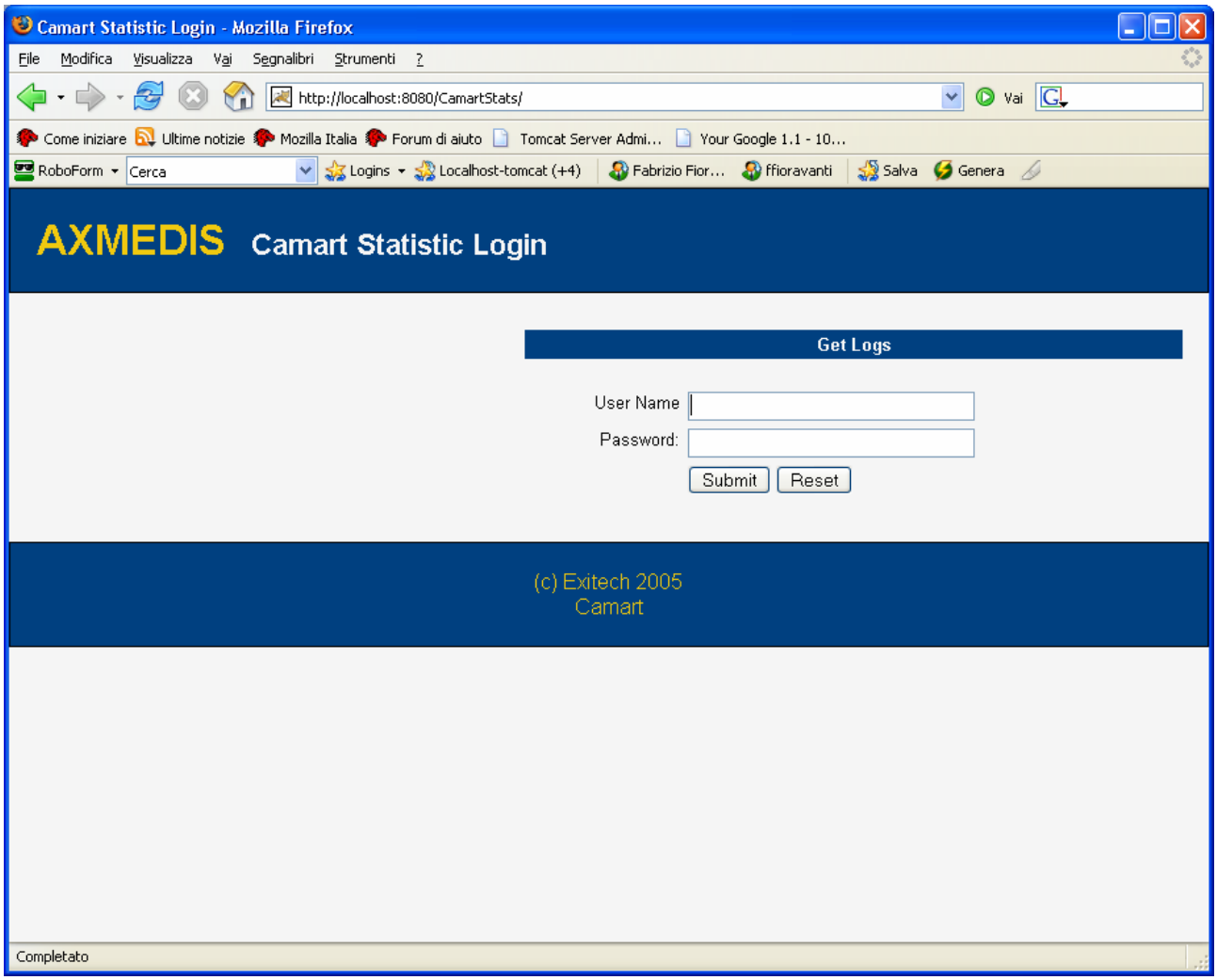
```
                                        </xs:simpleType>
                                </xs:attribute>
                        </xs:attributeGroup>
                </xs:schema>
        </types>
        <message name="CamartRequestMessage">
                <part name="parameters" element="cam:CamartRequest"/>
        </message>
        <message name="CamartReportMessage">
                <part name="parameters" element="cam:CamartReport"/>
        </message>
        <portType name="CamartStatsWS">
                <operation name="getStats">
                        <input message="y:CamartRequestMessage"/>
                        <output message="y:CamartReportMessage"/>
                </operation>
        </portType>
        <binding name="CamartStatsBinding" type="y:CamartStatsWS">
                <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
                <operation name="getStats">
                        <input>
                                <soap:body use="literal"/>
                        </input>
                        <output>
                                <soap:body use="literal"/>
                        </output>
                </operation>
        </binding>
        <service name="CamartStats">
                <port name="CamartStatsPort" binding="y:CamartStatsBinding">
                        <soap:address location="http://localhost:8080/CamartStats/ws"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"/>
                </port>
        </service>
</definitions>
```

A more intuitive view of the functionalities is reported in the following diagram:

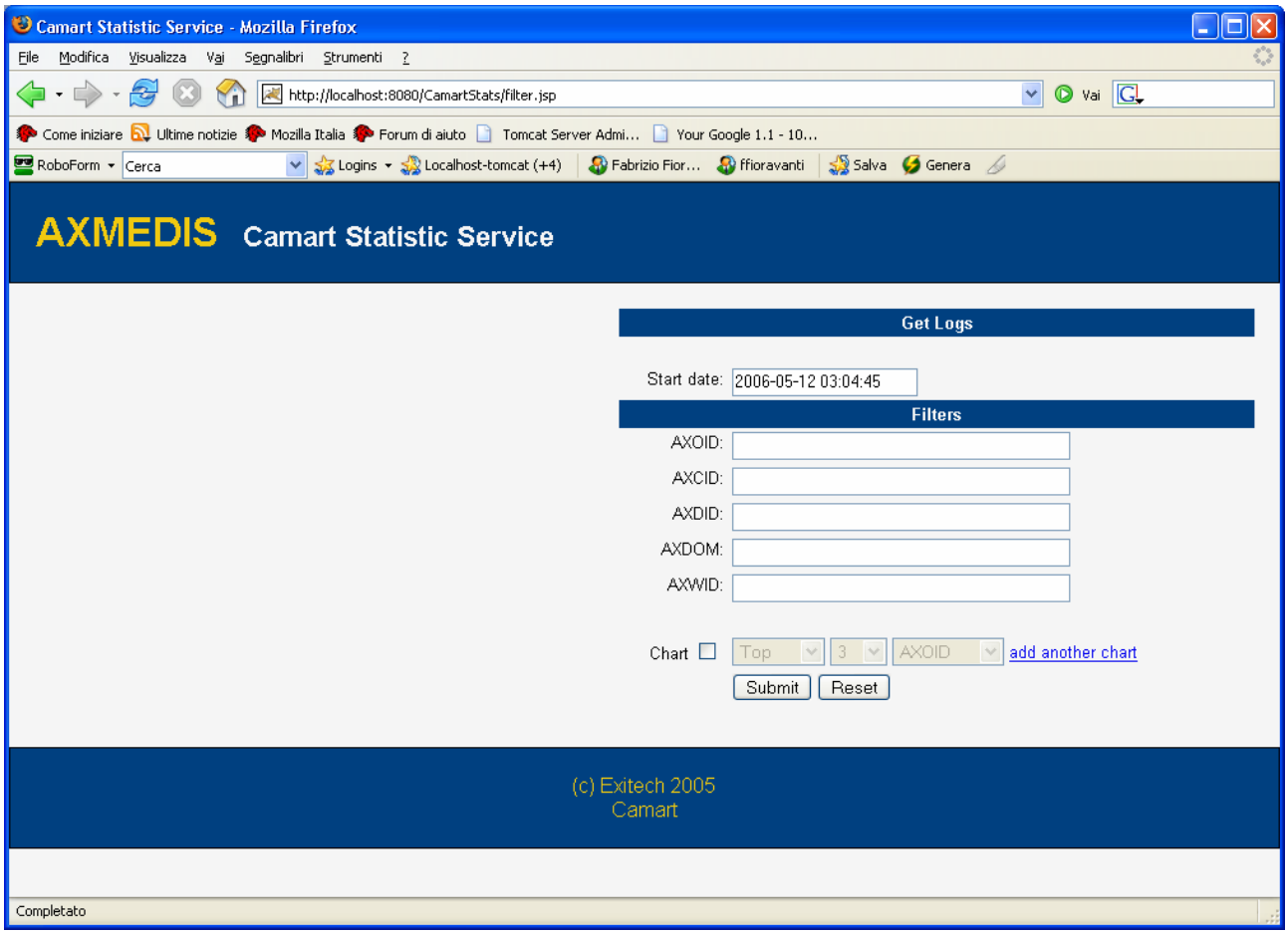


5.4.2 The prototype

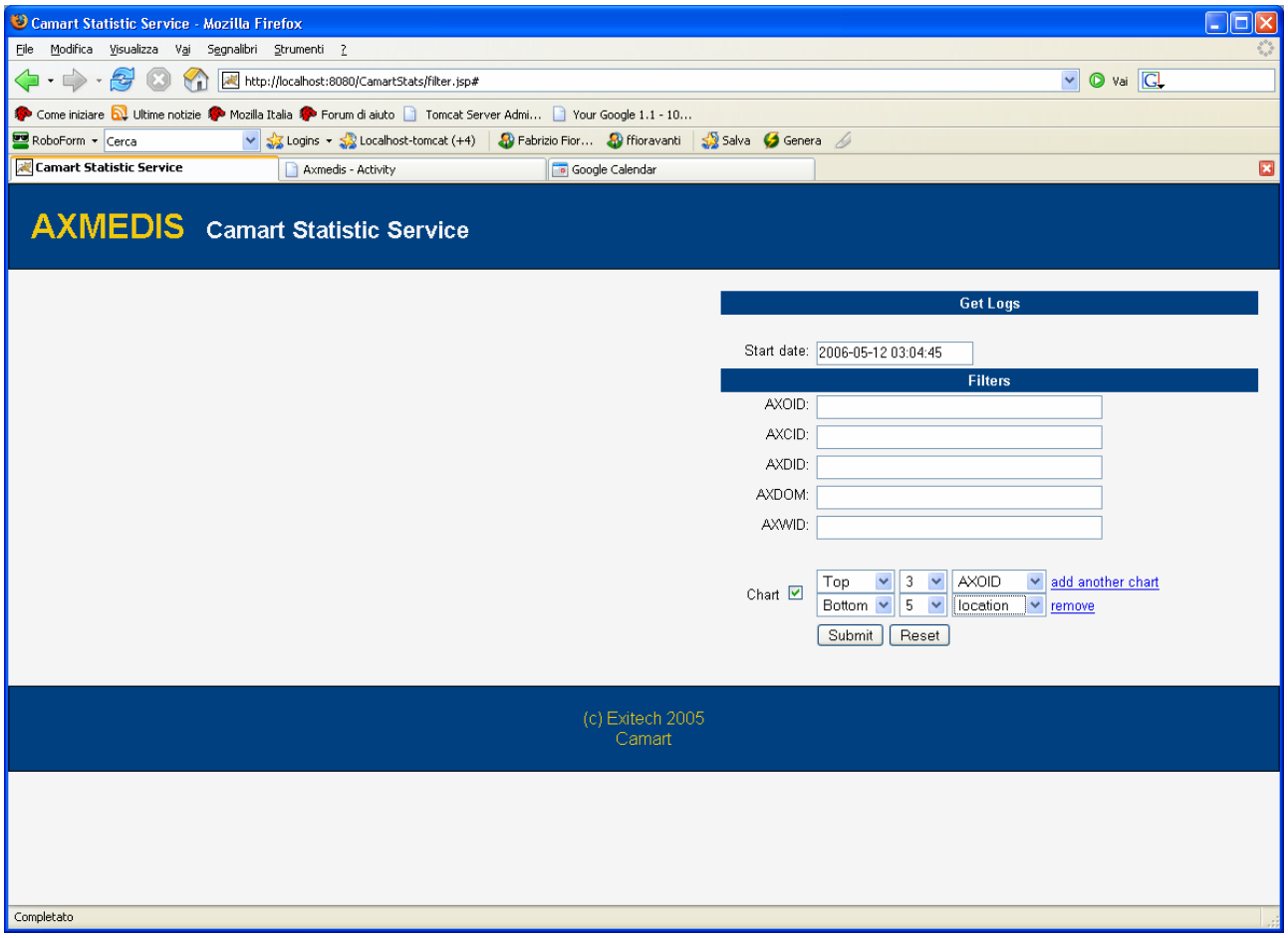
The system is divided mainly in two parts, that are the CAMART for statistical analysis, that in its first prototype is quite raw in the statistics provided and in the operators that can be applied.
The User interface is reported in the following. The first screen is a login screen:



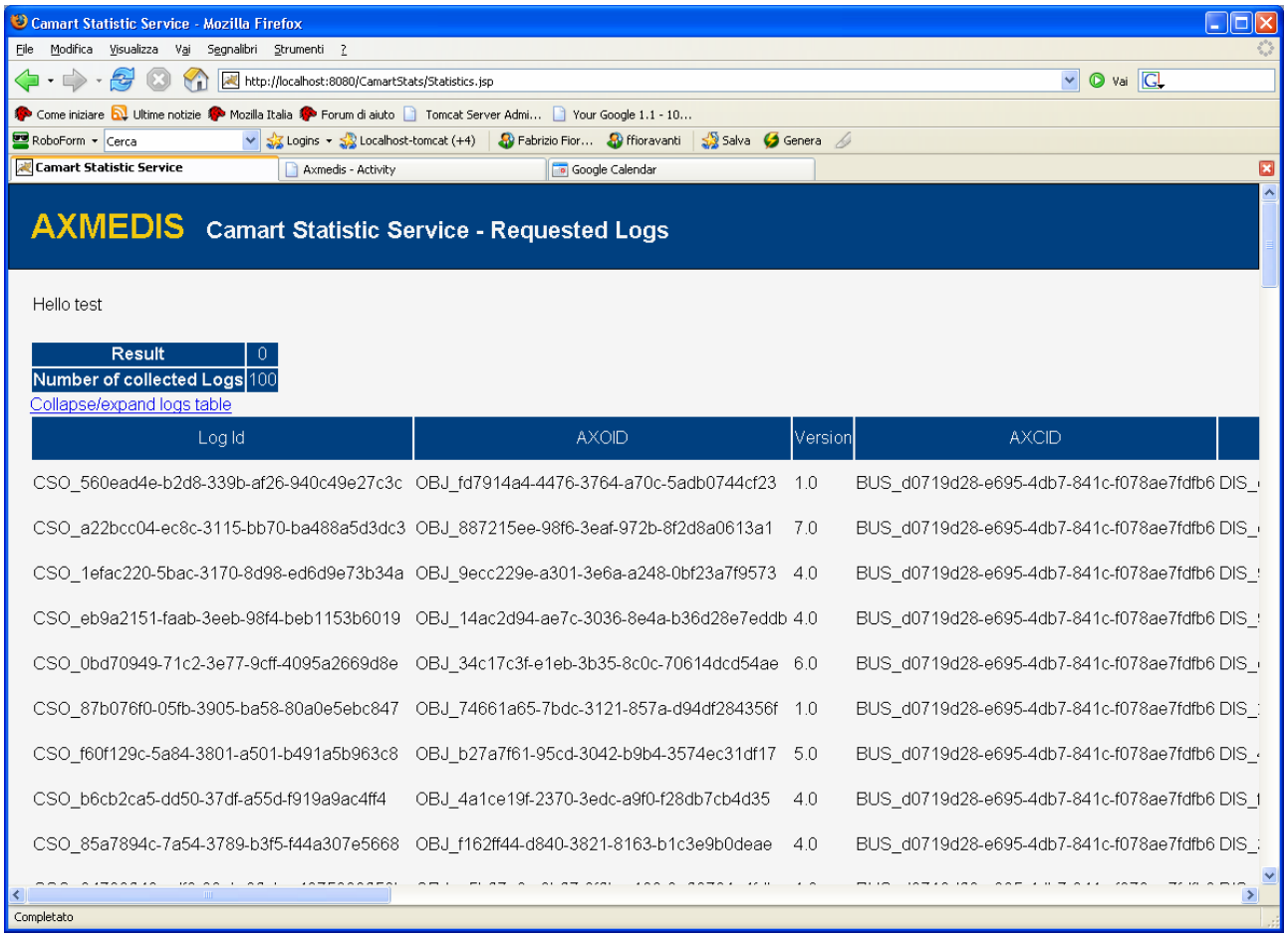
After the login



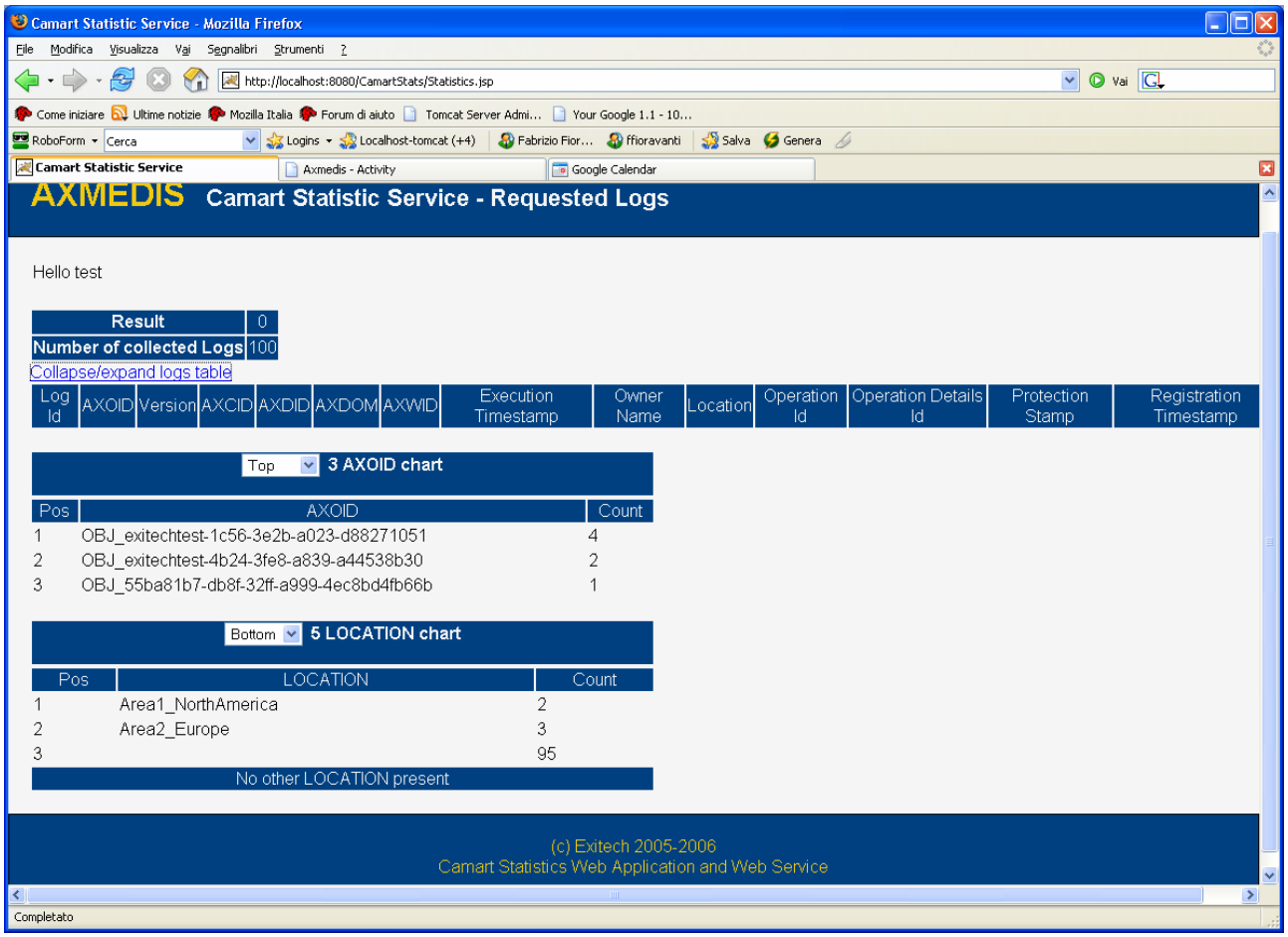
Where it can be evidenced that after providing a user name and password and a search criteria in the Date Field, the system will ask for the records contained in the AXCS and after getting such information the record are filtered according to some criteria based on “equal” operator applied to the fields AXOID, AXCID, AXDID, AXDOM, AXWID.
Moreover it is possible to add charts that are summary of the statistics in the form of Top and Bottom ten that can be provided for AXOID, AXCID, AXDID, location and operation.
By adding two charts we obtain:



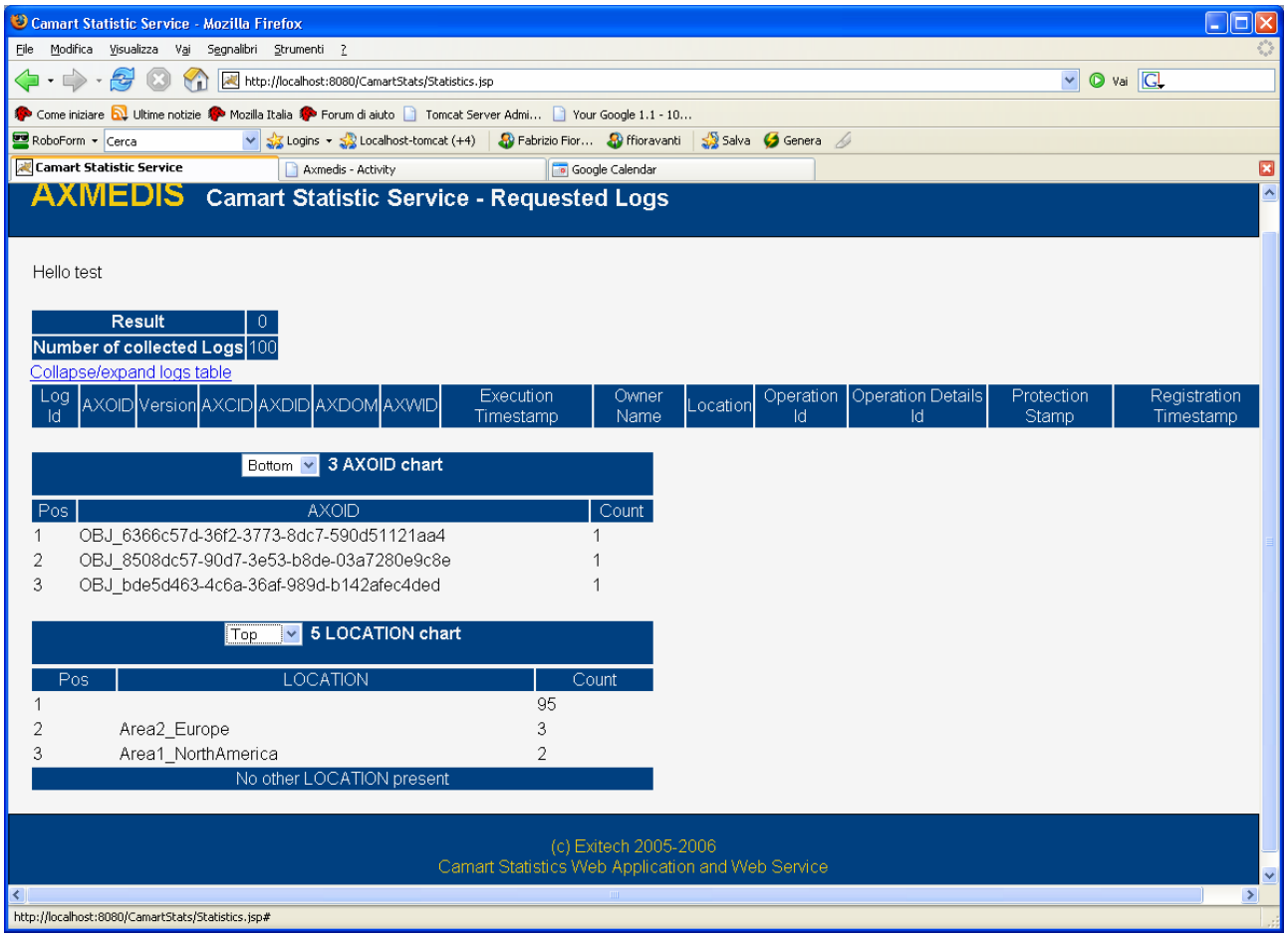
If a request without filtering is issued a web page like the following is obtained:



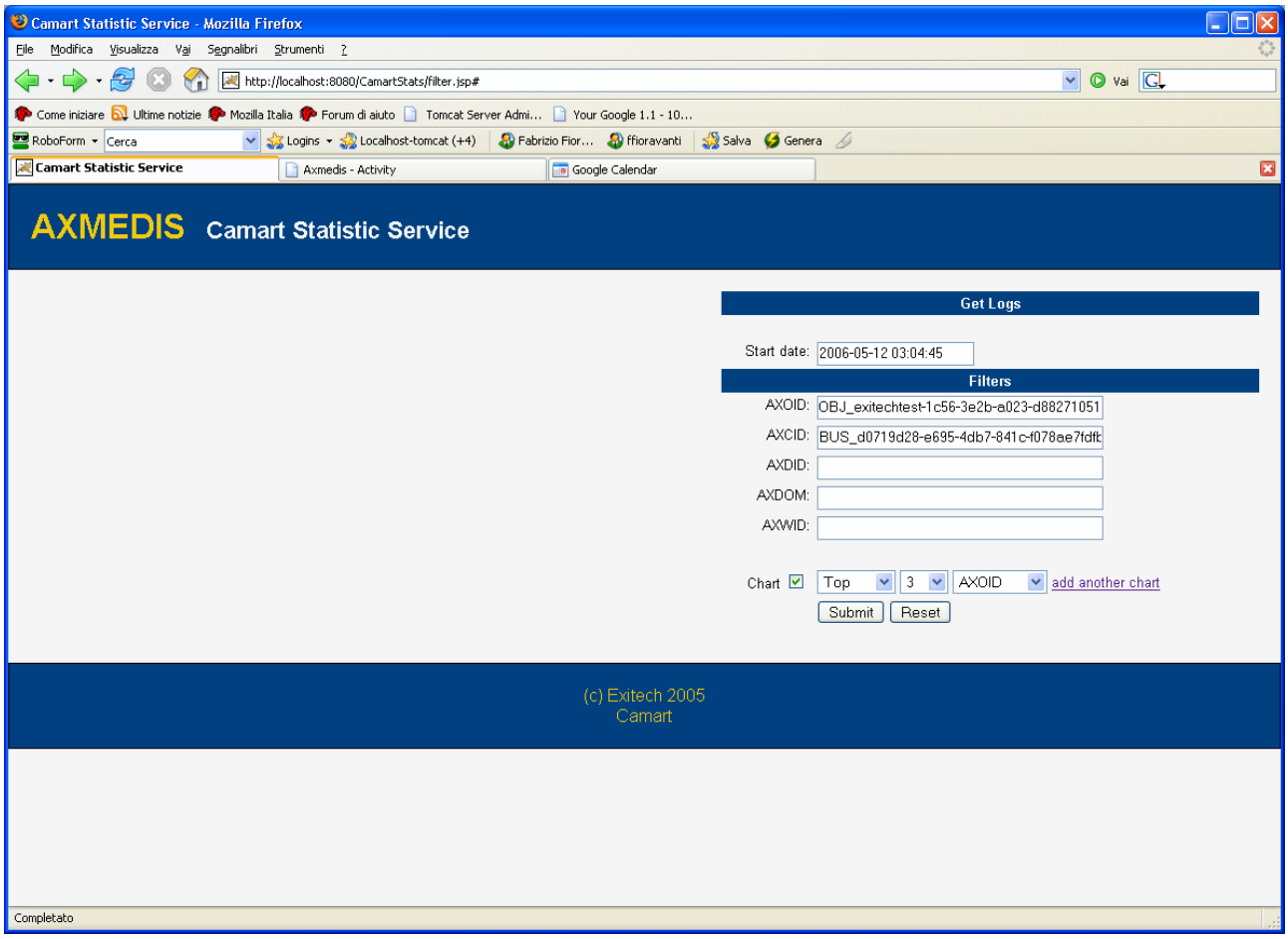
And by collapsing the results we obtain:



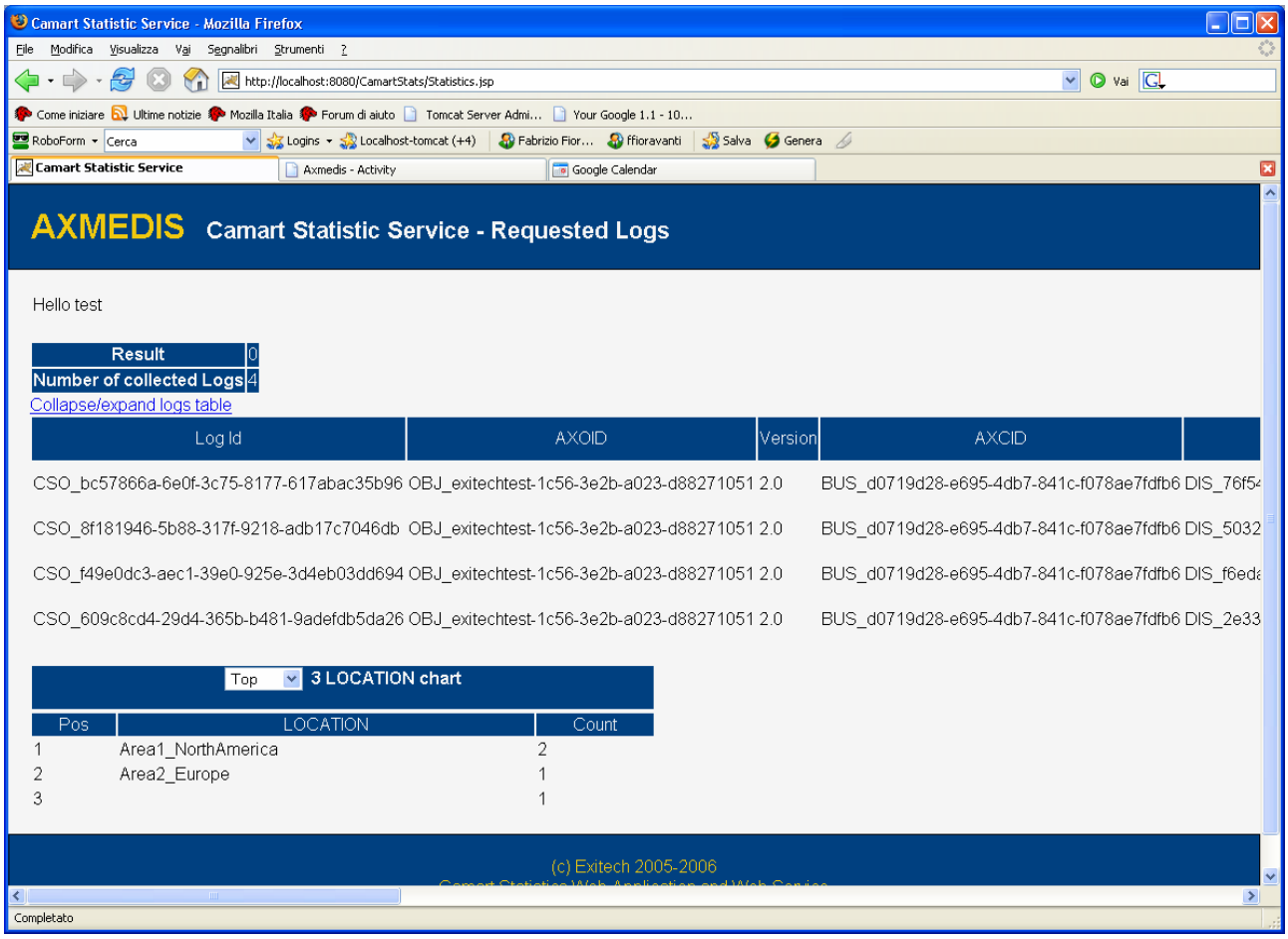
The charts can be provided in the form of the opposite of the request in terms of bottom and top:



The filtering is now provided AXCS server side in order to minimize the load on the network, and the results can be filtered as in the example below, where a filter on both AXOID and AXCID are provided.



The results obtained are:



5.4.3 Draft User Manual

Regarding the interface for statistics, the user manual is very simple since we have 3 sections that are:

- Dates: in this first part the range of date/time we want to analyze must be imposed. If the final date is left empty it is assumed the current date/time
- Filters: this section allows to filters the data for a variety of parameters such as AXOID, AXCID, AXDID, location etc, in order to have for example statistics only for a certain location (say Italy and for a certain creator)
- Charts: The user can ask to the system to have high level statistics such as top ten and bottom ten based on different parameters

5.4.4 Integration and compilation issues

Since Java and Apache Tomcat technologies, combined with W3C standard protocol (HTTP) and mark-up language (XHTML), were used, there is not any known issue related to interoperability and integration in different context.

Config parameter	Possible values
Camart.war\WEB-INF\classes\axdb.properties	
axdbUrl: URI of the axdb for the jdbc	jdbc:mysql://localhost/axdbv4?jdbcCompliantTruncation=false
axdbUser: user for the connection to the AXDB. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	axdbuser (usual user)
axdbPwd: password for the axdbUser for accessing database. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	mkzamk (usual pwd)

axdbMapping: file that contains the mapping for AXDB query engines **(not used in this project)**	C\:\\Inetpub\\wwwroot\\axmedis\\metadata-mapper.xml (usually it is there)
axdbDataSource: connection pool to be used with Tomcat (usually not to be changed unless you change also contex.xml). Used only if axdbManagerClass =it.exitech.axmedis.axdb.DataSourceAxdbManager	jdbc/listenerResource
axdbManagerClass: single ton class to be used for connecting to the AXDB. If it.exitech.axmedis.axdb.DataSourceAxdbManager is used please change also Camart.war/META-INF/context.xml	it.exitech.axmedis.axdb.DataSourceAxdbManager (recommended) it.exitech.axmedis.axdb.MysqlAxdbManager (for debug only)
initialConnections: number of connections in the pool. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	10 (suggested value)
increment: number of connection incremented when the pool is exhausted. Used only if axdbManagerClass =it.exitech.axmedis.axdb.MysqlAxdbManager	5 (suggested value)
Camart.war\WEB-INF\context.xml	
file that controls the DataSource to be created in Tomcat. Please change the parameters according to your database configuration	<?xml version="1.0" encoding="UTF-8"?> <Context docBase="Camart.war" path="/Camart"> <Resource auth="Container" driverClassName="org.gjt.mm.mysql.Driver" maxActive="8" maxIdle="4" name="jdbc/listenerResource" password="mkzamk" type="javax.sql.DataSource" url="jdbc:mysql://localhost/axdbv4?jdbcCompliantTruncation=false" username="axdbuser"/> </Context>
Camart.war\WEB-INF\classes\axcs.properties	
factoryuser: user entitled to download logs	mario
factorypassword: password of the user entitled to download logs	xxxxx
endpoint: endpoint for the statistic WS	http://localhost:8080/AXCSStatistics/services/Statistics

5.4.5 Errors reported and that may occur

Error code	Description and rationales
AccountException	Happens if user has not the rights to access the service, or if the HTTP session was terminated by inactivity time-out. AIIServlet should gently-fail upon this.

6 Analysis of CMS of partners (EXITECH, ALL)

This section is dedicated to the analysis of the partners' CMS in order to highlight the minimal set of data required by each partner on the basis of its profile (collecting society, distributor, content integrator, content producer, etc).
Each contributing partner has to provide or have provided:

- its role regarding the AII
- the minimal set of data needed by the Administrative part of the CMS
- the format of the interchange file generated by Administrative Information Integrator for the CMS, as described in the specification.

6.1 CMS data retrievable from AXCS (DSI)

Each Factory has its own database which stores pertinent data needed to make its own business.
AII will try to recover some factory's related data from AXCS, if they are present, otherwise the fields that are impossible to be recovered will be left blank. In any case, not all data requested by the partner are stored in AXCS databases. In the following, data retrievable from AXCS are reported:

Data Name	**Data Type**	**Data Description**	**Note**
Action	String	Use of the content (reading, printing, aggregating, editing.	
AreaCode	string	Geographical area in which the object is going to be used.	It is impossible to know where an object will be physically used: the only information we have in ActionLog table is the "Location" field which contains information about the country related to the pertinent collecting society
AXDID	String	Unique AXMEDIS Object Distributor ID that has performed the transaction	
AXOID	String	ID of the object inside the AXMEDIS system	
AXUID	String	ID of the User that has performed the transaction	
Device, Destination	string	Which use will be of the Axmedis Object (radio, television, kiosk, portable, others, etc)	the only information available is *EstimatedHWFingerprint* (in ActionLog table) Information about Device (pertinent Id) can be retrieved querying AXCS databases using *EstimatedHWFingerprint* value in the *actionlog* table in the AXCSAccounting database.
quantity, count, UtilizationsNumber	String	Quantity of the individual transaction object acquired or rented	It is always 1 because in ActionLog table there is one registration for each

			object usage (where usage means every kind of utilization, including aggregation and so on)
TransactionDate, TransactionTime, TransactionTimestamp	string	When the Object has been purchased	the time related information that can be provided by AXCS are ExecutionTimestamp and RegistrationTimestamp. The first refers to the instant when the action is executed, whether the latter refers to the instant the action is recorded in database. The information that will be provided in this field is the first (ExecutionTimestamp)
RegistrationTimestamp	String	instant the action is recorded in database	see above
transactionId	string	Unique ID of the AXMEDIS transaction	we can refer is the LogID in ActionLog table
LicenceId	String	ID of the licence	AXLID in AXCS

6.2 AFI CMS related data (AFI)

6.2.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Collecting Society
- Content Owner
- Content Creator

6.2.2 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes

Data Name	**Data Type**	**Data Description**
Transaction ID	string	Unique ID of the AXMEDIS transaction
AXOID	String	ID of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	ID of the User that has performed the transaction
		Data of the user (name, address, phone, mail)
Transaction Value	String	The overall value of the transaction
paymentForm	String	The code describing payment form selected. This code can have one of the following values: IC – Cash CC – Credit card CP – Coupon PP – Pre paid card
Destination	String	Which use will be of the AXMEDIS Object (radio, television, kiosk, portable, others, etc)
TransactionDate	String	When the Object is been purchased
UtilizationsNumber	String	How many times the objects is going to be used concerning this transaction
AreaCode	string	Geographical area in which the object is going to be

		used.

6.2.3 File format for data input

The file format can be one of the following:

- **CSV (comma separated values)**

Structure of .TXT file for loading data into MySql DB MySql Table for importing data.
File .txt as described below. Data separated by [;].

ISRC	>>>>>>	INT
AXOID		INT
AXUID		INT
Transaction ID		INT
PaymentForm		IC/CC/CP/PP > CHAR(2)
TrasactionValue		INT
Destination		A list to be defined > VARCHAR-
UtilizationNumber		INT
AreaCode		VARCHAR
TransactionDate		INT-Unix Timestamp

6.3 EUTELSAT CMS related data (EUTELSAT)

6.3.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Distributor

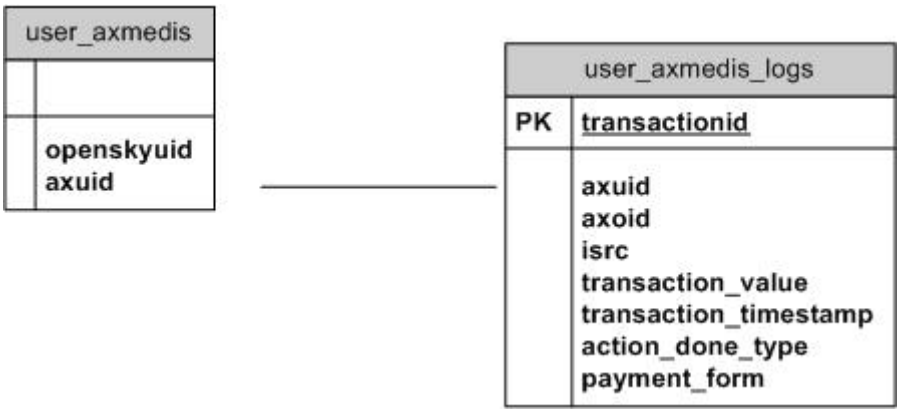
6.3.2 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes

Data Name	**Data Type**	**Data Description**
AXOID	String	Unique descriptor of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	User that has performed the transaction
action_done	String	The code describing expected usage for the object. This code can have one of the following values: P – Purchase/sell R – Rent S – Subscription (only for services)
paymentForm	String	The code describing payment form selected. This code can have one of the following values: IC – Cash CC – Credit card CP – Coupon PP – Pre paid card Dxxx – Differed where accepted values for xxx are: D030 – Differed 30 dd D060 – Differed 60 dd D090 – Differed 90 dd

		D120 – Differed 120 dd
transactionValue	String	The overall value of the transaction
transactionTimestamp	Timestamp	Timestamp of transaction occurrence

The received information will be stored in the following tables:



6.3.3 File format for data input

The file format can be one of the following:

- CSV (comma separated values) **Y**

Structure of .TXT file for loading data into Postgresql DB.
File .txt as described below. Data separated by [;].

ISRC		INT
AXOID		INT
AXUID		INT
Transaction ID		INT
PaymentForm		IC/CC/CP/PP > CHAR(2)
TrasactionValue		INT
TransactionTimestamp		INT-Unix Timestamp
ActionDoneType		VARCHAR

6.4 ILABS CMS related data (ILABS)

6.4.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Owner
- Content Distributor
- Content Creator
- Content Integrator

6.4.2 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes

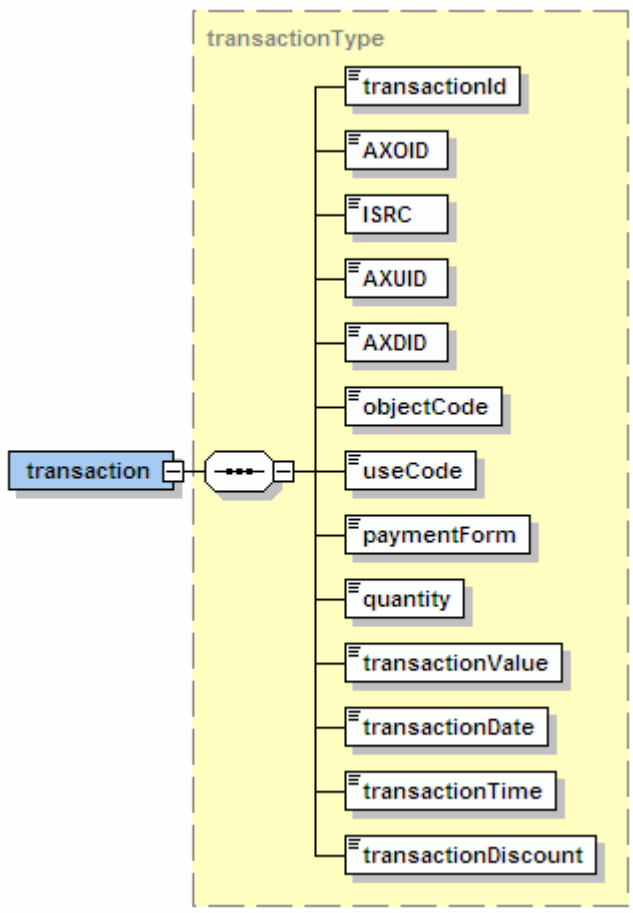
Data Name	**Data Type**	**Data Description**
transactionId	String	Unique transaction ID
AXOID	String	Unique descriptor of the object inside the AXMEDIS system

ISRC	String	Unique standard ID for identifying the piece
AXUID	String	User that has performed the transaction
AXDID	String	Unique AXMEDIS Object Distributor ID that has performed the transaction
objectCode	String	The code describing object's kind. This code can have one of the following values: RO – Raw object AO – Aggregated object CO – Course SO – Service OO – Other object
useCode	String	The code describing expected usage for the object. This code can have one of the following values: P – Purchase/sell R – Rent S – Subscription (only for services)
paymentForm	String	The code describing payment form selected. This code can have one of the following values: IC – Cash CC – Credit card CP – Coupon PP – Pre paid card Dxxx – Differed where accepted values for xxx are: D030 – Differed 30 dd D060 – Differed 60 dd D090 – Differed 90 dd D120 – Differed 120 dd
quantity	String	Quantity of the individual transaction object acquired or rented
transactionValue	String	The overall value of the transaction
transactionDate	String	Date of transaction occurrence
transactionTime	String	Time of transaction occurrence
transactionDiscount	String	Value of applied discount, 0 if none

6.4.3 File format for data input

The file format can be one of the following:

- **XML file**



ILABS Exchange file Data Format	
XML Schema	<?xml version="1.0" encoding="UTF-8"?> <!-- edited with XMLSPY v5 rel. 4 U (http://www.xmlspy.com) by Andrea Lorenzon (Giunti Interactive Labs S.r.l.) --> <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified"> <xs:element name="transaction" type="transactionType"/> <xs:complexType name="transactionType"> <xs:sequence> <xs:element name="transactionId" type="xs:string"/> <xs:element name="AXOID" type="xs:string"/> <xs:element name="ISRC" type="xs:string"/> <xs:element name="AXUID" type="xs:string"/> <xs:element name="AXDID" type="xs:string"/> <xs:element name="objectCode"> <xs:simpleType> <xs:restriction base="xs:token"> <xs:enumeration value="RO"/> <xs:enumeration value="AO"/> <xs:enumeration value="CO"/> <xs:enumeration value="SO"/> <xs:enumeration value="OO"/> </xs:restriction> </xs:simpleType> </xs:element> <xs:element name="useCode"> <xs:simpleType> <xs:restriction base="xs:token">

```
                        <xs:enumeration value="P"/>
                        <xs:enumeration value="R"/>
                        <xs:enumeration value="S"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="paymentForm">
                <xs:simpleType>
                    <xs:restriction base="xs:token">
                        <xs:enumeration value="IC"/>
                        <xs:enumeration value="CC"/>
                        <xs:enumeration value="CP"/>
                        <xs:enumeration value="PP"/>
                        <xs:enumeration value="D030"/>
                        <xs:enumeration value="D060"/>
                        <xs:enumeration value="D090"/>
                        <xs:enumeration value="D120"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="quantity" type="xs:string"/>
            <xs:element name="transactionValue" type="xs:string"/>
            <xs:element name="transactionDate" type="xs:string"/>
            <xs:element name="transactionTime" type="xs:string"/>
            <xs:element name="transactionDiscount" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

[EXITECH] The main drawback of this solution consists in the fact that it has to be generated one file for each transaction, but as discussed with Fuschi this seems to be the solution more close to ILABS needs. In this case it is necessary to define a unique name for the file, that can be an UUID or something based on object ID plus timestamp.

6.5 TISCALI CMS related data (TISCALI)

6.5.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Distributor

6.5.2 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes

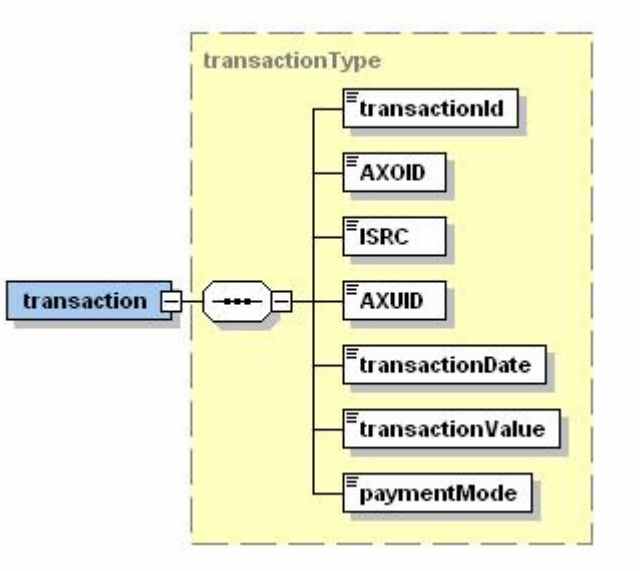
Data Name	**Data Type**	**Data Description**
transactionId	String	Unique ID of the AXMEDIS transaction
AXOID	String	Unique ID of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	Unique ID of the user that has performed the transaction
transactionDate	DateTime	Date and Time of the transaction
transactionValue	String	The overall value of the transaction
paymentMode	String	The code describing payment mode selected. This code can have one of the following values: CC - Credit card CP - Coupon

		PP - Pre paid card

6.5.3 File format for data input

The file format can be one of the following:

- XML file Y



XML Schema of .XML file for loading data into Postgresql DB

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:complexType name="transactionType">
        <xs:sequence>
            <xs:element name="transactionId" type="xs:string"/>
            <xs:element name="AXOID" type="xs:string"/>
            <xs:element name="ISRC" type="xs:string"/>
            <xs:element name="AXUID" type="xs:string"/>
            <xs:element name="transactionDate" type="xs:dateTime"/>
            <xs:element name="transactionValue" type="xs:string"/>
            <xs:element name="paymentMode">
                <xs:simpleType>
                    <xs:restriction base="xs:token">
                        <xs:enumeration value="CC"/>
                        <xs:enumeration value="CP"/>
                        <xs:enumeration value="PP"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="transaction" type="transactionType"/>
</xs:schema>
```

6.6 XIM CMS related data (XIM)

6.6.1 Role

6.6.2 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Integrator (primary function)
- Content Creator
- Content Owner (plan to be selling some own content)

6.6.3 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes

Data Name	Data Type	Data Description
AXOID	String	Unique descriptor of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	User that has performed the transaction
quantity	String	Quantity of the individual transaction object acquired or rented
transactionValue	String	The overall value of the transaction
transactionTimestamp	String	Timestamp of transaction occurrence

6.6.4 File format for data input

The file format can be one of the following:

- XML file: not necessary, unless this simplifies production of the data by being consistent with other partners
- CSV (comma separated values)
    - AXOID, ISRC, AXUID, quantity, transactionValue, transactionDate
    - tab delimiter

6.7 SEJER CMS related data (SEJER)

6.7.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Owner
- Content Distributor
- Content Integrator

6.7.2 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes

Data Name	Data Type	Data Description
AXOID	String	Unique descriptor of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	User that has performed the transaction
Device	String	Type of device (computer, PDA, mobile)
Action	String	Use of the content (reading, printing, aggregating, editing,…)
TransactionTime	DateTime	Timestamp of transaction occurrence
LicenseId	String	The identifier of the license used to grant access to the object

6.7.3 File format for data input

The file format can be one of the following:

- CSV (comma separated values)

- AXOID, ISRC, AXUID, Device, Action, Count
- comma delimiter

6.8 ANSC CMS related data (ANSC)

6.8.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Owner
- Content Distributor
- Content Creator (soon)
- Content Integrator (potentially)
-

6.8.2 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes.

Data Name	**Data Type**	**Data Description**
AXOID	String	Unique descriptor of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	User that has performed the transaction
Range	date	Range of dates to which the report refers
Quantity	String	Quantity of the individual transaction object acquired or rented
transactionValue	String	The overall value of the transaction (all ANSC objects)
transactionTimestamp	String	Timestamp of transaction occurrence
transactionperAXOID	String	Number of transaction per single AXIOID
objectCode	String	The code describing object's kind. This code can have one of the following values: RO – Raw object AO – Aggregated object CO – Course SO – Service OO – Other object
useCode	String	The code describing expected usage for the object. This code can have one of the following values: P – Purchase/sell R – Rent S – Subscription (only for services)
paymentForm	String	The code describing payment form selected. This code can have one of the following values: IC – Cash CC – Credit card CP – Coupon PP – Pre paid card Dxxx – Differed where accepted values for xxx are: D030 – Differed 30 dd D060 – Differed 60 dd D090 – Differed 90 dd D120 – Differed 120 dd

6.8.3 File format for data input

Since the CMS of the ANSC which manages the framework of digital content is separated from the administration software we are not able now to ask for a specific format or fields of data beyond them already chosen by other content partners.
In this case we think that a simple text file is the more flexible format or, alternatively, an XML file. CSV could be also n alternative since in the future we ca find the way to convert it and import in the administrative procedure which at present is NOT a CMS. Our CMS does not comprehend administrative data, just contents.

The file format can be one of the following:

- CSV (comma separated values) : Field sequence as in the table before, and semicolon as separator

6.9 HP CMS related data (HP)

NO INFORMATION PROVIDED

6.10 BBC CMS related data (BBC)

NO INFORMATION PROVIDED

6.11 TEO CMS related data (TEO)

6.11.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Distributor

6.11.2 Minimal set of data needed

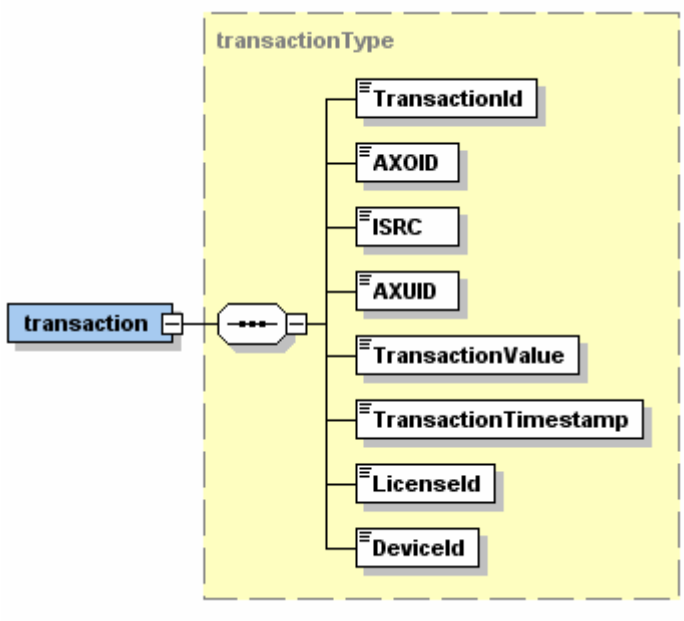
In the following table report the minimal set of data necessary for your administrative purposes.

Data Name	Data Type	Data Description
TransactionId	String	Unique transaction ID
AXOID	String	Unique descriptor of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	User that has performed the transaction
TransactionValue	String	The overall value of the transaction
TransactionTimestamp	DateTime	Timestamp of transaction occurrence
LicenseId	String	The identifier of the license used to grant access to the object
DeviceId	String	STB which consumed content fingerprint

6.11.3 File format for data input

The file format can be one of the following:

- XML file



TEO Exchange file Data Format XSD Schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
	<xs:complexType name="transactionType">
		<xs:sequence>
			<xs:element name="TransactionId" type="xs:string"/>
			<xs:element name="AXOID" type="xs:string"/>
			<xs:element name="ISRC" type="xs:string"/>
			<xs:element name="AXUID" type="xs:string"/>
			<xs:element name="TransactionValue" type="xs:string"/>
			<xs:element name="TransactionTimestamp" type="xs:dateTime"/>
			<xs:element name="LicenseId" type="xs:string"/>
			<xs:element name="DeviceId" type="xs:string"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="transaction" type="transactionType"/>
</xs:schema>
```

6.12 TI CMS related data (TI)

NO INFORMATION PROVIDED

6.13 SDAE CMS related data (SDAE)

6.13.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Integrator

6.13.2 Minimal set of data needed

In the following table report the minimal set of data necessary for your administrative purposes.

Data Name	Data Type	Data Description
NOTIFIER.ID	Integer	LCD commercial client Identifier
CONTENT.ID	Integer	LCD content identifier
DATE	Date	Date of use
MODALITY.ID	Integer	Use modality
PRICE_LCD	Float	LCD Price
TIME	Time	Time of use
COUNTRY.ID	string	ISO 2 code for country of use

6.13.3 File format for data input

The file format can be one of the following:

- XML file

LCD (xml) Report format

```
<?xml version="1.0" encoding="iso-8859-1" ?>
  <!DOCTYPE LCD.USES (View Source for full doctype...)>
- <LCD.USES>
  <NOTIFIER.ID>6</NOTIFIER.ID>
- <USE>
  <CONTENT.ID>456</CONTENT.ID>
  <DATE DAY="01" MONTH="01" YEAR="2004" />
  <MODALITY.ID>1</MODALITY.ID>
- <PRICE>
  <PRICE_CLI>2.5</PRICE_CLI>
    </PRICE>
  <COUNTRY.ID>ES</COUNTRY.ID>
    </USE>
- <USE>
  <CONTENT.ID>456</CONTENT.ID>
  <DATE DAY="01" MONTH="01" YEAR="2004" />
  <TIME HOURS="10" MINUTES="01" SECONDS="20" />
  <MODALITY.ID>1</MODALITY.ID>
- <PRICE>
  <PRICE_LCD>2.5</PRICE_LCD>
    </PRICE>
- <PRICE>
  <PRICE_CLI>2.5</PRICE_CLI>
    </PRICE>
  <COUNTRY.ID>ES</COUNTRY.ID>
    </USE>
- <USE>
  <CONTENT.ID>456</CONTENT.ID>
  <DATE DAY="01" MONTH="01" YEAR="2004" />
  <MODALITY.ID>1</MODALITY.ID>
- <PRICE>
  <PRICE_LCD>2.5</PRICE_LCD>
    </PRICE>
  <COUNTRY.ID>ES</COUNTRY.ID>
    </USE>
    </LCD.USES>
```

6.14 ELION CMS related data (ELION)

6.14.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Distributor

6.14.2 Minimal set of data needed

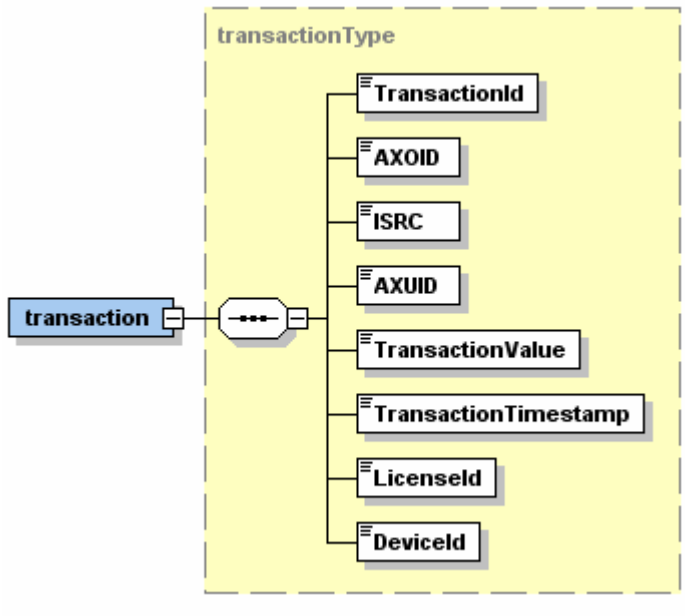
In the following table report the minimal set of data necessary for your administrative purposes.

Data Name	Data Type	Data Description

TransactionId	**String**	**Unique transaction ID**
AXOID	**String**	**Unique descriptor of the object inside the AXMEDIS system**
ISRC	**String**	**Unique standard ID for identifying the piece**
AXUID	**String**	**User that has performed the transaction**
TransactionValue	**String**	**The overall value of the transaction**
TransactionTimestamp	**DateTime**	**Timestamp of transaction occurrence**
LicenseId	**String**	**The identifier of the license used to grant access to the object**
DeviceId	**String**	**Device identity.**

6.14.3 File format for data input

- XML file



ELION Exchange file Data Format XSD Schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
	<xs:complexType name="transactionType">
		<xs:sequence>
			<xs:element name="TransactionId" type="xs:string"/>
			<xs:element name="AXOID" type="xs:string"/>
			<xs:element name="ISRC" type="xs:string"/>
			<xs:element name="AXUID" type="xs:string"/>
			<xs:element name="TransactionValue" type="xs:string"/>
			<xs:element name="TransactionTimestamp" type="xs:dateTime"/>
			<xs:element name="LicenseId" type="xs:string"/>
			<xs:element name="DeviceId" type="xs:string"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="transaction" type="transactionType"/>
```

```
</xs:schema>
```

6.15 VRS CMS related data (VRS)

6.15.1 Role

In the following the role of the partner regarding the administrative data to be collected is summarized.

- Content Creator Y

6.15.2 Minimal set of data needed

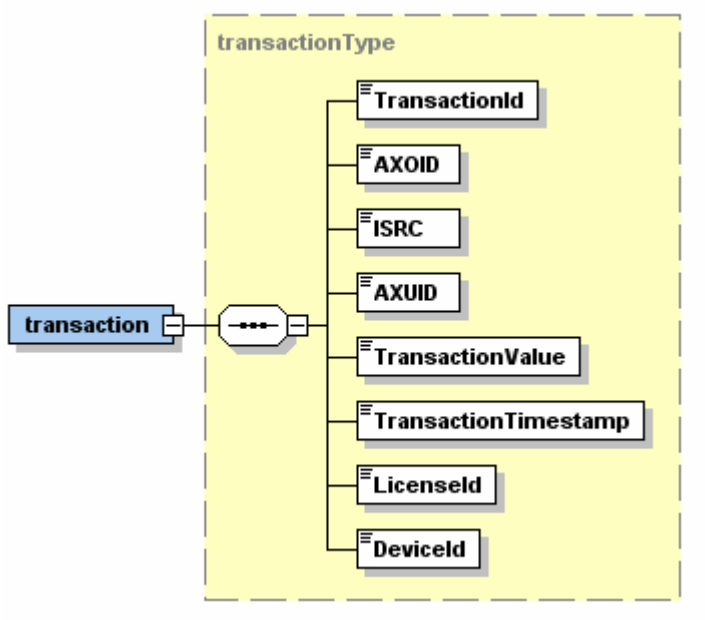
In the following table report the minimal set of data necessary for your administrative purposes.

Data Name	Data Type	Data Description
TransactionId	String	Unique transaction ID
AXOID	String	Unique descriptor of the object inside the AXMEDIS system
ISRC	String	Unique standard ID for identifying the piece
AXUID	String	User that has performed the transaction
TransactionValue	String	The overall value of the transaction
TransactionTimestamp	DateTime	Timestamp of transaction occurrence
LicenseId	String	The identifier of the license used to grant access to the object
DeviceId	String	STB which consumed content fingerprint

6.15.3 File format for data input

The file format can be one of the following:

- XML file



VRS Exchange file Data Format XSD Schema:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
	<xs:complexType name="transactionType">
		<xs:sequence>
			<xs:element name="TransactionId" type="xs:string"/>
			<xs:element name="AXOID" type="xs:string"/>
			<xs:element name="ISRC" type="xs:string"/>
			<xs:element name="AXUID" type="xs:string"/>
			<xs:element name="TransactionValue" type="xs:string"/>
			<xs:element name="TransactionTimestamp" type="xs:dateTime"/>
			<xs:element name="LicenseId" type="xs:string"/>
			<xs:element name="DeviceId" type="xs:string"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="transaction" type="transactionType"/>
</xs:schema>
```

7 Administrative Information Integrator format detailed specification

The union of the information requested by partners can be summarized as in the following table with the addition of the last column where the domain to which the information is related are reported.
Domain are the following:

- AXCS: information that is stored and provided by AXCS
- Business: Information related to business that is stored outside AXMEDIS system
- Factory: Information that can be retrieved in the factory, if present
- AII: Information generated by the Administrative Information Integrator

Data Name	**Data Type**	**Data Description**	**Information domain**
reportDate	String	Range of dates to which the report refers	AII
AXDID	String	Unique AXMEDIS Object Distributor ID that has performed the transaction	AXCS
logId	String	Unique ID of the AXMEDIS transaction	AXCS
AXLID	String	ID of the licence bounded to the transaction	AXCS
AXOID	String	ID of the object inside the AXMEDIS system	AXCS
AXUID	String	ID of the User that has performed the transaction	AXCS
device	String	Which use has been done for the AXMEDIS Object (radio, television, kiosk, portable, others, etc)	AXCS
executionTimestamp	String	Timestamp of the execution of the Transaction	AXCS
registrationTimestamp	String	Timestamp of the registration of the Transaction in AXCS	AXCS
location	String	Geographical area in which the object is used.	AXCS
operation	String	Use of the content (reading, printing, aggregating, editing).	AXCS
quantity	String	Quantity of the individual transaction object acquired or rented	AII, always 1
transactionValue	String	The overall value of the transaction	Business
userDetails	String	detail of the user	Business
paymentForm	String	The code describing payment form selected. This code can have one of the following values: IC – Cash CC – Credit card CP – Coupon PP – Pre paid card	Business
ISRC	String	Unique standard ID for identifying the piece	Factory, this field will be filled with the content of the field “identifier” of the “DCMI” table if present
objectCode	String	The code describing object’s kind: Raw object, Aggregated object, Course,	Factory, this field fill be filled with the content of the

		Service, Other object	field “type” of the “DCMI” table if present

AXCS and AII information will be provided always.
Factory information will be provided only if the information can be retrieved in the AXDB that means if the object with the given AXOID is inside the factory and is indexed in the AXDB, otherwise will be left blank.
Business Information cannot be provided at all

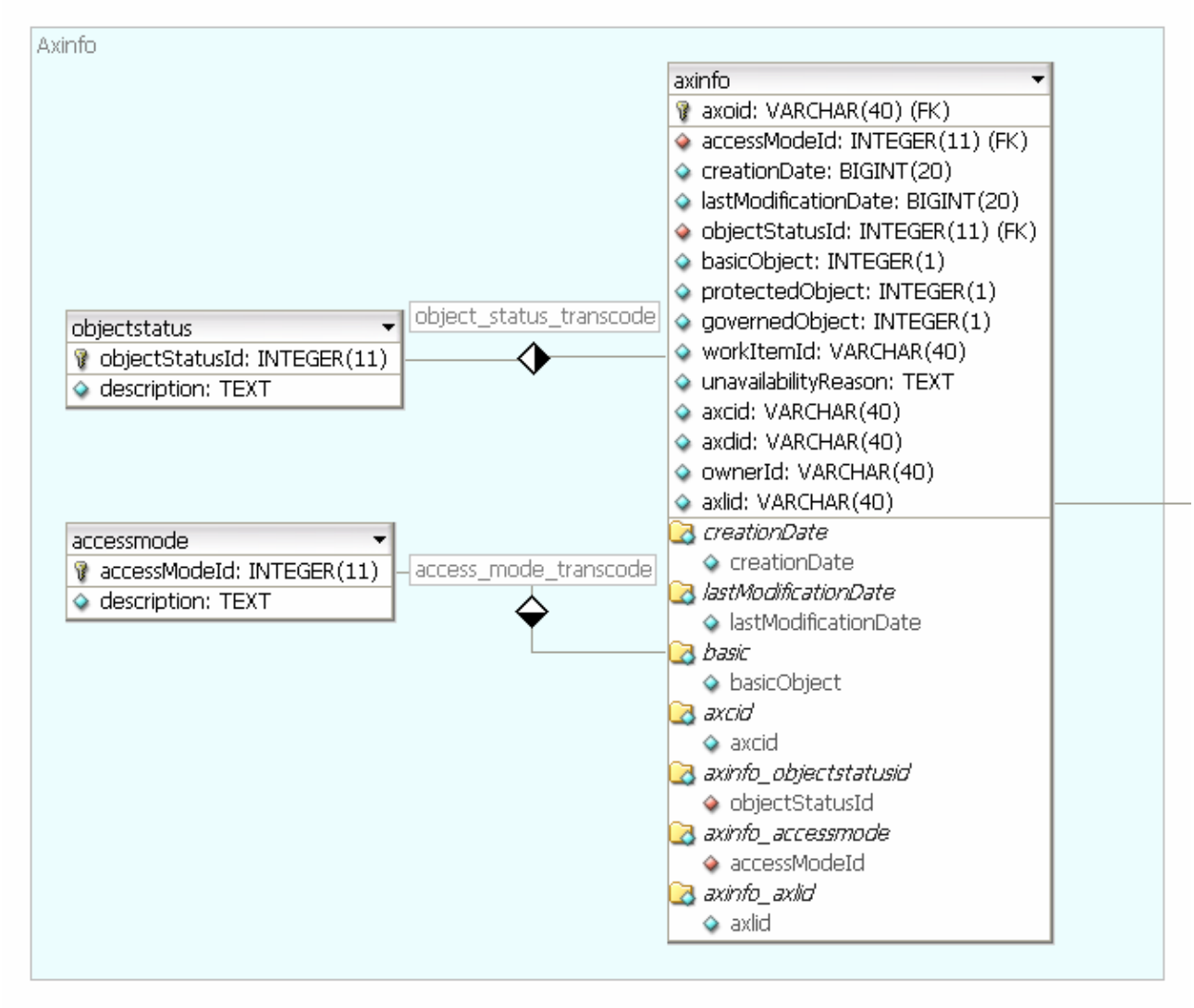
The mapping among the data requested by the partners and the previous table can be summarized in the following table where the information domain is reported and therefore it can be evidenced if the information will be provided or not.

Data Name	**Partner**	**Domain**	**Mapped AII Data**
action_done	EUTELSAT, SEJER	AXCS	operation
AreaCode	AFI	AXCS	location
AXDID	ILABS	AXCS	AXDID
AXLID	SEJER	AXCS	AXLID
AXOID	AFI, EUTELSAT, ILABS, TISCALI, XIM, SEJER, ANSC	AXCS	AXOID
AXUID	AFI, EUTELSAT, ILABS, TISCALI, XIM, SEJER, ANSC	AXCS	AXUID
Destination	AFI	AXCS	device
Device	SEJER	AXCS	device
ISRC	AFI, EUTELSAT, ILABS, TISCALI, XIM, SEJER, ANSC	Factory	ISRC
objectCode	ILABS, ANSC	Factory	objectCode
paymentForm	AFI, EUTELSAT, ILABS, TISCALI, ANSC	Business	---
quantity	ILABS, XIM, ANSC	AII	1
Range	ANSC	AII	reportDate
Transaction ID	AFI	AXCS	logId
transactionDate	AFI, ILABS, TISCALI	AXCS	executionTimestamp
transactionDiscount	ILABS	Business	---
transactionId	ILABS, TISCALI	AXCS	logId
transactionperAXOID	ANSC	AII	1
TransactionTime	ILABS	AXCS	executionTimestamp
transactionTimestamp	EUTELSAT, XIM, ANSC	AXCS	executionTimestamp
transactionValue	AFI, EUTELSAT, ILABS, TISCALI, XIM, ANSC	Business	---
useCode	ILABS, ANSC	AXCS	operation
UserDetail	AFI	Business	---
UtilizationsNumber	AFI	AII	1

As a summary, we can say that:

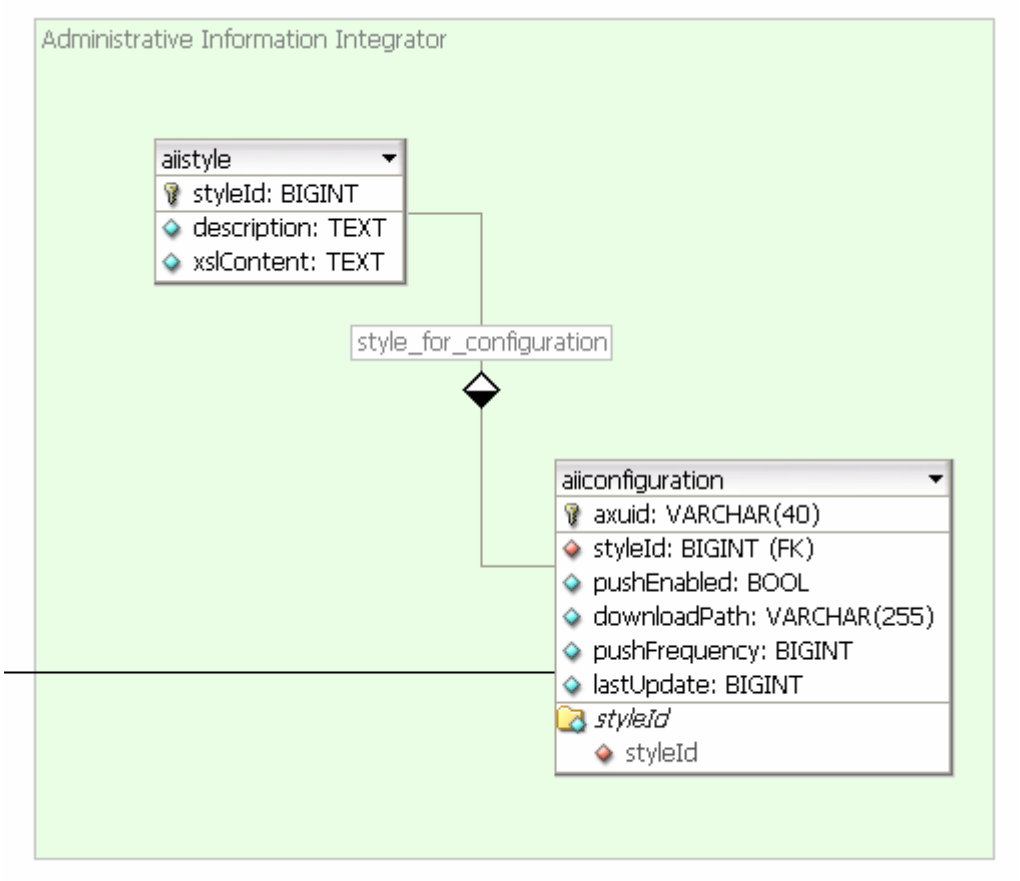
- Business information cannot be provided
- Factory information will be provided if present
- AXCS information will be stored in the AII DB
- AII information will be generated on the fly

A new version of the table related to the AXCS log needs to be generated starting from the data collected in this deliverable, and therefore the new implementation of such tables will be as in the next figure. Only indexes are subject to change.



As already stated in the introduction, the AII can be configured for each user in the company and therefore more than 1 profile for each company can be created. This allows to different people to manage differently formatted information, at least in the so called “polling” mode where the user can ask on demand the information needed.

This part needs the support of a database structure that can be modelled as showed below:

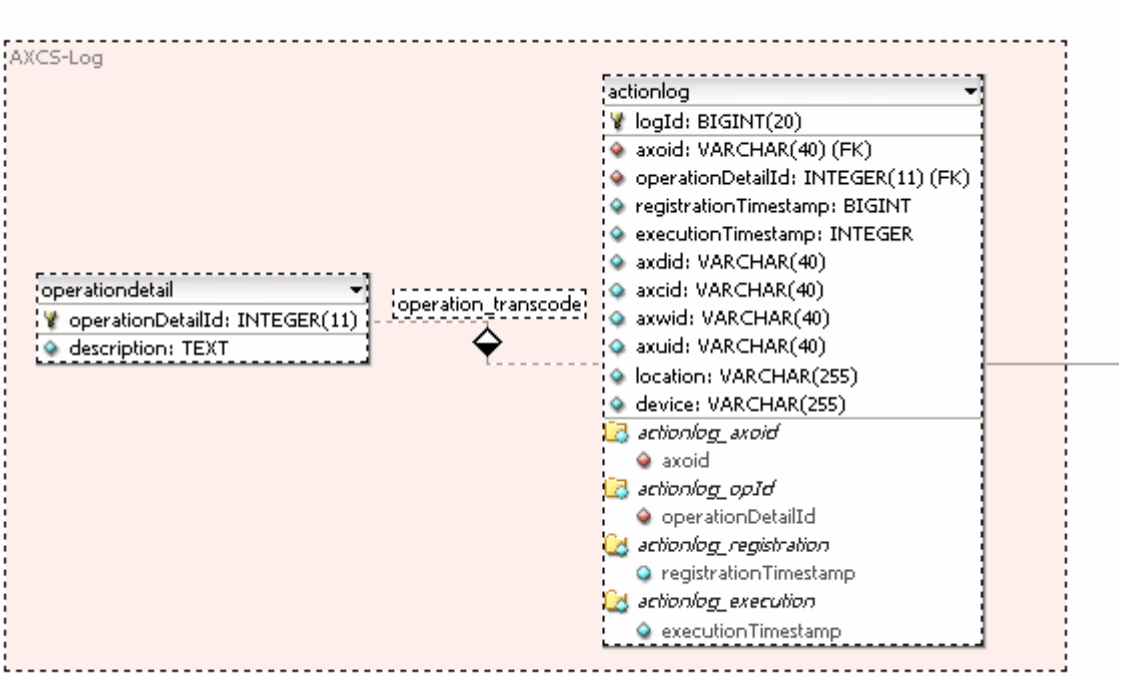


The DB structure suppose that the same style can be used by different people in the company and therefore put it in an external table, while in the main table together with the AXIUD are reported the main information needed to provide the file to the CMS/CRM system. These information are mainly:

- pushEnable shows if the push modality is enabled or less.
- downloadPath: the path where the user will download the file generated by the AII. This must must contains also the credential to access such as ftp://user:passwd@server.foo.com/dir
- pushFrequency is the frequency of pushing operation in minutes
- lastUpdate contains the registrationTimestamp of the last log that that has been downloaded by the user

8 Table description for database AXDB (CAMART and AII related tables) and XML schemas

In this section the AXDB table involved with CAMART and AII are described.
The table for storing the information provided by AXCS Reporting Web service is reported below with the description of the fields.



AXCS-Log

actionlog

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
logId	**BIGINT(20)**	PK	NN				AI
axoid	VARCHAR(40)		NN				
operationDetailId	INTEGER(11)		NN	UNSIGNED			
registrationTimestamp	BIGINT		NN				
executionTimestamp	INTEGER		NN	UNSIGNED			
axdid	VARCHAR(40)						
axcid	VARCHAR(40)						
axwid	VARCHAR(40)						
axuid	VARCHAR(40)		NN			general user of the system that has performed the operation: not an internal user	
location	VARCHAR(255)						
device	VARCHAR(255)						

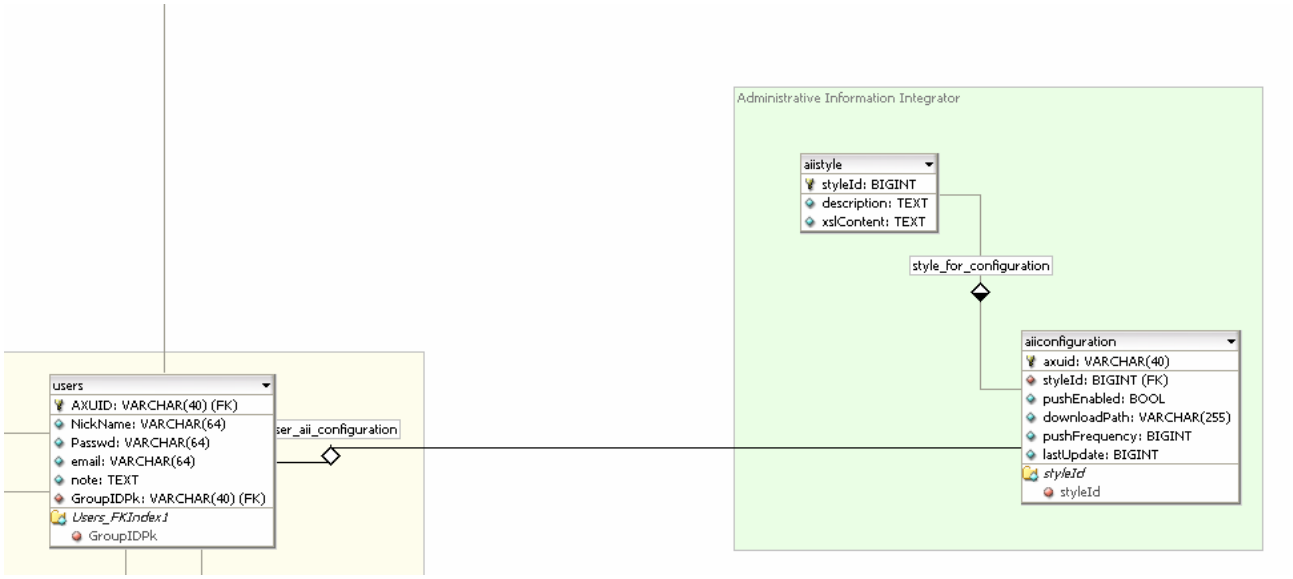
IndexName	IndexType	Columns
PRIMARY	PRIMARY	logId
actionlog_axoid	Index	axoid
actionlog_opId	Index	operationDetailId
actionlog_registration	Index	registrationTimestamp
actionlog_execution	Index	executionTimestamp

operationdetail

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

IndexName	IndexType	Columns
PRIMARY	PRIMARY	operationDetailId

The table for storing the information needed by AII below with the description of the fields.



Administrative Information Integrator

aiistyle

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

IndexName	IndexType	Columns
PRIMARY	PRIMARY	styleId

aiiconfiguration

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
axuid	**VARCHAR(40)**	PK	NN				
styleId	BIGINT		NN				
pushEnabled	BOOL		NN				
downloadPath	VARCHAR(255)		NN				
pushFrequency	BIGINT		NN				

lastUpdate	BIGINT		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	axuid
styleId	Index	styleId

UserGroup

users

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
AXUID	**VARCHAR(40)**	PK	NN				
NickName	VARCHAR(64)		NN				
Passwd	VARCHAR(64)		NN				
email	VARCHAR(64)		NN				
note	TEXT						
GroupIDPk	VARCHAR(40)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	AXUID
Users_FKIndex1	Index	GroupIDPk

8.1 Formal description of format AII Record

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
	<!-- version 1.1 (2-Mar-2006)
-->
	<xs:element name="AIIRecord">
		<xs:annotation>
			<xs:documentation>Root element for an Administrative Information Intergrator
record</xs:documentation>
		</xs:annotation>
		<xs:complexType>
			<xs:sequence>
				<xs:element name="logId" type="xs:string">
					<xs:annotation>
						<xs:documentation>Unique ID of the Axmedis
transaction</xs:documentation>
					</xs:annotation>
				</xs:element>
				<xs:element name="executionTimestamp" type="xs:dateTime">
					<xs:annotation>
						<xs:documentation>timestamp of the execution of the
Transaction</xs:documentation>
					</xs:annotation>
				</xs:element>
				<xs:element name="registrationTimestamp" type="xs:dateTime">
					<xs:annotation>
						<xs:documentation>Timestamp of the registration of the Transaction in
AXCS</xs:documentation>
					</xs:annotation>
				</xs:element>
				<xs:element name="AXOID">
					<xs:annotation>
						<xs:documentation>ID of the object inside the axmedis
system</xs:documentation>
					</xs:annotation>
					<xs:simpleType>
						<xs:restriction base="xs:string">
							<xs:maxLength value="40"/>
							<xs:whiteSpace value="collapse"/>
						</xs:restriction>
					</xs:simpleType>
				</xs:element>
				<xs:element name="AXUID">
					<xs:annotation>
						<xs:documentation>ID of the User that has performed the
transaction</xs:documentation>
					</xs:annotation>
					<xs:simpleType>
						<xs:restriction base="xs:string">
							<xs:maxLength value="40"/>
							<xs:whiteSpace value="collapse"/>
						</xs:restriction>
					</xs:simpleType>
				</xs:element>
				<xs:element name="AXDID">
					<xs:annotation>
						<xs:documentation>Unique AXMEDIS Object Distributor ID that has
performed the transaction</xs:documentation>
					</xs:annotation>
					<xs:simpleType>
						<xs:restriction base="xs:string">
```

```
                                                <xs:maxLength value="40"/>
                                                <xs:whiteSpace value="collapse"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>
                        <xs:element name="AXLID">
                                <xs:annotation>
                                        <xs:documentation>ID of the licence bounded to the
transaction</xs:documentation>
                                </xs:annotation>
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:maxLength value="40"/>
                                                <xs:whiteSpace value="collapse"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>
                        <xs:element name="device" type="xs:string">
                                <xs:annotation>
                                        <xs:documentation>Which use has been done for the Axmedis Object
(radio,television,kiosk,portable,others,etc)</xs:documentation>
                                </xs:annotation>
                        </xs:element>
                        <xs:element name="operation" type="xs:string">
                                <xs:annotation>
                                        <xs:documentation>Use of the content (reading, printing, aggregating,
editing).</xs:documentation>
                                </xs:annotation>
                        </xs:element>
                        <xs:element name="location" type="xs:string">
                                <xs:annotation>
                                        <xs:documentation>Geographical area in which the object is
used.</xs:documentation>
                                </xs:annotation>
                        </xs:element>
                        <xs:element name="ISRC" type="xs:string" minOccurs="0">
                                <xs:annotation>
                                        <xs:documentation>Factory Infromation: Unique standard ID for
identifying the piece</xs:documentation>
                                </xs:annotation>
                        </xs:element>
                        <xs:element name="objectCode" type="xs:string" minOccurs="0">
                                <xs:annotation>
                                        <xs:documentation>Factory Information: this field fill be filled with the
content of the field “type” of the “DCMI” table if present </xs:documentation>
                                </xs:annotation>
                        </xs:element>
                </xs:sequence>
        </xs:complexType>
</xs:element>
<xs:element name="AIIReport">
        <xs:annotation>
                <xs:documentation>Root element of internal AII report</xs:documentation>
        </xs:annotation>
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="reportDate">
                                <xs:annotation>
                                        <xs:documentation>Range of dates to which the report
refers</xs:documentation>
                                </xs:annotation>
                                <xs:complexType>
                                        <xs:sequence>
                                                <xs:element name="startDate" type="xs:dateTime">
                                                        <xs:annotation>
```

```
                                                                    <xs:documentation>starting date/time of the
range</xs:documentation>
                                                                    </xs:annotation>
                                                            </xs:element>
                                                            <xs:element name="endDate" type="xs:dateTime">
                                                                    <xs:annotation>
                                                                            <xs:documentation>ending date/time of the
range</xs:documentation>
                                                                    </xs:annotation>
                                                            </xs:element>
                                                    </xs:sequence>
                                            </xs:complexType>
                                    </xs:element>
                                    <xs:element ref="AIIRecord" minOccurs="0" maxOccurs="unbounded"/>
                            </xs:sequence>
                    </xs:complexType>
            </xs:element>
</xs:schema>
```

8.2 Formal description of format for statistics record

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
        <!-- version 1.0 (30-Mar-2006)
-->
        <xs:element name="CamartRecord">
                <xs:annotation>
                        <xs:documentation>Root element for a Camart record</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                        <xs:sequence>
                                <xs:element name="AXCSID" type="xs:string">
                                        <xs:annotation>
                                                <xs:documentation>log ID</xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                                <xs:element name="executionTimestamp" type="xs:dateTime">
                                        <xs:annotation>
                                                <xs:documentation>timestamp of the execution of the
Transaction</xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                                <xs:element name="registrationTimestamp" type="xs:dateTime">
                                        <xs:annotation>
                                                <xs:documentation>Timestamp of the registration of the Transaction in
AXCS</xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                                <xs:element name="AXOID">
                                        <xs:annotation>
                                                <xs:documentation>ID of the object inside the axmedis
system</xs:documentation>
                                        </xs:annotation>
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:maxLength value="40"/>
                                                        <xs:whiteSpace value="collapse"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>
                                <xs:element name="AXDID">
                                        <xs:annotation>
```

```
                                                <xs:documentation>Unique AXMEDIS Object Distributor ID that has performed the transaction</xs:documentation>
                                        </xs:annotation>
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:maxLength value="40"/>
                                                        <xs:whiteSpace value="collapse"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>
                                <xs:element name="AXCID" type="xs:string">
                                        <xs:annotation>
                                                <xs:documentation>Creator Id</xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                                <xs:element name="operation" type="xs:string">
                                        <xs:annotation>
                                                <xs:documentation>Use of the content (reading, printing, aggregating, editing).</xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                                <xs:element name="location" type="xs:string">
                                        <xs:annotation>
                                                <xs:documentation>Geographical area in which the object is used.</xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                                <xs:element name="ISRC" type="xs:string" minOccurs="0">
                                        <xs:annotation>
                                                <xs:documentation>Factory Infromation: Unique standard ID for identifying the piece</xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                                <xs:element name="objectCode" type="xs:string" minOccurs="0">
                                        <xs:annotation>
                                                <xs:documentation>Factory Information: this field fill be filled with the content of the field “type” of the “DCMI” table if present </xs:documentation>
                                        </xs:annotation>
                                </xs:element>
                        </xs:sequence>
                </xs:complexType>
        </xs:element>
        <xs:element name="CamartReport">
                <xs:annotation>
                        <xs:documentation>Root element of internal Camart report</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                        <xs:sequence>
                                <xs:element ref="CamartRecord" minOccurs="0" maxOccurs="unbounded"/>
                                <xs:element name="Statistics" minOccurs="0" maxOccurs="unbounded">
                                        <xs:complexType>
                                                <xs:sequence>
                                                        <xs:element name="AXOID" minOccurs="0" maxOccurs="unbounded">
                                                                <xs:annotation>
                                                                        <xs:documentation>ID of the object inside the axmedis system</xs:documentation>
                                                                </xs:annotation>
                                                                <xs:complexType>
                                                                        <xs:simpleContent>
                                                                                <xs:extension base="xs:string">
                                                                                        <xs:attribute name="count" type="xs:int" use="required"/>
                                                                                </xs:extension>
                                                                        </xs:simpleContent>
```

```
                                                                </xs:complexType>
                                                        </xs:element>
                                                </xs:sequence>
                                                <xs:attribute name="type" use="required">
                                                        <xs:simpleType>
                                                                <xs:restriction base="xs:string">
                                                                        <xs:enumeration value="TopChart"/>
                                                                        <xs:enumeration value="BottomChart"/>
                                                                </xs:restriction>
                                                        </xs:simpleType>
                                                </xs:attribute>
                                                <xs:attribute name="cardinality" type="xs:int"/>
                                                <xs:attribute name="restrictBy" type="xs:string"/>
                                        </xs:complexType>
                                </xs:element>
                        </xs:sequence>
                </xs:complexType>
        </xs:element>
</xs:schema>
```

9 Demostrator Fact Sheet

9.1 Main purposes of the demonstrator

Main purposes of Core Accounting manager and Reporting Tool (CAMART) is strictly bound with database for logs (provided by AXCS) since it has to collect information regarding the B2B activities and B2C actions. AXCS will not store forever its logs and therefore it is necessary for CAMART to gather time by time such logs and store locally in the AXMEDIS database. Such information will be collected on scheduled time interval and CAMART will act as a client of the AXCS Reporting Web Service.
AXMEDIS system is scalable and therefore we have to deal with the fact that some installation can have AXDB, AXCS and other supporting tools on different machines, while others can be less distributed due to a lesser need for speed or storage capacity.
The core accounting manager is a sort of Client side of the bridge between the AXDB and the AXCS databases in order to allow AXCS to be independent by the database. The server side in the AXCS is the Web Service: AXMEDIS Reporting Web Service. The CAMART can be interpreted as a part of the AXMEDIS Database Interface, since is the part of the system that allows writing data related to Action-Logs into the AXMEDIS DB.

Administrative Information integrator is a critical part of the AXMEDIS system since it is the real bridge between the AXMEDIS world and the world of company's CMS and CRM for taking in account administrative and legal aspects (such reclaim for payment not done and so on).
Main purposes of this component is to operate in a dual manner: used for polling information from AXMEDIS system when needed by distributor for example, or used for pushing information in the CMS as soon as they are available for example in the case of collecting societies.
The AII can also completely remotely managed by a Web Service that offers all the functionalities guaranteed by the web application.
The operating mode is determined by accounting people during the installation/configuration of the system when it will be established whose fields have to be exported from the DB to the CMS and the frequency of exporting. When a frequency is set, the Administrative Information Integrator will work in push mode, pushing information in the CMS import area, otherwise it operates in polling mode by starting the update in the CMS by a link to a web page.

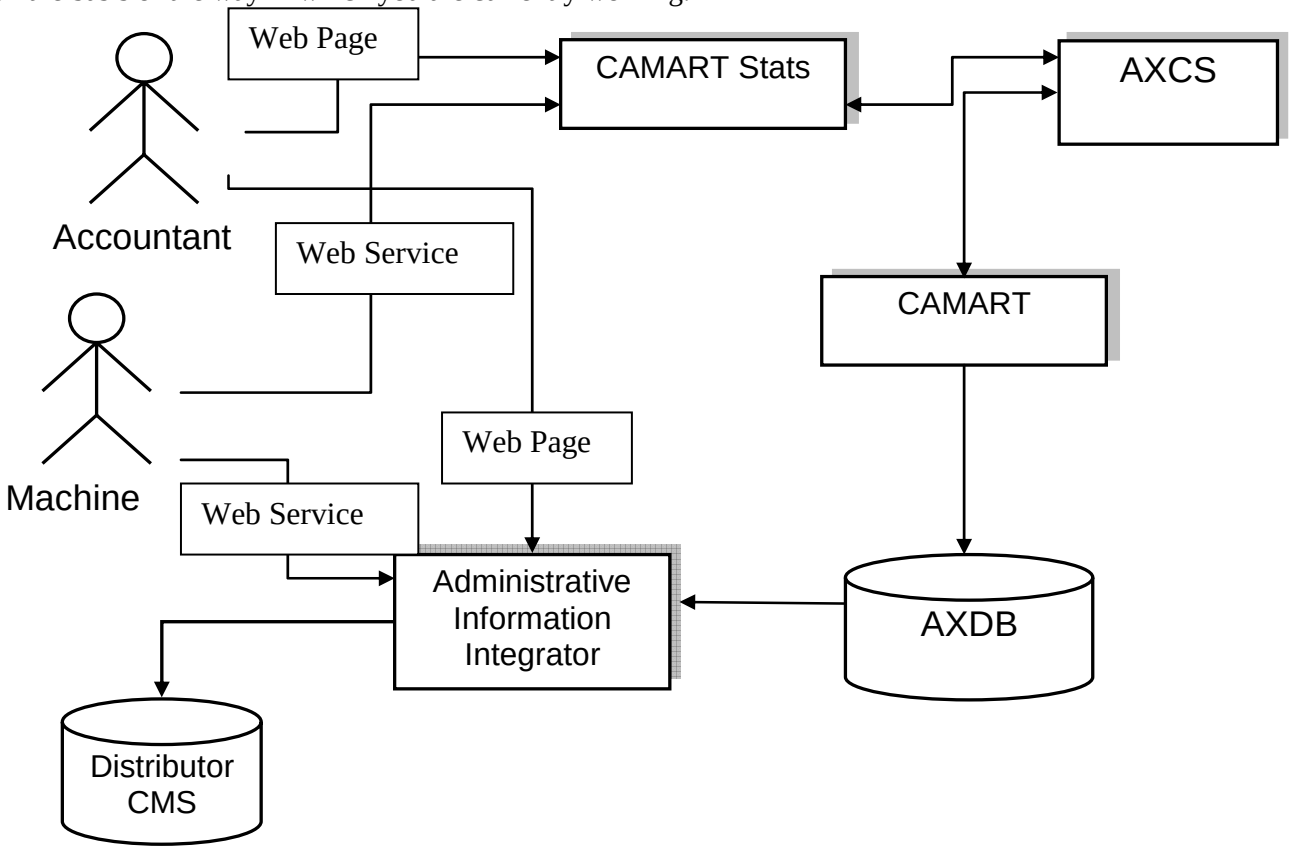
The role of Core Accounting manager and Reporting Tool (CAMART) for Statistics is strictly bound with database for logs (provided by AXCS) since it has to gather information from AXMEDIS Certifier and Supervisor about Action Log and provide them to the user via web page or web service interface.

By using this demonstrator you can:

- Get logs coming from AXCS web service
- Organize such logs in the internal database of the factory
- Generate internal XML format in polling mode
- Generate the XML format provided by one of the partners according to the specification and publishing at a predefined time frequency the resulting XML in an ftp directory or file path
- Generate also in polling mode an XML format that is transformed according to the profiled XSLT.
- Generate top-bottom ten on demand for statistics.
- Use a web service for gathering statistics instead of the GUI only
- Use a Web service to remotely control AII completely in order to be able to automate log collection

9.2 Review of the architecture integration with AXMEDIS

CAMART and related tool are strictly bounded with some AXMEDIS components: the AXCS and the AXDB.
The relationships among CAMART and the underlying AXMEDIS components are depicted in the following picture.
All the tools can be used by physical person (Accountant in the picture) by a simple Web Interface or by an automated tool (Machine in the picture) by a web service interface.
This allows as easy integration in your factory allowing you to select the best approach for event logs access on the basis of the way in which you are currently working.



9.3 Description of the effective installation

The demonstrator is a set of 3 web application distributed as WAR that can be deployed directly (after the initial configuration) on top of Tomcat >5.5.20. They must have a connection with AXMEDIS database that can be in the same server or in a different one.
It is needed therefore for a full installation to have at least a server with Mysql 4.x and Tomcat 5.5.y where y is greater or equal to 20.
This server must be installed with an AXDB but it is not necessary that you install AXDB related services and web services, the database itself is enough. After that you can deploy on Tomcat the web application of Camart, CamartStats and AII.

9.4 AXMEDIS tools

The AXMEDIS tools used by this demonstrator are:

- AXMEDIS Database for storing and retrieving logs and configurations
- AXCS for retrieving logs and statistics by the means of the exposed Reporting and Statistic WS

9.5 Target Market

This demonstrator is strictly integrated with AXMEDIS and therefore we have decided to publish the source code inside the AXMEDIS framework in order to bind it with the platform.
The target market is the AXMEDIS factory and therefore it is necessary that its deployment and usage are connected with the installation of the framework.
Moreover since several future AXMEDIS user can use different CMS technologies, the possibility to perform training on the logs format customization or the writing of log converter will arise.

9.6 Description of the business model

Installed with the AXMEDIS framework when a factory is set-up comprised with the AXMEDIS framework fee or as an additional fee to be paid.

9.7 Description of content

Involved contents are AXCS logs and administrative data, but no multimedia content are involved.

9.8 Final Users/Clients

These tools are not for final user, they are tool for people operating in the factory.

9.9 Partners involved and roles

DSI: for providing AXCS

10 Usage of Camart/CamartStats on WP9 and WP12 demonstrators

10.1 BBC, EUTELSAT, TELECOM

Have you used CAMART in your demonstrator	NO
How easy was CAMART integration? (from 1 to 5, where 1 means very easy, 5 very difficult)	
How many records are present in the actionlog table of the AXDB linked to CAMART	
Have you used CAMARTSTATS in your demonstrator	NO
How easy was CAMARTSTATS integration? (from 1 to 5, where 1 means very easy, 5 very difficult)	
How many record do you obtain from CAMARTSTATS in your most complete query	

Not Used

10.2 ILABS

Have you used CAMART in your demonstrator	YES
How easy was CAMART integration? (from 1 to 5, where 1 means very easy, 5 very difficult)	2
How many records are present in the actionlog table of the AXDB linked to CAMART	294
Have you used CAMARTSTATS in your demonstrator	YES
How easy was CAMARTSTATS integration? (from 1 to 5, where 1 means very easy, 5 very difficult)	2
How many record do you obtain from CAMARTSTATS in your most complete query	8677

10.3 TISCALI

Have you used CAMART in your demonstrator	NO
How easy was CAMART integration? (from 1 to 5, where 1 means very easy, 5 very difficult)	
How many records are present in the actionlog table of the AXDB linked to CAMART	
Have you used CAMARTSTATS in your demonstrator	YES
How easy was CAMARTSTATS integration? (from 1 to 5, where 1 means very easy, 5 very difficult)	1
How many record do you obtain from CAMARTSTATS in your most complete query	> 100

10.4 OTHER PARTNER

No answer has been received from other partners

11 Bibliography

DE 9.1.1 Specification of CMS integration and feedback
DE 9.1.2 Mock-up for CMS Integration and Feedback
DE9.1.3 First Prototype for CMS Integration and Feedback
DE 2.1.1.2.1 User Requirement First Update
DE 2.1.1.2.2 Use case First Update
DE3-1-2-2-15 Specification of AXMEDIS Accounting and Reporting
Video Demonstration: http://www.axmedis.org/documenti/view_documenti.php?doc_id=2131