Project no. 033572

CASPAR

Cultural, Artistic and Scientific knowledge for Preservation, Access and Retrieval

Instrument: Information Society Technologies

Thematic Priority: 2.5.10 Access to and preservation of cultural and scientific resources

CASPAR Overall Component Architecture and Component Model

Document identifier:	**CASPAR-D1301-TN-0101-1_1**
Submission Date:	**17-05-2007**
Due Date:	**15-05-2007**
Work package:	**1300**
Partners:	**All Partners**
WP Lead Partner:	**ENG**
Document status	**FINAL**

Abstract: This document details of the architecture of the **CASPAR** components which were described in the Conceptual Model [**D1201**].

Delivery Type	Report
Author(s)	**CASPAR** Consortium
Approval	David Giaretta
Summary	
Keyword List	
Availability	☒ **PUBLIC**

Document Status Sheet

Issue	Date	Comment	Author
0_0	5 May 2007	Initial draft bringing together contributions	David Giaretta
0_1	8 May 2007	Adding revised contributions for KM, FindAids, DAMS, Authenticity	Luigi Briguglio
0_2	9 May 2007	Revision of POM	Luigi Briguglio
0_3	9 May 2007	Added missing picture; Revision of POM, DAMS and DRM, Deployment, Application Usage Added References	Pasquale Andriani, Luigi Briguglio
0_4	10 May 2007	Revision of Chapter 4 and Chapter 5: added component models according to descriptions	Luigi Briguglio
0_5	11 May 2007	Revision by DG, revision of Authenticity use cases and descriptions according to UU contribution,	Luigi Briguglio
0_6	13 May 2007	Addition of text to clarify a number of points	David Giaretta
0_7	14 May 2007	Added CASPAR Architecture Paper section, Integration of Chapter 2, Revision of Chapter 5, Revision of Chapter 6 Added Glossary	Luigi Briguglio
0_8	15 May 2007	Revision with feedbacks from internal peer-review	Luigi Briguglio
0_9	15 May 2007	Revision of styles and formats	Pasquale Andriani
1_0	15 May 2007	Final version	David Giaretta
1_1	17 May 2007	Correct minor typos	David Giaretta

Project information

Project acronym:	**CASPAR**
Project full title:	Cultural, Artistic and Scientific knowledge **for Preservation, Access and Retrieval**
Proposal/Contract no.:	**IST-2006-033572**

Project Officer: Carlos Oliveira

Address:	INFSO-E3 Information Society and Media Directorate General Content - Learning and Cultural Heritage Postal mail: Bâtiment Jean Monnet (EUFO 1167) Rue Alcide De Gasperi / L-2920 Luxembourg Office address: EUROFORUM Building - EUFO 1167 10, rue Robert Stumper / L-2557 Gasperich / Luxembourg
Phone:	+352 4301 33052
Fax:	+352 4301 33190
Mobile:	
E-mail:	Carlos.Oliveira@ec.europa.eu

Project Co-ordinator: David Giaretta

Address:	STFC (formerly CCLRC), Rutherford Appleton Laboratory Chilton, Didcot, Oxon OX11 0QX, UK
Phone:	+44 1235 446235
Fax:	+44 1235 446362
Mobile:	+44 (0) 7770326304
E-mail:	d.l.giaretta@rl.ac.uk

CONTENT

1 INTRODUCTION

The objective of this deliverable is to present a high level view of the design of the overall component **CASPAR** architecture. This design follows the Conceptual Model closely, with additional input from the User Requirements and Scenario Specifications [**D4101**].

The architectural process does not follow a pure waterfall model but is being developed, along with the Conceptual Model, in an iterative process in which the phased implementation influences both the Architecture and Conceptual Model, as described in the **CASPAR** Guidelines **[D1202]**.

This document provides the overall architecture design, the key component model, and the first level of detail in the class diagrams, interfaces and specification of interactions. Preliminary data elements, user interfaces, outputs and other interfaces are also described.

The **CASPAR** Guidelines document [**D1202**] points to the shared UML repository (used by the Enterprise Architect[1] tool), where much greater detail, for everything from Requirements, Use Cases, Component design, Class diagrams to source code, is available.

The core of this deliverable is represented by the following chapters:

- *Architecture Process* describes the process to manage and control the **CASPAR** Architecture;
- *OAIS-based Design* describes an overview on OAIS-based systems, one of the main requirements of **CASPAR** Architecture. In this chapter the responsibilities of each OAIS functional component are highlighted, consistent with the OAIS Reference Model;
- *Key Preservation Components Use Cases* describes in more details "OAIS-based Use Cases" and defines "Key Preservation Components Use Cases" for the **CASPAR** Architecture;
- *Key Preservation Components Description* describes how OAIS responsibilities and Key Preservation Use Cases are satisfied by **CASPAR** Architecture Key Preservation Components;

Furthermore, in order to provide the most complete information for the implementation phase, this deliverable addresses the following topics:

- *Implementation Guidelines* describes some basic guidelines to be followed during application development;
- *High Level Scenarios* describes how the components work together in **CASPAR**;
- *Deployment* reports some example of deployment configuration.

The deliverable is completed by:

- *References*, which provides a reading list used during the preparation of this deliverable.
- *Glossary* represents the important phase of fixing terminology encountered during this first phase of the **CASPAR** Project.

1.1 HOW TO READ THIS DOCUMENT

[1] http://www.sparxsystems.com.au/

This document assumes that the reader is familiar with the Open Archival Information Systems (OAIS) Reference Model. It is also essential to have read the **CASPAR** Guidelines [**D1202**] and the **CASPAR** Conceptual Model [**D1201**]. Some familiarity with the **CASPAR** Description of Work [**DoW**] and some familiarity with UML would be useful.

A high level view may be obtained from sections 3, 4, 7 and 8, while more details of the components are to be found in sections 5 and 6, and, as noted above, full details are available from the project's shared UML repository and its Web-based browsable version – both described in the **CASPAR** Guidelines [**D1202**].

1.2 CASPAR ARCHITECTURE PAPERS

The **CASPAR** Consortium has produced and submitted some scientific papers which contain many topics analysed and described in this deliverable.

They represent an important result and reference of the **CASPAR** WorkPackage 1300 Overall Architecture. Other papers are planned or in preparation.

Id	Y.Tzitzikas'2007
Author(s)	Yannis Tzitzikas
Title	Dependency Management for the Preservation of Digital Information
Keywords	Digital Information Preservation; Representation Information Gap; Knowledge Management
Reference	18th International Conference on Database and Expert Systems Applications, DEXA'2007, Regensburg, Germany, September 2007

Id	IBM'2007
Author(s)	Michael Factor, Dalit Naor, Simona Rabinovici-Cohen, Leeat Ramati, Petra Reshef, and Julian Satran
Title	The Need for Preservation Aware Storage - A Position Paper
Keywords	Digital Information Preservation; Preservation Aware Storage
Reference	ACM SIGOPS Operating Systems Review, Special Issue on File and Storage Systems, Volume 41, Issue 1 (January 2007), pages 19-23.

The above submitted papers are attached in the appendix of this deliverable.

1.3 APPLICABLE DOCUMENTS AND REFERENCE DOCUMENTS

Applicable documents

[A1]	Description of Work, April 2006
[A2]	Risk Form

Reference documents

[R1]	**CASPAR** proposal, Sept 2005
[DoW]	**CASPAR** Description of Work
[D1101]	D1101 Review of the State of the Art
[D1201]	D2101 **CASPAR** Conceptual Model
[D1202]	D1202 **CASPAR** Guidelines
[D4101]	D4101 User Requirements and Scenario Specifications

1.4 GLOSSARY

[Ax]	Applicable Document
[Rx]	Reference Document
CASPAR	Cultural, Artistic and Scientific knowledge for Preservation, Access and Retrieval
DoW	Description of Work
EC	European Commission
EPM	Executive Project Management
IPC	IP Coordinator
IST	Information Society Technologies
PACP	Partner Administrative Contact Point
PO	Project Officer
PPR	Project Progress Report
PQE	Project Quality Engineer
PTCP	Partner Technical Contact Point
RegRep	Registry/Repository
R&D	Research and Development
SQE	Stream Quality Engineer
ST	Stream
TN	Technical Note
WP	Work Package
WPL	Work Package Leaders

2 THE ARCHITECTURE PROCESS

Design and programming are human activities; forget that and all is lost.

[Bjarne Stroustrup, 1991]

According to many references in the technical literature (see section 9), an architecture is a description (sometime termed an abstraction) of the run-time elements of a software system. That architecture has to describe and justify how requirements and needs are satisfied.

The Architecture should flow from a Conceptual Model which has captured solutions to user as well as functional and non-functional requirements which come from many sources. In the case of the CASPAR Conceptual Model the OAIS Reference Model provides such requirements as well as a large number of important concepts.

The Architecture has to identify parts of the software system, commonly called software components, each having specific responsibilities, which work together in order to produce the expected results and provide the required functionality. The description comprises the details of the software components including their externally visible properties and the relationships between them.

The **CASPAR** Overall Component Architecture and Component Model is described using the Unified Modelling Language (UML), a standard defined by the Object Management Group (OMG – see http://www.uml.org/).

UML, version 2.1.1 at the time of writing, provides a framework to model and formalise specifications for application structure, behaviour, architecture, business process and data structures.

Along with the Meta Object Facility (MOF), UML provides a key foundation for Model-Driven Architecture (MDA), which attempts to provide a way to unify all aspects of development and integration from business modelling, through architectural and application modelling, to development, deployment, maintenance, and evolution.

While not a panacea, MDA does provide us with certain level of independence from the current implementation specifics; tools can generate much of the code to suit other programming languages and other support frameworks.

There are a number of other general features which we require from the Architecture, much of which is now quite standard software development technique. For example we aim to have

- a clear definitions of the interfaces involved – which can then be implemented in a variety of ways (as with MDA).
    - This is an advantage from the point of view of preservation because we are then not tied to a specific implementation, or specific programming languages, which would otherwise need to be maintained over time.
- loose-coupling of components – which allows a great flexibility of deployment.
    - From a preservation point of view this means that we are not reliant on a rigid structure which must be maintained over time.

- no central point of failure
    - from a preservation point of view makes sense because no single institution can be relied on in the long term. In particular the **CASPAR** project will certainly not itself exist over the long term.

The development process, as discussed in the CASPAR Guidelines [**D1202**], does not follow a pure waterfall model but is an iterative process where lessons learned during one implementation cycle affect the set of requirements which in turn influences the Architecture and hence the next implementation cycle.

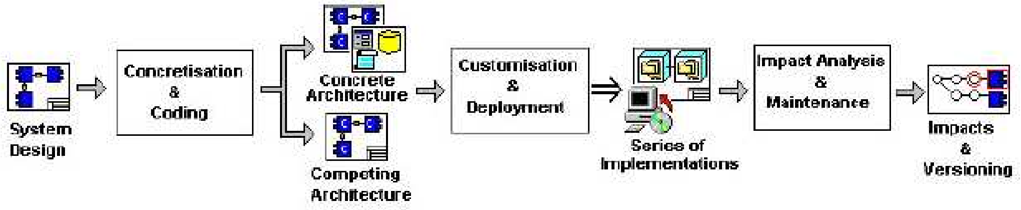
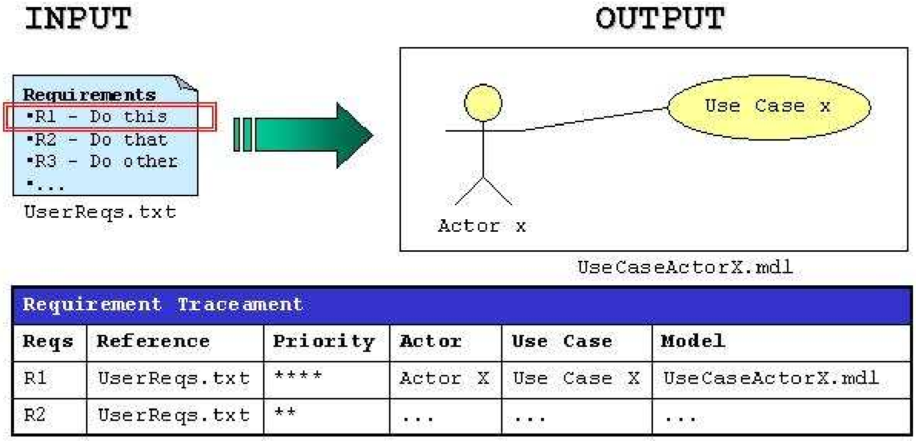


Figure 1 : From System Design to Versioning

For that reason it is important that we have a method of tracing the link between the requirements and the architectural components.

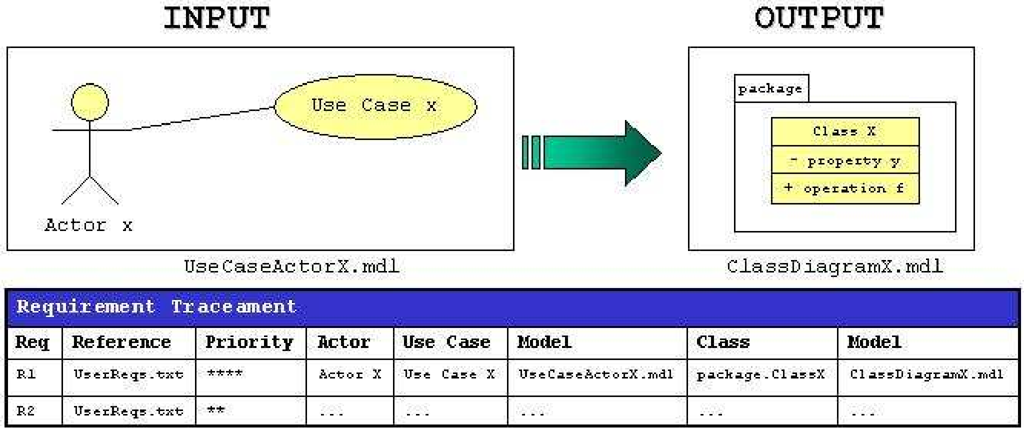
Enterprise Architect, the tool adopted in **CASPAR** for the modelling activities (i.e. mainly WP1200 **CASPAR** methodology and models and WP1300 Overall Architecture), allows the management of the relationships between Requirements, Actors, Use Cases, Class Diagrams and Component Diagrams.



Requirement Traceament

Reqs	Reference	Priority	Actor	Use Case	Model
R1	UserReqs.txt	****	Actor X	Use Case X	UseCaseActorX.mdl
R2	UserReqs.txt	**	...	...	...

Figure 2 : Requirements - Use Cases Traceability



Requirement Tracement

Req	Reference	Priority	Actor	Use Case	Model	Class	Model
R1	UserReqs.txt	****	Actor X	Use Case X	UseCaseActorX.mdl	package.ClassX	ClassDiagramX.mdl
R2	UserReqs.txt	**	...	...	...	...	...

Figure 3 :Use Case - Classes/Components Traceability

3 HIGH LEVEL SCENARIOS

The following scenarios should help to provide an overall view of the use of the **CASPAR** components. These are actually a single scenario viewed in increasing detail, using OAIS and [**D1202**] terminology.

3.1 SIMPLEST SCENARIO VIEW

At the most basic level the scenario consists of the following steps:

- someone or something (Producer in OAIS terminology) creates some digitally encoded information
- the digital object is stored somewhere – an OAIS archive or repository - which takes care of it and in particular ensures that the information which is encoded in the bits remains understandable (to a defined Designated Community) over the long term
- at various points in time users (Consumer in OAIS terminology) request the data and try to use it.

3.2 INTERMEDIATE LEVEL SCENARIO VIEW

Looking at the next level of detail, we can break down these steps and describe where **CASPAR** is relevant.

ACTIVITY	**CASPAR involvement**
1: some digital object is created – let us for the sake of clarity assume this is a piece of data from a scientific instrument	In principle the creation of the data does not involve **CASPAR** infrastructure, however increasingly the data capture integrates the capture of metadata and if that is the case then **CASPAR** techniques come in to play.
2: the data is deposited in an archive for long term preservation, for a specified Designated Community	In order to be deposited in an archive some prior arrangements presumably will have been made, in which the appropriate Representation Information and Provenance, Packaging, Packaging Descriptions and the other OAIS related information will need to be created. **CASPAR** tools will help in this process. For example • the RepInfo Toolbox will guide the creation of Representation Information • a RepInfoLabel may be created, pointing to existing applicable Representation Information in a Registry/Repository known to the user or recommended by the archive • new RepInfo may be created using a number of tools, for example - a bit level description may be created - the meanings associated with the numbers can be captured

	and a new data dictionary created, perhaps associated with a discipline ontology - a view of sections of the data as a table may be created with the Virtualisation toolkit - the relationship of the various component parts of the data product can be described All this Representation Information may be packed with the data and/or some or all of it may be deposited in a Registry/Repository and the CPID packed with the data or its RepInfoLabel. • the PDI toolbox will help in the capture of Provenance, Fixity, Reference and also Context. The appropriate evidence for to support the authenticity of the data may be captured as part of this. • the DRM tools will enable the rights to be described in a preservable way
3: The archive creates an Archival Information Package and collects the appropriate (perhaps quite dataset specific) data management information to assist its internal management and external Finding Aids etc.	The **CASPAR** Packaging tools help to create the AIP, either as a single file or a logical object, for example a manifest which points to the various AIP components. The Representation Information may be packaged with the data or a Registry/Repository may be referenced. In any case we assume that the AIP contains a pointer to further Representation Information. If the piece of data is part of a larger data collection the Collection Information is collected. The AIP is stored in the Preservation Data Store (PDS), which maintains the integrity and technical Provenance of the data over time. Appropriate preservation policies and data virtualisation techniques are used.
4: Over time various preservation actions must be taken	
a. the data is passed from one hardware/software system within the archive	The PDS virtualisation techniques support changes in hardware. The Orchestration manager may play a role by distributing alerts about obsolescence of hardware. Changes in software systems are assisted by being able to pass on the preservation policies to the new systems.
b. it is passed from one archive to another, in a chain of preservations	The **CASPAR** AIP structures, collection descriptions, PDS namespace virtualisation and preservation policy descriptions support the handing on of the whole collection. The Orchestration manager may have played a role

	here in “match-making”, allowing the original archive to alert others that it needed to hand on its holdings.
c. it is migrated from one format to another	The Orchestration Manager may, for example, distribute alerts about obsolescence of formats for particular Designated Communities. The Representation Information may be used in generic applications to migrate from the initial format to another. Alternatively if special applications are used then the Representation Information for the new format must be created and pointed to by the RepInfoLabel.
d. evidence of its authenticity is accumulated, as is other Provenance information	Technical provenance e.g. about copying or moving will be recorded by the PDS. Other evidence supporting authenticity will be integrated using the **CASPAR** authenticity tools.
e. additional Representation Information is accumulated as the knowledge base of the Designated Community changes	The Orchestration component allows sharing of information about changes in the knowledge base of various designated communities. The RepInfo Gap Manager may determine the implied need for yet other RepInfo. People who have indicated their expertise in a particular area may be requested to create the required RepInfo.
5: At some point a user - let us assume for simplicity s/he is a member, in the sense of having the appropriate knowledge base, of the Designated Community – requests a copy of the data:	The current archive will no doubt have its own access system, but this may be supplemented by the **CASPAR** Access component to allow additional sophisticated querying to locate data.
a. the appropriate Access Controls and Rights Management is checked	The DRM information will have been migrated over time and will be used to match the current Rights Management environment.
6: On receipt of a copy of the data, the user wishes to examine it in a scientific application	Scientific applications will almost certainly have changed significantly since the time the data was originally collected and analysed. The data may have been migrated over time to some format suitable for the current favourite tool of the Designated Community. This would allow display and manipulation. The associated Data Dictionaries and ontologies would provide the meaning associated with each number within the data object. Alternatively the Representation Information may consist simply of a document describing the way in which the information is encoded, including the meaning and relationships between the data elements. In this case the user would have to create appropriate software, based on the document.

	Midway between these extremes is Representation Information which captures the data description using a formal language, or set of languages. In this case the associated interpreters could be used to extract, for example, numbers or whole images or tables from the data and these could be used in the application. For example an application which deals with Tables may recognise from the Representation Information that there is a virtualisation description which allows the data to be looked at as a table. The application could then take this virtualisation and look at those tabular aspects of the data.

4 OAIS-BASED DESIGN

The OAIS Reference Model is, by its own admission, not a design, nevertheless it provides a number of models (Functional, Information, Data Flow) on which we can build in order to produce some of the fundamental design concepts (classes in Object-Oriented terminology) for the **CASPAR** Architecture.

Putting together aspects of the OAIS data flow diagrams, the typical Digital Preservation scenario underlying the OAIS Reference Model (OAIS), together with related activities, which have to be performed in order to preserve digital heritage object for a long term may be summarised in those 8 steps (see also Figure 5):

1. Select **Content Objects** to be preserved over the long term;
2. Add **Information** in order to support Objects maintenance (e.g. transformation). That may be done in 3 ways:
    - Add Information manually
    - Add Information by using a "wizard"
    - Add Information automatically extracted from Objects
3. Create Submission Information Package (**SIP**)
4. Import/Ingest SIP. Ingestion includes the following 3 activities:
    - Check data completeness;
    - Check formal correctness;
    - Transform one or more SIPs into Archival Information Package (**AIP**), including;
        1. Representation Information
        2. Preservation Description Information
        3. Packaging Information
        4. Descriptive Information
5. Store AIP in the **Archival Storage** System;
6. Transfer Descriptive Information into the **Data Management** System;
7. AIP and relative Information are accessed via Archival Storage and Data Management System;
8. AIP and relative Information are delivered as Dissemination Package (**DIP**) via **Access** System.

OAIS defines 6 "functional components" and 3 main actors for a Preservation System:

Functional Components

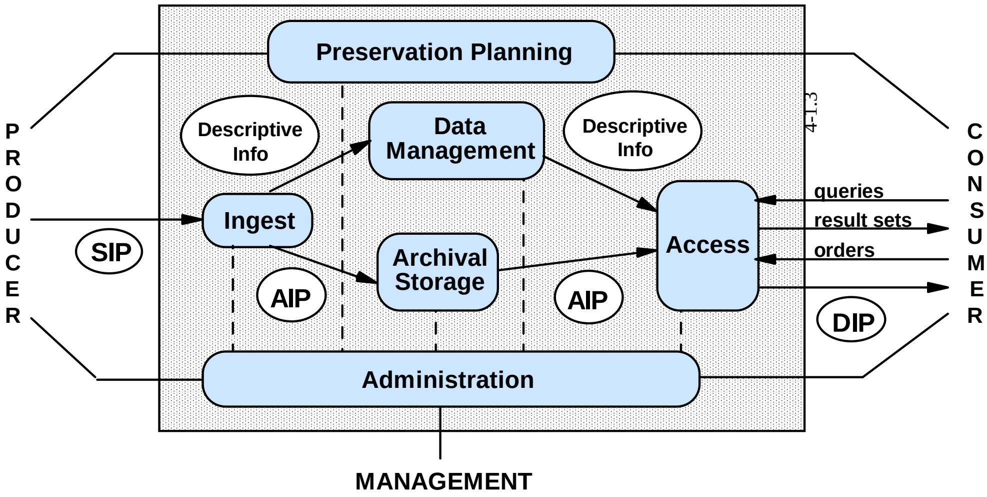


Figure 4 OAIS Functional Model

1. Ingest
2. Access
3. Archival Storage
4. Data Management
5. Administration
6. Preservation Planning

Main Actors

1. Producer
2. Management/Administrator
3. Consumer

In this section the main responsibilities of the OAIS Functional Entities are described. More details of specialised aspects are provided in subsequent sections, in particular for Archival Storage (see section 6.5) and Access (see section 6.8). The section on Packaging (see section 6.10) describes a number of concepts used throughout the other components, while sections 6.1 and 6.3 provide details of the infrastructure components described in the Conceptual Model.

The flow of Information Packages (IPs) through the OAIS is shown in Figure 5 as a UML process diagram which highlights the processing of IPs as they pass from producer to consumer. In summary, a SIP is constructed by the Producer, from Data and Information Objects, this is validated and delivered to the Archive for Ingest. At the point of Ingest the SIPs are received, they may need to be buffered and re-order into the correct sequence as a Set of SIPs. The Set of SIPs will be transformed to construct an AIP, at this point additional Information Objects such as PDI and RepInfo may be added. The newly constructed AIP will be validated for completeness and stored for long term preservation. At the point of Dissemination a consumer will make some request to receive a DIP. The appropriate AIPs will need to be retrieved from storage and transformed into the required DIP. The DIP should ideally be validated for correct structure and content and then delivered to the consumer.

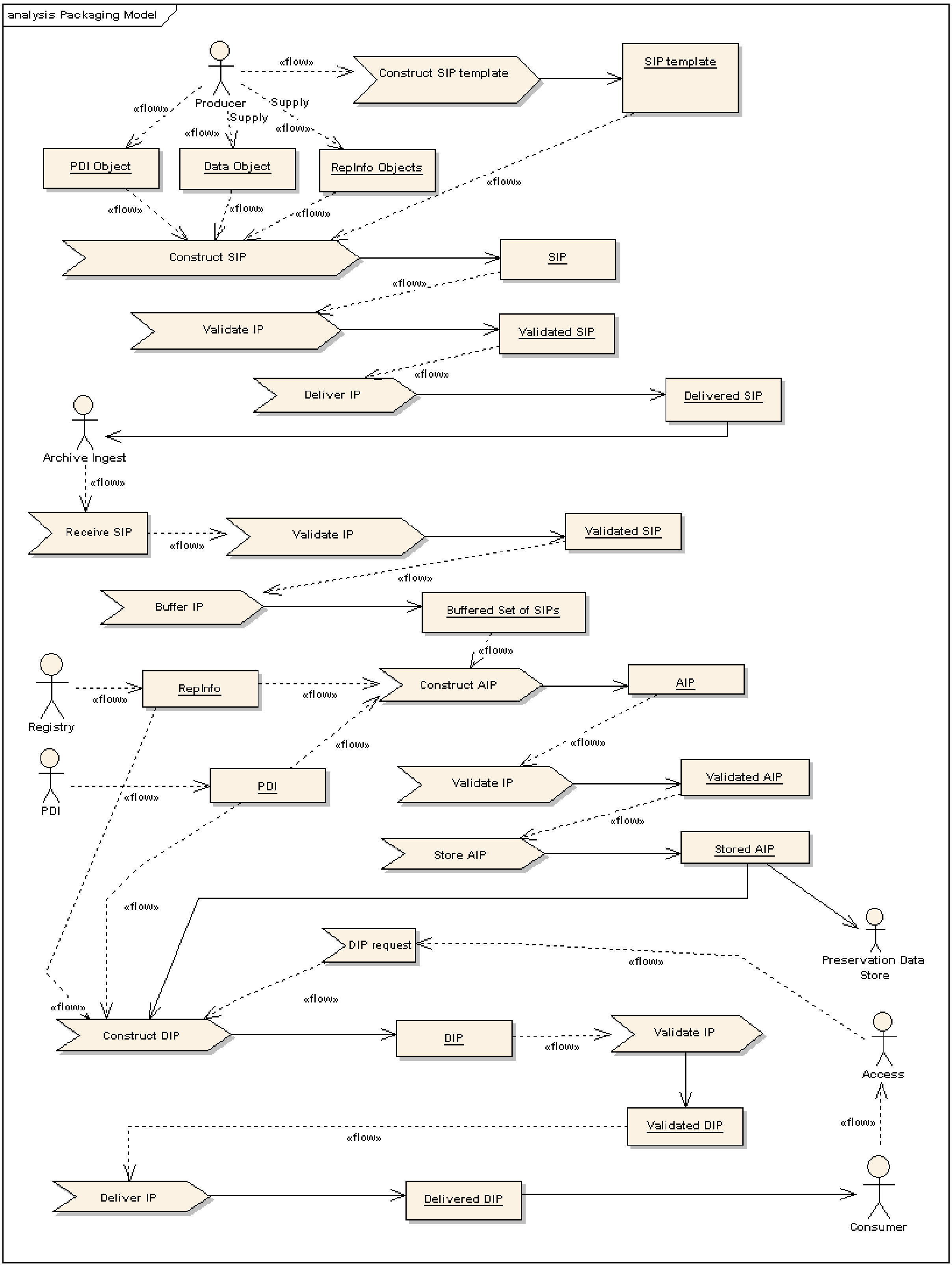


Figure 5 OAIS Process View

4.1 OAIS FUNCTIONAL MODEL

Each of the Functional Entities is described in turn; note that these are “logical” entities and need not be implemented as single components.

4.1.1 Ingest

The OAIS entity that contains the services and functions that accept Submission Information Packages from Producers, prepares Archival Information Packages for storage, and ensures that Archival Information Packages and their supporting Descriptive Information become established within the OAIS.

Ingest has the following responsibilities:

- Accept SIP submitted by producer
    - Receive SIP
    - Perform quality assurance on SIP
- Prepare contents for storage and management within the Archive
    - Generate AIP compliant with Archive Data Formatting and Documentation Standard
    - Extract Descriptive Information from AIP for inclusion in the archive database
    - Coordinate updates to Archival Storage and Data Management

4.1.2 Access

The OAIS entity that contains the services and functions which make the archival information holdings and related services visible to Consumers.

Access has the following responsibilities:

- Query Processing
    - Forward request to Data Management
    - Response Presentation (Result Set)
- Retrieval and Delivery of requested content as a DIP
    - Forward to Archival Storage
- Perform any necessary transformations (suitable for Dissemination)
- Security and Access Control

4.1.3 Archival Storage

The OAIS entity that contains the services and functions used for the storage and retrieval of Archival Information Packages.

Archival Storage has the following responsibilities:

- Storage of AIP
    - Receive AIP
    - Add AIP to Permanent Storage
    - Manage storage hierarchy

- Maintenance of AIP
    - Refresh media on which archive holdings are stored
    - Perform routine and special error checking
    - Provide disaster recovery capabilities
- Retrieval of AIP
    - Provide AIP to Access to fulfil orders

4.1.4 Data Management

The OAIS entity that contains the services and functions for populating, maintaining, and accessing a wide variety of information. Some examples of this information are catalogues and inventories on what may be retrieved from Archival Storage, processing algorithms that may be run on retrieved data, Consumer access statistics, Consumer billing, Event Based Orders, security controls, and OAIS schedules, policies, and procedures.

Data Management has the following responsibilities:

- to populate the Descriptive Information, which identifies and documents archive holdings, and Administrative Data, used to manage the archive
- to maintain the Descriptive Information and Administration Data
    - Update DBs as new information arrives or existing information is modified or deleted
- to access the Descriptive Information and Administration Data
    - Perform queries on DBs
    - Generate Reports in response to requests from other functional components within OAIS

4.1.5 Administration

The OAIS entity that contains the services and functions needed to control the operation of the other OAIS functional entities on a day-to-day basis.

Administration has the following responsibilities:

- Overall Archive Administration:
    - Solicit and negotiate submission agreements with Producers
    - Audit submission to ensure they meet archive standards
    - Maintain configuration management of system hw and sw
    - Provide system engineering functions to:
        - Monitor and improve archive operations
        - Inventory content of the archive
        - Report on content of the archive
        - Migrate/update content of the archive
    - Establish and Maintain archive standards and policies

4.1.6 Preservation Planning

An identified group of potential Consumers who should be able to understand a particular set of information is responsible for mapping out a preservation strategy. The Designated Community may be composed of multiple user communities.

Preservation Planning has the following responsibilities:

- Monitoring OAIS Environment
- Detect Changes Impacts in Designated Community KB
- Mapping out Preservation Strategy
- Provide recommendations to ensure information remains accessible and understandable to Designated Community

Figure 6 indicates one way in which **CASPAR** Key Preservation Components may be used within a repository, mapping to the OAIS Functional Model. The Key Components are described in detail in the next sections of this document.

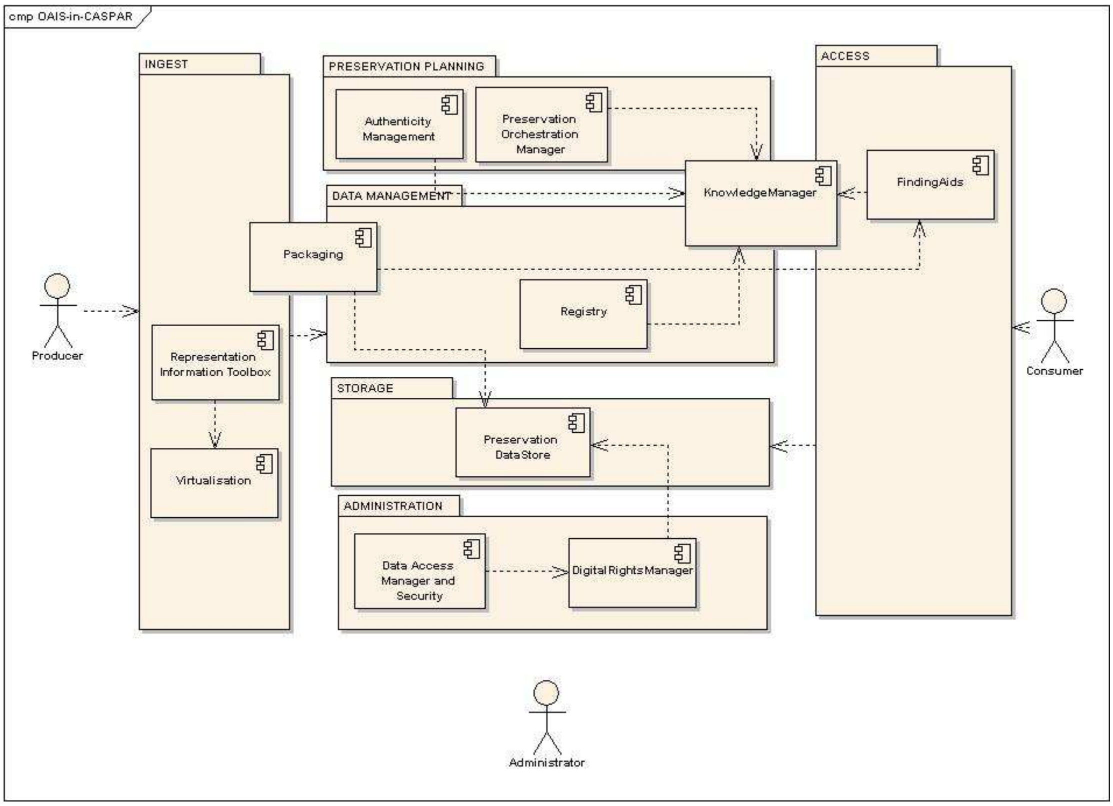


Figure 6 – Illustration of a possible mapping of CASPAR Key Components onto the OAIS Functional Model

4.2 OAIS INFORMATION MODEL

The Information Model in OAIS may be taken and the following more detailed UML class diagram may be drawn in order to show some rather general operations, such as get/put for sub-components, In other words one knows that given an Information Object one must be able to obtain (1) the Data Object as well as (2) the associated Representation Information, and thus we can add the appropriate methods to the Information Object class. Given the very general nature of an Information Object, it is very hard to have any other general methods.

The Representation Information (RepInfo) class is a special type of Information Object and we show a number of other methods which indicate a further level of analysis which is possible, and which is discussed in sections 5 and 6.

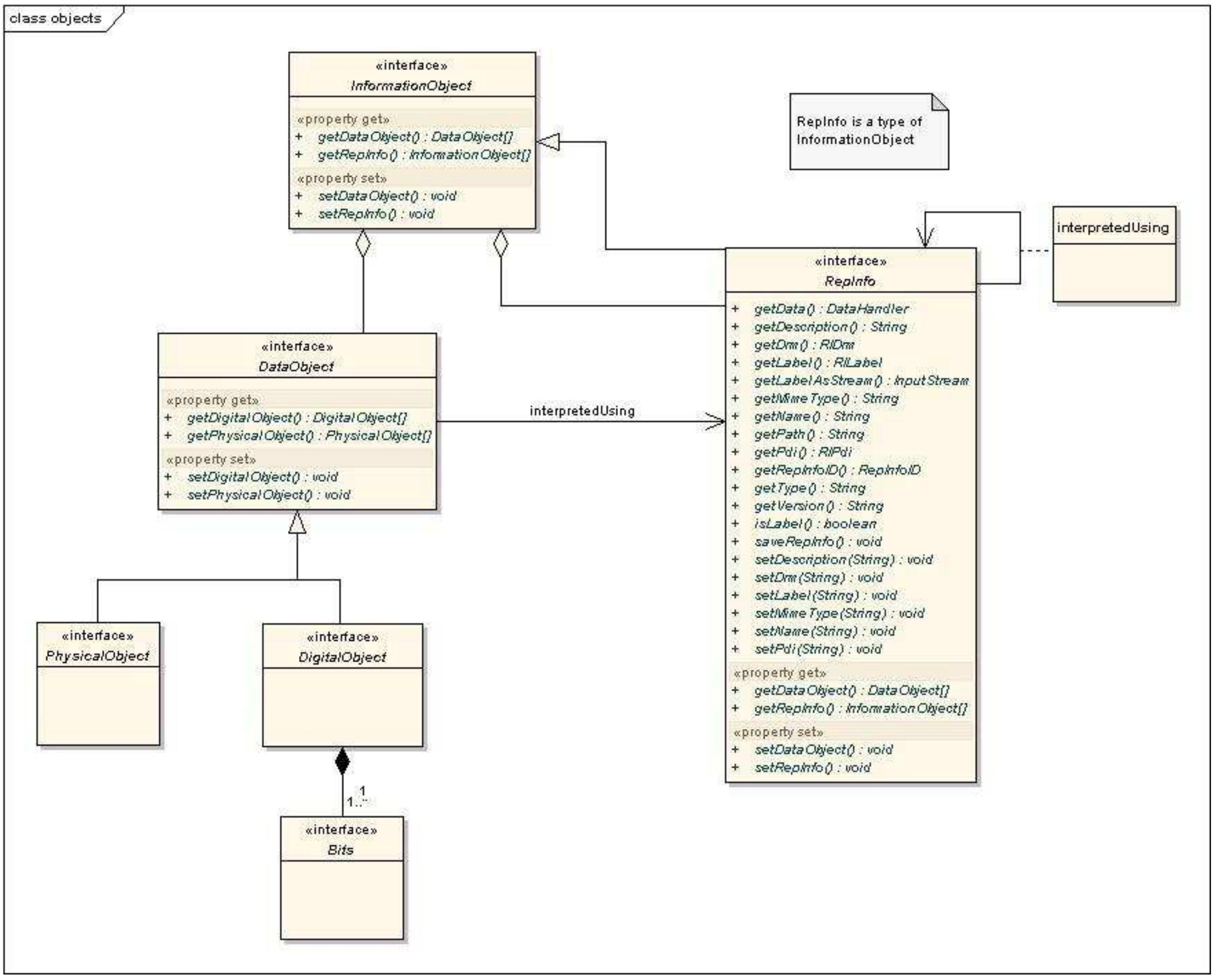
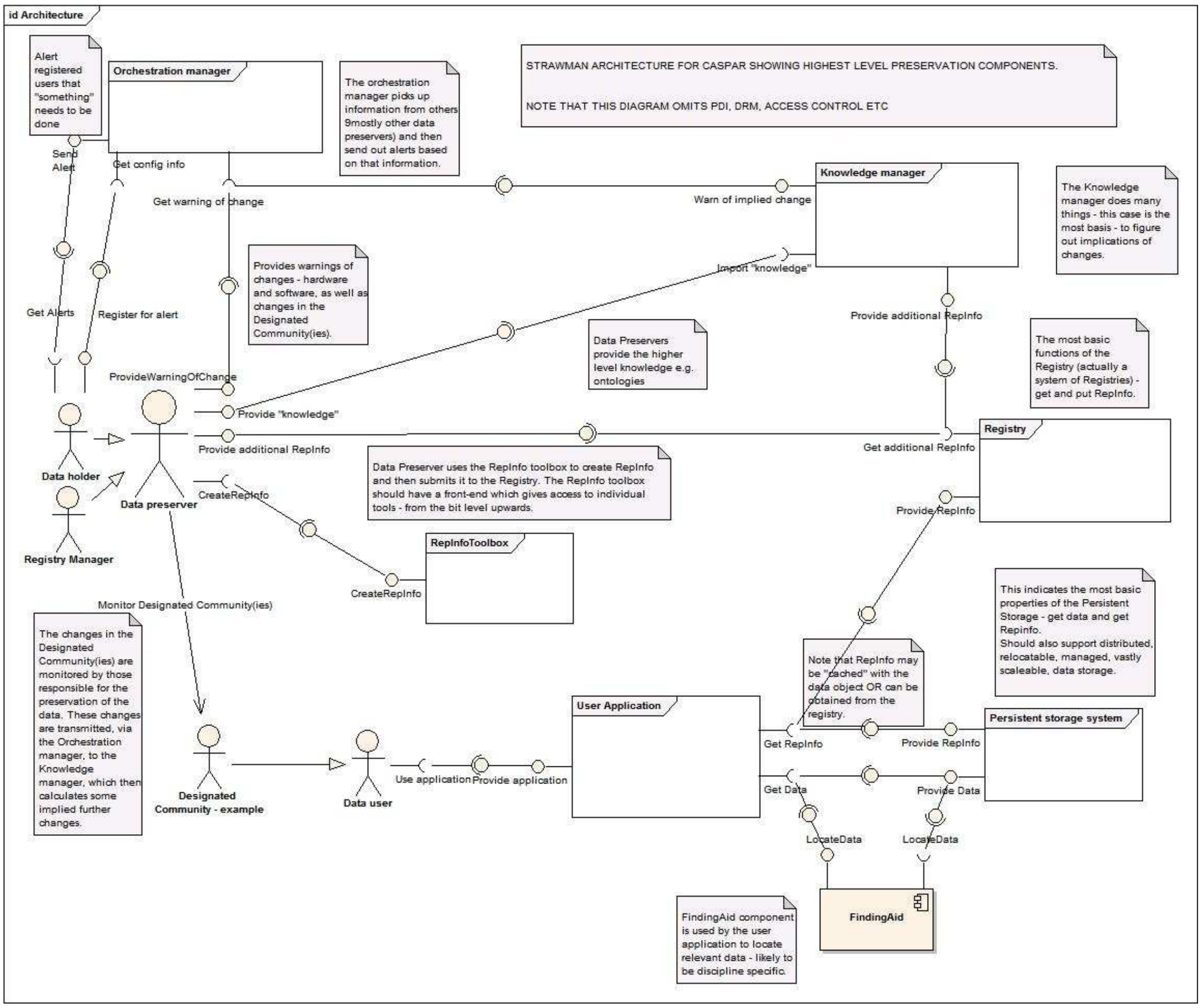


Figure 7 - Details for Information Model in OAIS: Representation Information

5 KEY PRESERVATION COMPONENTS USE CASES

The Infrastructure Components diagram, which is discussed in the Conceptual Model, is reproduced here for convenience. These components facilitate the sharing of the effort needed to preserve digitally encoded information over the long term. Note that **CASPAR** does not aim to produce a general turnkey archive system.



Alert registered users that "something" needs to be done

The orchestration manager picks up information from others 9mostly other data preservers) and then send out alerts based on that information.

STRAWMAN ARCHITECTURE FOR CASPAR SHOWING HIGHEST LEVEL PRESERVATION COMPONENTS.

NOTE THAT THIS DIAGRAM OMITS PDI, DRM, ACCESS CONTROL ETC

The Knowledge manager does many things - this case is the most basis - to figure out implications of changes.

Provides warnings of changes - hardware and software, as well as changes in the Designated Community(ies).

Data Preservers provide the higher level knowledge e.g. ontologies

The most basic functions of the Registry (actually a system of Registries) - get and put RepInfo.

Data Preserver uses the RepInfo toolbox to create RepInfo and then submits it to the Registry. The RepInfo toolbox should have a front-end which gives access to individual tools - from the bit level upwards.

The changes in the Designated Community(ies) are monitored by those responsible for the preservation of the data. These changes are transmitted, via the Orchestration manager, to the Knowledge manager, which then calculates some implied further changes.

This indicates the most basic properties of the Persistent Storage - get data and get Repinfo.
Should also support distributed, relocatable, managed, vastly scaleable, data storage.

Note that RepInfo may be "cached" with the data object OR can be obtained from the registry.

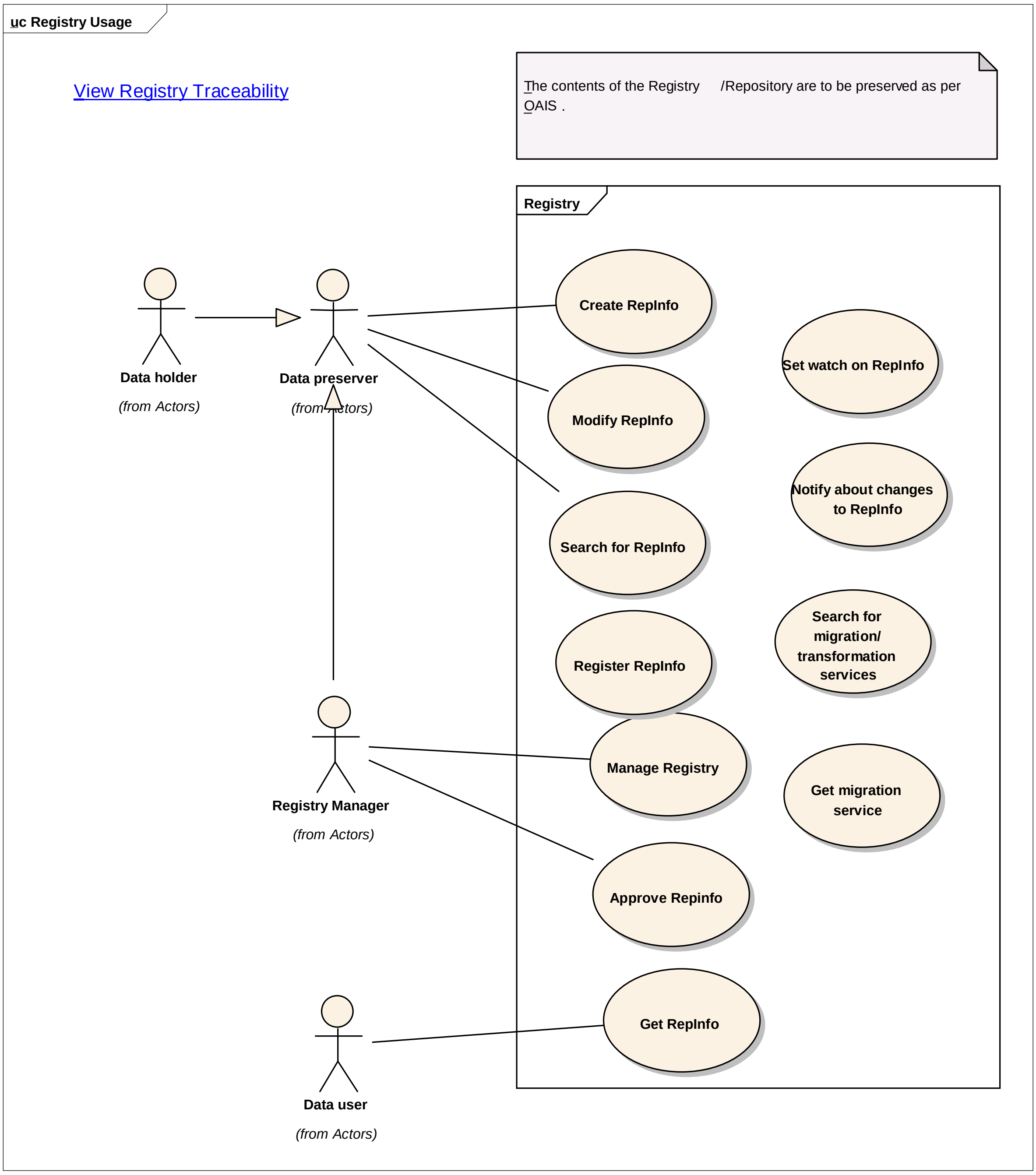
FindingAid component is used by the user application to locate relevant data - likely to be discipline specific.

Figure 8 Infrastructure Components

Use Cases for these components are described next, by adopting the following template table.

Use Case Name	Use case name - it identifies univocally the use case in the project
Actors	List of all actors that are involved in the use case
Generalisation of	Is the use case generalisation of some other use case? If yes, the name is specified.
Specialisation of	Is the use case specialisation of some other use case? If yes, the name is specified.
Contains	The use case can contain some other use cases
Preconditions	What are conditions needed that use case could be performed?
Postconditions	What conditions we get after the use case is done?
Description	Describe the use case as much as possible
Scenario	Give detailed scenario how the use case is executed
Example	Provide a sample for the described scenario
Alternative Scenario	Same for an alternative scenario, if any. Create as many as needed alternative scenario row.

5.1 REGISTRY USE CASES



5.1.1 Approve Repinfo

Use Case:	Approve Representation Information which has been registered with the Registry/Repository (RegRep) system
Pre-condition:	User has registered RepInfo and an associated, possibly incomplete, RepInfoLabel, which has been marked as "preliminary"
	Administrator has satisfied himself that the RepInfo is OK
Description:	

	Administrator authenticates himself with the RegRep system Administrator selects the RepInfo Administrator marks the RepInfo as "approved". Note that this approval can only be a recommendation. RepInfo which is correct but not approved should remain accessible.
Post-condition:	RegRep keep a log of the transactions
	Original submitter is notified
Exceptions:	Administrator cannot be authenticated
Variations:	
	Multiple RepInfo sets are approved RepInfo can only be accessed by authorised users

5.1.2 Get RepInfo

Use Case:	Given a (Persistent) Identifier pointing to some RepInfo
Pre-condition:	User has a PersistentID associated with some data, or with some other RepInfo
Description:	User enters the PersistentID in the appropriate infrastructure to send a request to the RegRep. User receives response from the RegRep which contains the RepInfo and an associated RepInfoLabel
Post-condition:	Log is kept at the RegRep
Exceptions:	PersistentID is not recognised by the RegRep system
Variations:	Multiple PersistentID are sent. RepInfo can only be accessed by authorised users

5.1.3 Modify RepInfo

Use Case:	Modify some existing RepInfo - create new version. All versions should be accessible by root (logical) Identifier.
Pre-condition:	User has prepared RepInfo update and an associated, possibly still incomplete, RepInfoLabel
Description:	User authenticates himself with the RegRep system. INCLUDES use case where user registers for the first time with the RegRep system. User submits the PersistentID and the updated RepInfo and the updated RepInfoLabel. RegRep system returns a PersistentID associated with the RepInfo
Post-condition:	RegRep keep a log of the transactions. RegRep stores the updated RepInfo as a new version with associated ownership details
Exceptions:	User cannot be authenticated. PersistentID does not exist in the RegRep system
Variations:	User is not allowed to update RepInfo. RepInfoLabel is not valid. Multiple RepInfo sets are sent. RepInfo can only be accessed by authorised users

5.1.4 Notify about changes to RepInfo

Use Case:	Notification is sent about changes in Representation Information to registered users
Pre-condition:	User has registered to be notified about changes in RepInfo
Description:	RepInfo is altered. RegRep system sends notification to all users who have registered to be notified
Post-condition:	RegRep keep a log of the transactions
Exceptions:	User cannot be contacted. User is no longer registered with the RegRep System. The associated DRM forbids notification to be sent
Variations:	Notification may be requested for changes in other types of info in the RegRep

5.1.5 Register RepInfo

Use Case:	Register Representation Information with the RegRep system
Pre-condition:	User has prepared RepInfo and an associated, possibly incomplete, RepInfoLabel
Description:	User authenticates himself with the RegRep system. INCLUDES use case where user registers for the first time with the RegRep system. User submits the RepInfo and the RepInfoLabel. User may propose an ID. RegRep system returns a PersistentID associated with the RepInfo
Post-condition:	RegRep keep a log of the transactions. RegRep stores the RepInfo with associated ownership details
Exceptions:	User cannot be authenticated. User is not allowed to deposit RepInfo. RepInfoLabel is not valid
Variations:	Multiple RepInfo sets are sent. RepInfo can only be accessed by authorised users. User specifies who is allowed to modify this RepInfo and RepInfoLabel

5.1.6 Search for RepInfo

Use Case:	Search for Representation Information in the RegRep
Pre-condition:	User is preparing RepInfo for some data (s)he holds. Before creating a particular type of RepInfo, e.g. Structure, Semantics, Software etc, (s)he searches to see whether or not some appropriate RepInfo has already been registered with the RegRep system.
Description:	User enters search criteria into the RegRep system - could involved any of the classification schemes or data models used by the RegRep. User system may first interrogate RegRep system for applicable search criteria e.g. classification schemes etc. RegRep responds with list of RegRep entries which satisfy the search criteria. User selects appropriate RepInfo item or returns to task (1) with new criteria or decides that no appropriate RepInfo exists
Post-	Log is kept at the RegRep of the search

condition:	
Exceptions:	Search criteria are not recognised by the RegRep system
Variations:	RepInfo can only be accessed by authorised users

5.2 KNOWLEDGE MANAGER USE CASES

As noted in [**D1201**] Knowledge Management techniques are used throughout the **CASPAR** system, much of it behind the scenes. The Use Cases shown here address two aspects of this

(1) the high level Use Cases to address the RepInfo Gaps which arise as the knowledge base of the Designated Community changes and

(2) some low level Use Cases which support (1) and many other applications which are not shown in detail in this document

5.2.1 High level use cases

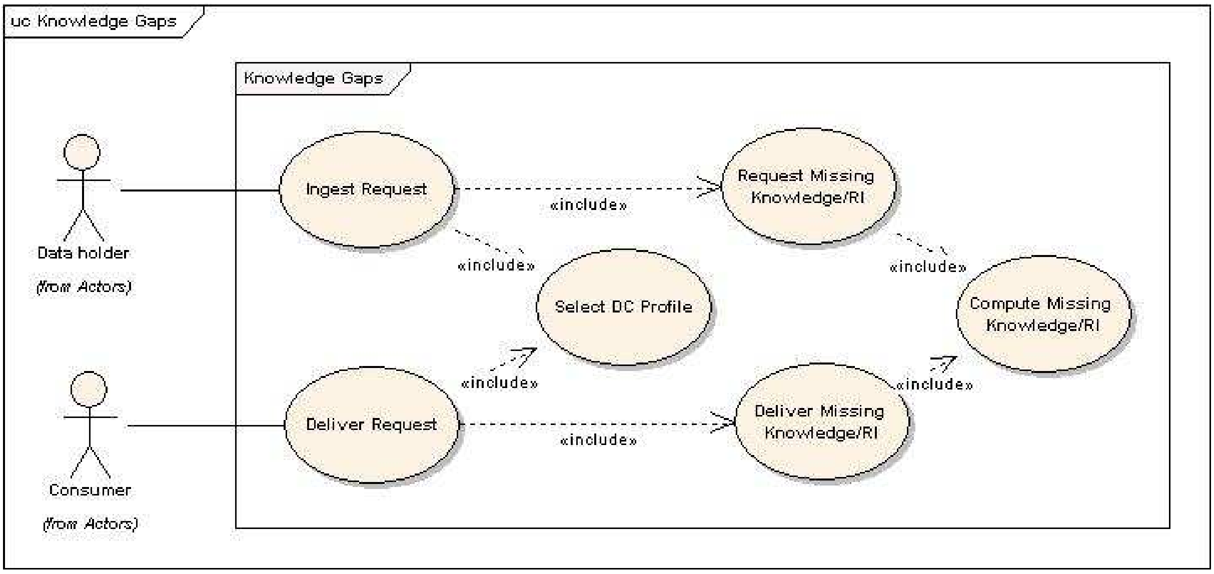
5.2.1.1 RepInfo Gaps

As the Knowledge Base of a Designated Community (hereafter we use the term DC for short) changes over time, or else the definition of a DC changes, there will arise "gaps" in the required Representation Information, for example because things which were common knowledge among the DC are no longer common knowledge. Leaving such "gaps" to develop would mean that the digitally encoded information would become unintelligible over time.

The Knowledge Base of a DC must somehow be captured. One way to do this is to adopt the notion of DCProfiles as described below. The corresponding use case diagram follows.

The notions of DCProfile and of RepInfo Gap were introduced in [**D1201**], the Conceptual Model, and a more formal definition is available at [Y.Tzitzikas'2007]. That paper can be construed as part of this deliverable.

In what follows we use "Knowledge/RI" as a general term, but in order to focus on the specific type of gaps we are addressing, the term RepInfo is used in the Use Case descriptions.



5.2.1.2 Ingest Request

Use Case	Ingest request
Actors	Data holder
Generalisation of	None
Specialisation	None

of	
Contains	Request Missing RepInfo, Select DC Profile
Preconditions	
Postconditions	
Description	1/ The actor asks the ingestion of the desired objects 2/ UC: Select DC Profile 3/ UC: Compute Missing RepInfo 4/ If the result of 3/ is not empty then UC: Request Missing RepInfo
Scenario	
Example	1) User *u* wants to ingest an object *o* 2) He selects the corresponding DC profile 3) If the RepInfo information of the object is not known by the selected DC profile, the creation of the missing RepInfo modules will be requested.
Alternative scenario	

5.2.1.3 Deliver Request

Use Case	Deliver request
Actors	Consumer
Generalisation of	none
Specialisation of	none
Contains	Select DC Profile, Deliver missing RepInfo
Preconditions	
Postconditions	
Description	The missing knowledge is delivered to the user
Scenario	1/ The actor asks the delivery of the desired objects 2/ UC: Select DC Profile 3/ UC: Compute missing knowledge 4/ if the result of 3/ is not empty then UC: Deliver Missing RepInfo
Example	1) User *u* wants to get an object *o* 2) The user select his DC profile(s) 3) The system compute the missing modules (those required for the intelligibility of *o*) and delivers them to the user.
Alternative scenario	

5.2.1.4 Compute Missing RepInfo

Use Case	Compute Missing RepInfo
Actors	
Generalisation	none

of	
Specialisation of	none
Contains	none
Preconditions	One or more DC Profiles have been selected and one or more objects have been selected
Postconditions	
Description	The system computes the missing knowledge. In brief, the missing knowledge is the difference between the whole set of RepInfo needed to understand the data object to be preserved and the RepInfo that it is assumed known by the selected profile. More details are available in a separate document.
Scenario	
Example	
Alternative scenario	

5.2.1.5 Select DC Profile

Use Case	Select DC Profile
Actors	Data holder, Consumer
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	
Postconditions	
Description	A list of all profiles will be provided and the user will select the desired ones.
Scenario	1/ The set of all recorded DC profiles are listed 2/ The actor selects the desired ones.
Example	
Alternative scenario	

5.2.1.6 Request missing RepInfo

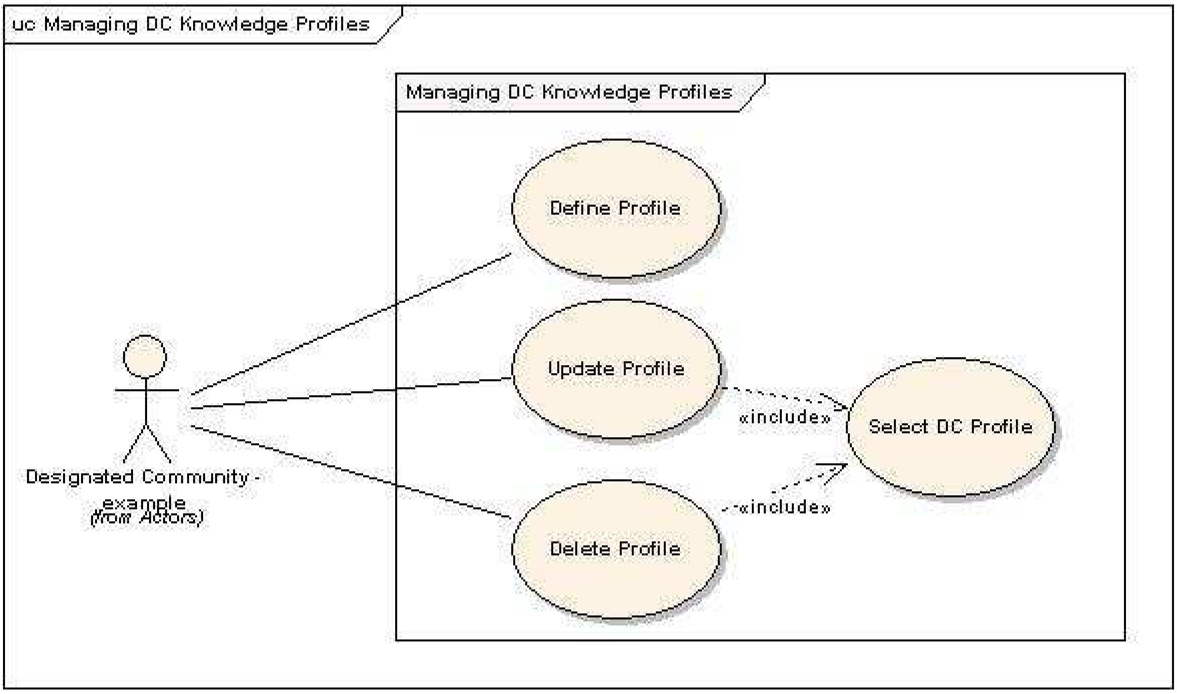
Use Case	Request missing RepInfo
Actors	
Generalisation of	None
Specialisation of	None
Contains	Compute Missing RepInfo

Preconditions	The result of UC: Compute Missing RepInfo is not empty.
Postconditions	
Description	
Scenario	The actor is asked to create the missing RepInfo by adding it to the Registry.
Example	
Alternative scenario	

5.2.1.7 Deliver missing RepInfo

Use Case	Deliver missing RepInfo
Actors	
Generalisation of	none
Specialisation of	none
Contains	Compute Missing RepInfo
Preconditions	The result of UC: Compute Missing RepInfo is not empty.
Postconditions	
Description	The actor is given the missing RepInfo.
Scenario	
Example	
Alternative scenario	

5.2.1.8 DC Profiles management


uc Managing DC Knowledge Profiles
Managing DC Knowledge Profiles
Define Profile
Update Profile
Select DC Profile
Delete Profile
«include»
«include»
Designated Community - example
(from Actors)

5.2.1.9 Define Profile

Use Case	Define profile
Actors	Designated Community (DC)
Generalisation of	none
Specialisation of	none
Contains	None
Preconditions	The actor has the access rights to define a DC profile.
Postconditions	A new profile for a specific DC will be created
Description	Each profile will be associated with a set of instances of the class Representation Information. A user can define a profile by getting a list from the registry and then selecting the desired elements.
Scenario	
Example	A new DC community appears.
Alternative scenario	

5.2.1.10 Update profile

Use Case	Update Profile
Actors	Designated community
Generalisation of	None
Specialisation of	None
Contains	Select DC Profile
Preconditions	The actor has the access rights to define a DC profile.
Postconditions	The updated DC profile is now stored.
Description	An existing profile of a DC will be updated
Scenario	The actor deletes or adds elements to the selected profile.
Example	
Alternative scenario	

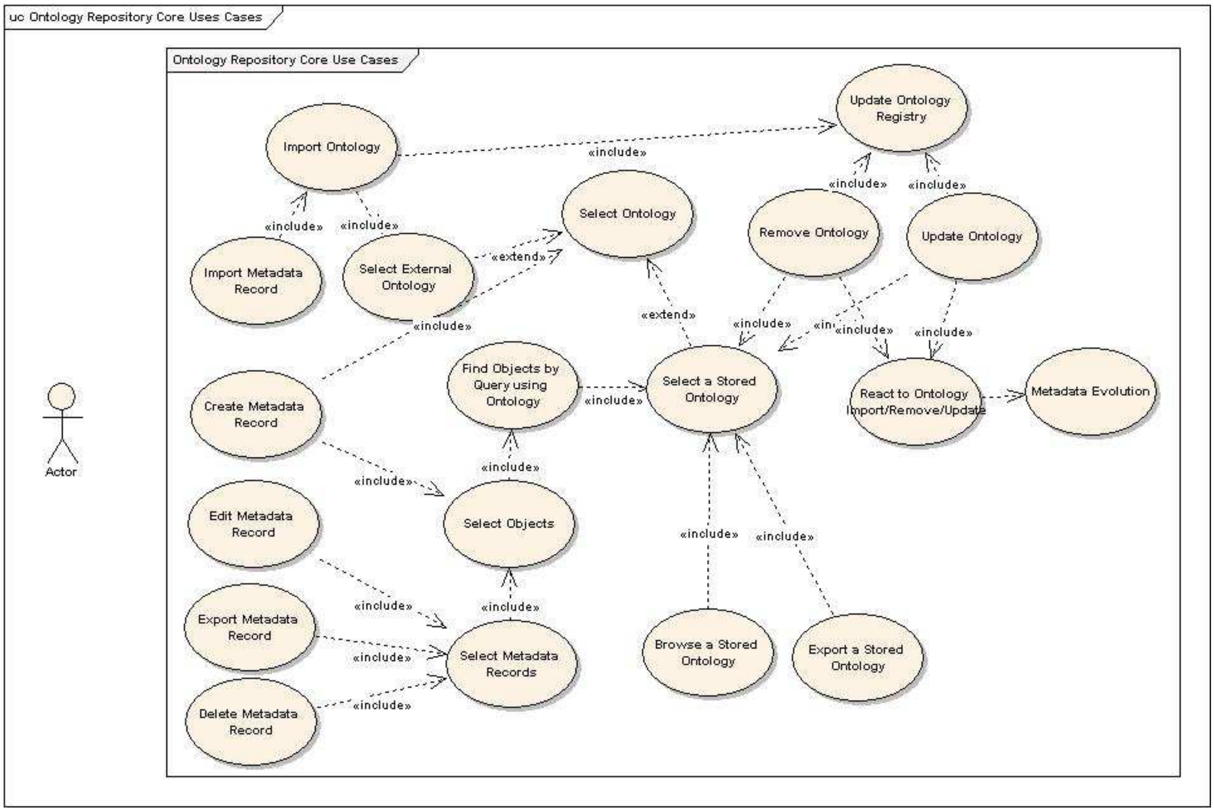
5.2.1.11 Delete profile

Use Case	Delete profile
Actors	Designated community
Generalisation of	None
Specialisation of	None
Contains	Select DC Profile
Preconditions	The actor has the access rights to delete a DC profile.

Postconditions	The selected profile is deleted.
Description	An existing profile of a DC will be deleted.
Scenario	
Example	
Alternative scenario	

5.2.2 Low level use cases

5.2.2.1 Core Ontology Management Use Cases



5.2.2.2 Browse a Stored Ontology

Use Case	Browse a stored ontology
Actors	No human actors (so far)
Generalisation of	none
Specialisation of	none
Contains	Select a stored ontology
Preconditions	
Postconditions	
Description	A set of core services for navigating the graph of an ontology should be

	provided.
Scenario	Another component could navigate through the graph of an ontology
Example	Some component is required to find some specific properties.
Alternative scenario	

5.2.2.3 Create Metadata Record

Use Case	Create Metadata Record
Actors	No human actors (so far)
Generalisation of	none
Specialisation of	none
Contains	Select ontology, Select objects
Preconditions	
Postconditions	The KM repository will be updated with the new record. A message will be generated in case that the creation of the desired metadata records could not be performed.
Description	The actor selects the object(s) and then the ontologies he wants to use.
Scenario	Several methods should be possible * Import a file in the from of RDF-XML (see also UC: Import Metadata Record) * Use a Declarative Update Language * Through a User Interface
Example	Some component wants to describe some object(s) using one or more ontologies.
Alternative scenario	

5.2.2.4 Delete Metadata Record

Use Case	Delete Metadata Record
Actors	No human actors (so far)
Generalisation of	none
Specialisation of	none
Contains	Select Metadata Records
Preconditions	The metadata record should already exist
Postconditions	The metadata record will be deleted. In case that the metadata record could not be deleted (for some reasons) the system will generate a message.
Description	A metadata record is deleted.
Scenario	1/ The Actor selects the desired metadata records (UC: Select Metadata Records)

	2/ The Actor requests their deletion. 4/ The System checks if this can be performed.
Example	A component requires the deletion of a metadata record.
Alternative scenario	

5.2.2.5 Edit Metadata Record

Use Case	Edit Metadata Record
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Select Metadata Records
Preconditions	The metadata record should already exist
Postconditions	
Description	
Scenario	Indicative steps: 1/ The Actor selects the desired metadata records (see UC: Select Metadata Records) 2/ The Actor edits them (e.g. through an UI) 3/ The System checks the validity of the requested updates. If valid the ontology repository is updated. Notes: • RUL could be used for the case where bulk metadata updates are needed • Alternatively (only if the metadata record is not connected to any other records) one could first ask its deletion and then create a new one
Example	
Alternative scenario	

5.2.2.6 Export Metadata Record

Use Case	Export Metadata Record
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Select Metadata Records
Preconditions	
Postconditions	

Description	
Scenario	1/ The Actor selects the desired metadata records (see UC: Select Metadata Records) 2/ The selected metadata are exported in the desired format (e.g. RDF/XML, TRIGG, ...)
Example	Export a DIP.
Alternative scenario	

5.2.2.7 Export a Stored Ontology

Use Case	Export a Stored Ontology
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Select a Stored Ontology
Preconditions	
Postconditions	
Description	
Scenario	1/ The Actor selects the desired Ontology 2/ The Actor selects the desired output format 3/ The System returns the selected Ontology in the selected format.
Example	
Alternative scenario	

5.2.2.8 Find Objects by Query using Ontology

Use Case	Find Objects by Query using Ontology
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Select a Stored Ontology
Preconditions	
Postconditions	
Description	
Scenario	The user selects an ontology (from the stored ones) and then (s)he formulates a query in order to find the desired objects. Some ontologies may be associated with some pre-cooked queries (where the user would have to provide only some parameters). In addition, the user may want to store a query that (s)he has formulated.

Example	
Alternative scenario	

5.2.2.9 Import Metadata Record

Use Case	Import Metadata Record
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Import Ontology
Preconditions	
Postconditions	
Description	The metadata record could be an RDF/XML or TRIG stream. If the metadata record uses an ontology that is not stored in the repository, then it should be fetched (to allow validation) and then stored to the repository.
Scenario	
Example	A SIP expressed in RDF/XML format.
Alternative scenario	

5.2.2.10 Import Ontology

Use Case	Import Ontology
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Update Ontology Register, Select External Ontology
Preconditions	
Postconditions	
Description	Imports an external ontology to the repository. This is important for validating the metadata that instantiate the ontology. In addition, even ontologies should be preserved and curated. The system should fetch, parse and check the syntactic/semantic validity of the ontology before importing it to the repository.
Scenario	
Example	Import the CIDOC CRM. Import an ontology allowing the description of scientific workflows.
Alternative	

scenario	

5.2.2.11 Metadata Evolution

Use Case	Metadata Evolution
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	
Preconditions	
Postconditions	
Description	
Scenario	1/ The Actor provides as input the two ontologies , say O and O' (may both stored at the repository). 2. If O' is backwards compatible with O then no further action is needed. If not, then a metadata migration program is needed. 3. If there is such a program (e.g. provided by the authority responsible for O) then it should be loaded and enacted. 4. If not, then the user should be able to define a migration plan (by formulating mapping rules). A availability of a comparison function that computes the delta between these ontologies might help. 5. The migration plan is enacted. The user can say whether he wants to keep stored of the old version of the metadata (in case the migration/transformation was lossy).
Example	Metadata Migration (from one ontology to another). i.e. metadata with respect to O should be expressed with respect to O'.
Alternative scenario	

5.2.2.12 React to Ontology Import/Remove/Update

Use Case	React to Ontology Import/Remove/Update
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	
Preconditions	
Postconditions	
Description	The goal is to preserve the information after a change. Suppose a new version of an ontology emerges. This may correspond to a new version of a metadata schema, a new version of a scientific thesaurus (capturing DC's knowledge), or a new Knowledge Base (ontology + instances) modelling some pieces of current real world knowledge. The Knowledge Repository

	should be updated to ensure that its contents are updated and preserved
Scenario	
Example	
Alternative scenario	

5.2.2.13 Remove Ontology

Use Case	Remove Ontology
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Update Ontology Register, Select External Ontology, React to Ontology Import/Remove/Update
Preconditions	
Postconditions	
Description	If the ontology in instantiated or if there are other ontologies (stored in the repository) that reuse/extend some parts of this ontology, then the deletion will not be allowed. In case of deletion, ontology registry gets updated.
Scenario	
Example	
Alternative scenario	

5.2.2.14 Select External Ontology

Use Case	Select External Ontology
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	Select Ontology
Contains	
Preconditions	
Postconditions	
Description	
Scenario	The Actor provides the URI of the external ontology (and may its representation in XML/RDF or in any other format).
Example	
Alternative scenario	

5.2.2.15 Select Metadata Records

Use Case	Select Metadata Records
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Select Objects
Preconditions	
Postconditions	
Description	Involves selecting objects and ontologies
Scenario	1/ The user selects a set of objects (see UC: Select Objects) 2/ The user selects some (or all) of the ontologies that the metadata of the selected objects use 3/ The corresponding metadata are selected.
Example	
Alternative scenario	

5.2.2.16 Select Objects

Use Case	Select Objects
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	Find Objects By Query Using Ontology
Preconditions	
Postconditions	
Description	The selection of object may involve querying the ontology repository.
Scenario	
Example	
Alternative scenario	

5.2.2.17 Select Ontology

Use Case	Select Ontology
Actors	No human actors (so far)
Generalisation of	Select a Stored Ontology, Select External Ontology
Specialisation of	None
Contains	None

Preconditions	
Postconditions	
Description	For more details see “Select a Stored Ontology” and “Select External Ontology”
Scenario	
Example	
Alternative scenario	

5.2.2.18 Select a Stored Ontology

Use Case	Select a Stored Ontology
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	Select Ontology
Contains	None
Preconditions	
Postconditions	
Description	This Use Case should allow an actor to select one (or more than one) ontology that is stored in the repository. The system could just return a list with all stored ontologies. Alternatively or additionally, the actor could use the Registry Ontology in order to find the desired ontology (e.g. by browsing it or by formulating a query).
Scenario	
Example	
Alternative scenario	

5.2.2.19 Update Ontology Registry

Use Case	Update Ontology Registry
Actors	No human actors (so far)
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	
Postconditions	
Description	The ontology repository will have a registry. Whenever an ontology is imported, deleted, or updated the registry should be updated.

	A special Registry Ontology could be used for describing the ontologies of the repository. It may include information about versioning.
Scenario	
Example	
Alternative scenario	

5.3 PRESERVATION ORCHESTRATION MANAGER USE CASES

The Preservation Orchestration Manager will provide a number of notification/alert services including:

- alert repositories, which have registered appropriately, of the probable need to take action to ensure the preservation of their holdings. This action could range from the need for migration to new formats to the obsolescence of hardware to the availability of relevant Representation Information.
- alert repositories that certain collections of digital objects are in need of a new host to ensure their long term preservation
- alert registered experts who have the appropriate expertise that additional Representation Information needs to be created or pointed to in order to ensue the continuing understandability of certain digitally encoded information

The alerts themselves are generated from information which have been submitted by registered users (i.e. people), and also a Gap Manager.

Not indicated explicitly here is the use of a user registration service, the use cases for which are provided in section 5.6.

This infrastructure must itself be persistent.

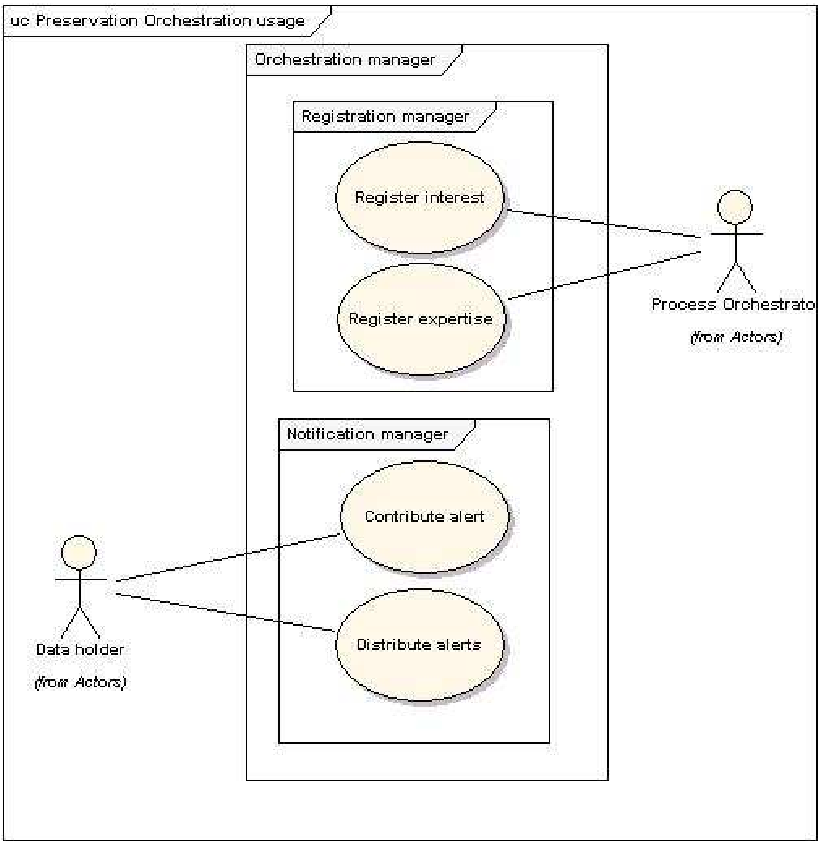


Figure 9 Orchestration manager use cases

5.3.1 Register Interest

Use Case	Register interest
Actors	Data preserver
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	
Postconditions	Actor is registered with the Orchestration manager, with known interests and area of expertise.
Description	A person registers with the Orchestration manager, informing it of his/her areas of interest, data holdings, hardware in use
Scenario	The person registers, providing • Contact details • Identification/ authentication details • Areas of interest according to one or more classification schemes
Example	
Alternative scenario	

5.3.2 Register expertise

Use Case Name	Register expertise
Actors	Person
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	
Postconditions	Person's areas of expertise are registered with the Orchestration manager
Description	A person registers, or if already registered then logs-in, with the Orchestration manager. He/she registers his/her areas of expertise, classified under one or more classification schemes, and indicates his/her willingness to be contacted to generate Representation Information.
Scenario	
Example	
Alternative scenario	

5.3.3 Contribute alert

Use Case Name	Contribute Alert
Actors	Data Holder
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	
Postconditions	The Orchestration Manager hold information about the alert and will attempt to match this to registered experts.
Description	The actor registers an alert according to one or more of the classification schemes which are available.
Scenario	The actor becomes aware of some change in one or more areas, for example • Hardware • Software • Environment e.g. some web site which is a common reference • Knowledge base of a Designated Community The actor identifies him/herself with the Orchestration Manager and then registers these changes.
Example	A person realises that a certain type of holographic storage will soon no longer be commercially supported.
Alternative scenario	

5.3.4 Distribute alert

Use Case Name	Distribute Alert
Actors	Data Holder
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	An alert has been created, based on some warning from a registered user
Postconditions	The alert has been distributed to appropriate people
Description	An alert is distributed to registered experts
Scenario	An alert has been contributed to the Orchestration manager and, if necessary, approved. The alert classification is compared to the areas of interest or expertise of registered users. The persons contacted are requested to take appropriate action, for example:

	• Migrate holdings to new media or formats • Create new Representation Information • Offer to take over the preservation of a particular data collection
Example	
Alternative scenario	

5.3.5 Respond to alert

Use Case Name	Respond to alert
Actors	Data Holder
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	An alert has been distributed to number of registered users
Postconditions	The response to the alert is distributed appropriately
Description	One or more of people respond to an alert with a resolution of the issue
Scenario	An alert has been distributed. One or more people respond with a resolution of the issue. This resolution may come in many forms.
Example	
Alternative scenario	

5.4 REPRESENTATION INFORMATION TOOLS USE CASES

The Representation Information Toolbox guides users into a collection of individual tools, each of which creates a specific type of Representation Information. These latter tools will consist of publicly available tools as well as ones created by the **CASPAR** project.

Use Case Name	Create Representation Information
Actors	Data Preserver
Generalisation of	N/A
Specialisation of	N/A
Contains	N/A
Preconditions	Data preserver has the ability to create Representation Information about the data object.
Postconditions	New Representation Information is created and will be made available. This may be of the form of a number of description languages, or it may be that, at this particular time, only a straightforward text document will fulfil the need.
Description	New Representation Information is to be created, and the tool provides assistance to the actor. The Representation Information tool is an portal from which any number of other tools may be accessed.
Scenario	The Data Preserver knows, or has been alerted, that some additional Representation Information is needed. There are many types of RepInfo e.g. Structure, Semantics, Software, etc and for each of these there are many possible ways of describing it. The RepInfo tools provide a number of hints and suggestions to the Data Preserver, and acts as a portal to a number of additional tools.
Example	FITS (a data format used in Astronomy) is becoming obsolete. The Data Preserver is informed of this and creates appropriate Representation Information. The tool suggests a number of different suggestions, for example: • what standards are involved and where can they be found? • Are simple objects such as “image” or “table” contained in a FITS file? • There are a number of tools for defining structure: o DRB is useful for describing data which is …. o EAST is useful for describing data which is ….. o DFDL is useful for …. o Etc

	Based on these suggestions the Data Preserver creates one or more of: Structural RepInfo: • collecting the documents which define the FITS format • a description of FITS using DRB Semantic Information • the semantics of the keywords in the header which define the World Co-ordinate systems • a data dictionary for the standard FITS keywords • a data dictionary for the keywords used by a number of major observatories Software • collect JAVA software for reading FITS files

5.5 PRESEVATION DATASTORE USE CASES

Today, more and more storage systems offload advanced functionality and structure-awareness to the storage layer. Functions that were traditionally carried out by the application or the operating system are gradually becoming integral parts of an 'intelligent storage system'. Object-store devices (OSDs), for example, offload space allocation and security to the storage device. Functions such as bit-to-bit data migration, block-level data integrity, and even encryption, are carried out by advanced, intelligent disks and tapes. Some systems (e.g. provenance-aware storage system (PASS)) already track the provenance of data at the storage level rather than storing it in a standalone database.

At the heart of any solution to the preservation problem lies a **storage** component. This is the portion of the system that manages the long-term storage and maintenance of digital material entrusted to the OAIS. Digital preservation systems will be more robust and have less probability for data corruption or loss if they offload preservation related functionality to the storage layer. We call such a storage component with built-in support for preservation by the term **preservation aware storage**. OAIS-based storage is a specific type of preservation aware storage, which is based on OAIS notions, functions and information model.

The major requirements of an **OAIS-based preservation aware storage** are:

- Encapsulate and physically co-locate in the storage, where appropriate, the raw data and its complex interrelated metadata objects, such as representation information, provenance, and fixity. This ensures that at least the unique Representation Information needed for interpretation is not separated from the raw data and thus never lost (if the raw data survives), while pointing to the rest of the Representation Information network.
- Utilise the locality property and execute data intensive functions such as fixity computations within the storage component.
- Include, or point to, the representation information of metadata such as that of fixity and provenance, so that they can be validated and interpreted when migrating to newer systems.
- Handle technical provenance events internally. The applications on top of the preservation aware storage should be freed from managing events that can be handled internally in the storage. Moreover, the types of provenance events are richer and also include events related to migration and transformation.
- Support the loading and execution of external transformations during the migration process. Additionally, it should facilitate on demand triggering of those transformations.
- Support media migration, as opposed to system migration, in which migration from one system to another can be done by physically detaching the media from one system and attaching it to the new system.
- Maintain referential integrity including updating all the links during the migration process such that they remain valid in the new system. This requires an awareness of certain meta-data fields that represent links, both internally to the system and externally.

- Ensure that data collections can be handed over to other systems in the chain of preservation, including supporting information such as the virtualised name spaces on which access depends
- Be able to support, and describe when requested, the preservation policies which the repository management/administration imposes on particular data holdings, including the
    - the number and frequency of backups
    - post or pre-processing demanded of the data
    - access controls
- Ensure that if, in the unlikely event of data loss, that there is a graceful loss of data. If some data is lost, a good preservation system must minimize the effect of this data loss and prevent cases where data in the system that is still intact cannot be read or interpreted.

Preservation DataStore is an OAIS-based preservation aware storage that concentrates mainly on assisting logical preservation. It supports the requirements described above, and is adequate for heterogeneous data, heterogeneous applications and heterogeneous storage media. Preservation DataStore serves as the archival storage entity described in the OAIS Functional Model.

The Preservation DataStores (PDS) include the following use cases:

1. Ingest AIP
2. Access AIP
3. Migrate AIP
4. Transform AIP
5. Add Transformer
6. Set preservation policies
7. Get preservation policies

5.5.1 Ingest AIP

Use Case Name	Ingest AIP
Actors	Storage Manager Ingest entity, PDS, Representation Information (RepInfo) Registry
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	Valid AIP
Postconditions	The AIP is ingested in PDS and the AIP unique identification (AIP ID), which can for example be used within a Curation Persistent Identifier (CPID), is returned to the Ingest Entity. The latter merges the AIP ID into the package description of the collection description prior to the addition of it to the Data Management entity.
Description	Persists the AIP for long term preservation while performing additional

	functionality as fixity computation, RepInfo extraction from RepInfo Registry, provenance updates, and referential integrity validation.
Scenario	(1) The Ingest entity actor sends to the PDS actor a “storage request” with an AIP (2) The PDS actor processes the AIP as described in section 5.1. In this processing stage, it may retrieve some RepInfo packages from the RepInfo Registry actor. (3) The PDS actor sends to the Ingest entity actor a “storage confirmation” indicating (or verifying) the storage identification information for the AIP
Example	Ingest a satellite product packaged in SAFE format
Alternative scenario	

5.5.2 Access AIP

Use Case Name	Access AIP
Actors	Storage Manager PDS, Access entity
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	Valid AIP unique identification (AIP ID)
Postconditions	The required AIP is given to the Access entity. The latter transforms the set of AIPs and associated Package Descriptions into a set of DIPs.
Description	Retrieves an AIP stored within the OAIS according to the given AIP ID.
Scenario	(1) The Access entity actor sends to the PDS actor an “AIP request” with an AIP ID (2) The PDS actor processes the request as described in section 5.2. (3) The PDS actor sends to the Access entity actor a “notice of data transfer”.
Example	Access a satellite product packaged in SAFE format that has the given AIP ID.
Alternative scenario	

5.5.3 Migrate AIP

Use Case Name	Migrate AIP
Actors	Storage Manager Access entity in old system, PDS in old system, Ingest entity in new system, PDS in new system
Generalisation of	None

Specialisation of	None
Contains	The use case contains the Ingest AIP, Access AIP, and optionally transform AIP use cases.
Preconditions	Valid AIP unique identification (AIP ID)
Postconditions	The AIP is migrated. Then, the AIP ID is merged by the Ingest entity of the new system into the package description of the collection description prior to the addition of it to the Data Management entity of the new system
Description	Migrates an AIP with the given AIP ID from an old system to a new system. In the migration a transformation may optionally occur, resulting in a new AIP (same logical ID but with new version id).
Scenario	(1) Perform the Access AIP use case in the old system (2) Optionally, perform the Transform AIP use case and then a new AIP is generated. (3) Perform the Ingest AIP use case in the new system
Example	Migrate a satellite product packaged in SAFE format to a new preservation system.
Alternative scenario	

5.5.4 Transform AIP

Use Case Name	Transform AIP
Actors	Storage Manager Preservation Planning, PDS
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	Valid AIP and valid transformer
Postconditions	A new AIP is generated and ingested into the system
Description	Transforms an AIP with the given AIP ID to a new AIP with a new AIP ID (same logical ID but with new version id).
Scenario	(1) The Preservation Planning entity actor sends to the PDS actor a transformation request for a given AIP (2) The PDS actor transforms the AIP and generates and new AIP with a new AIP ID (3) The PDS actor sends a “transformation confirmation” with the AIP ID of the new AIP to the Preservation Planning entity actor.
Example	The preservation planning requests to transform a GOME product from EGOC format to SAFE format.
Alternative scenario	

5.5.5 Add Transformer

Use Case Name	Add Transformer
Actors	Storage Manager

	Preservation Planning, PDS
Generalisation of	none
Specialisation of	none
Contains	none
Preconditions	Valid transformer
Postconditions	A new transformer is defined in the system and can be published to Preservation Planning
Description	Add a transformer to the system that can be used later on to employ transformations on AIPs
Scenario	(1) The Preservation Planning entity actor sends to the PDS actor an “add transformer” request with the given transformer (2) The PDS actor adds the transformer to the system (3) The PDS actor sends an “add transformer confirmation” with the transformer ID to the Preservation Planning entity actor.
Example	Add a GOME EGOC -to-SAFE transformer to PDS
Alternative scenario	

5.5.6 Set preservation policies

Use Case Name	Set the preservation policies which should be applied to one or more data objects
Actors	Storage Manager Ingest entity, PDS, Representation Information (RepInfo) Registry
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	
Postconditions	The preservation policies are followed by the PDS, and appropriate logs are kept of the resulting actions.
Description	Preservation policies cover many aspects including for example: • Number of backup copies, off-site and on-site, on-line and near-line, and replication • Access controls • Distribution of the information among the individual pieces of virtualised storage • Maintenance of collection level information
Scenario	The preservation policies must be set on a particular data collection, for example 12 off-site copies must be kept, and as new members of the collection are added these copies must be made of that new member.
Example	
Alternative scenario	

5.5.7 Request information about preservation policies

Use Case Name	Request information about preservation policies
Actors	Storage Manager Ingest entity, PDS, Representation Information (RepInfo) Registry
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	Preservation policies have been set
Postconditions	
Description	The preservation policies which apply to a particular data set or data object are supplied
Scenario	The preservation policies which apply to a particular data object or data collection is requested. The policies are returned in some way, for example in some formal language or a simple text description, and the appropriate piece of Representation Information is also provided.
Example	During the digital repository certification process, the preservation policies must be provided, and perhaps logs are examined in order to ensure the policies are being followed.
Alternative scenario	

5.6 DATA ACCESS MANAGER AND SECURITY (DAMS) USE CASES

5.6.1 User Management

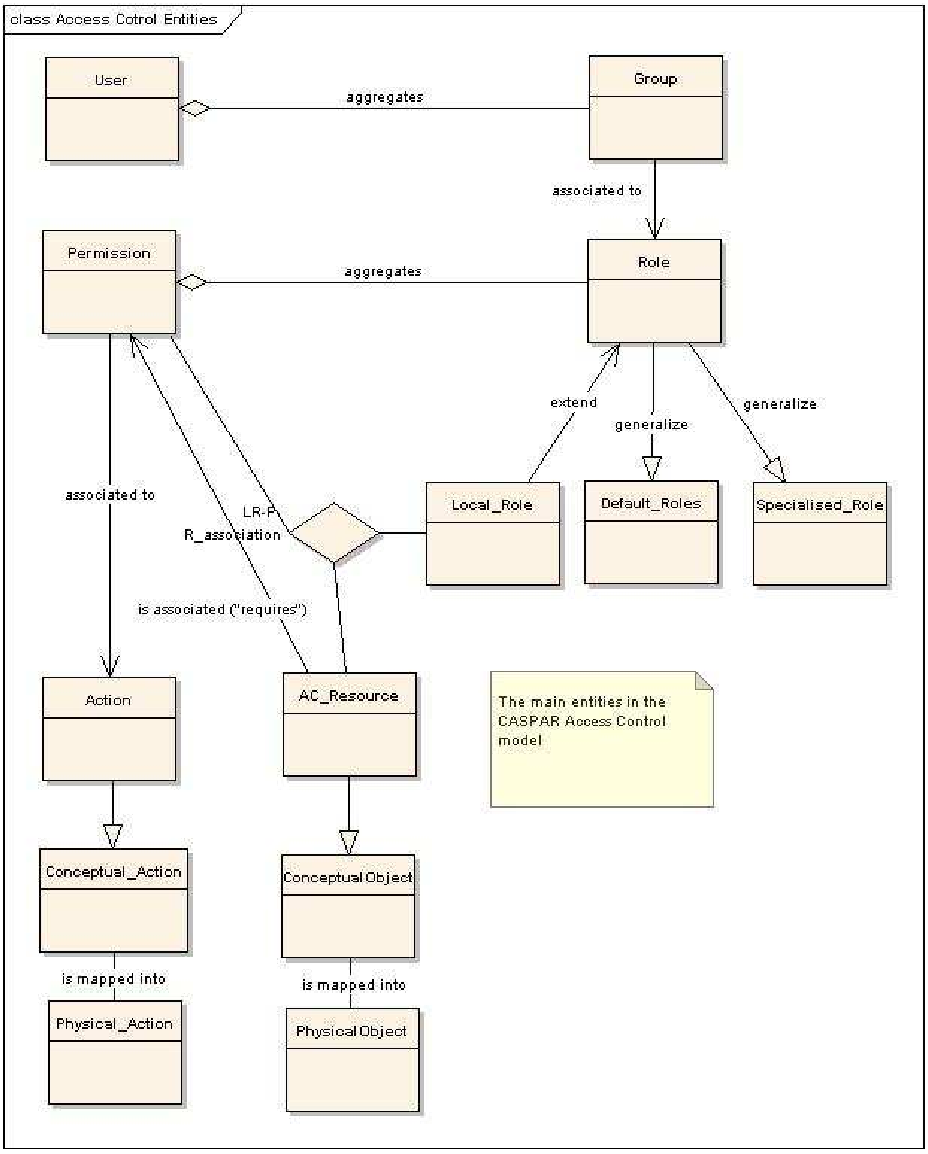
The following diagram depicts the classes that build up the Access Control model.

In short, users are aggregated into groups, permissions are aggregated into roles and roles are assigned to groups. This allows the separation of the responsibilities of user management and permissions management, and facilitates the loose coupling that makes the sustainability of the system over time a practical proposition.

Moreover, groups are not Designated Communities, as the former serve finally to associate permissions to users, while the latter serve to define the knowledge base profile of users, and more specifically, to Consumer users.

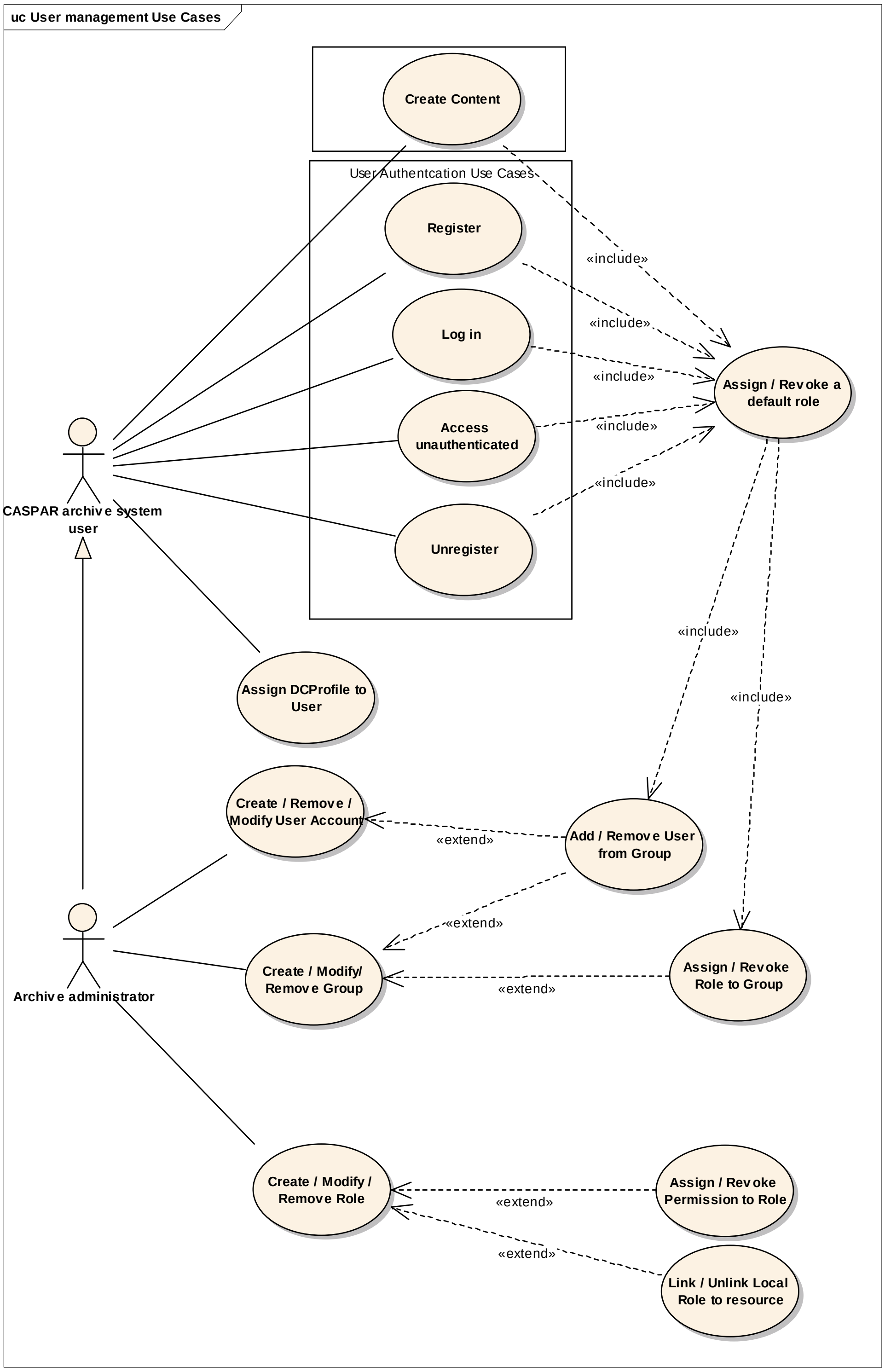
The following types of roles are foreseen:

1. Default roles: i.e.: anonymous users, registered users, managers and owners
2. User roles: To specialise authorization for registered users
3. Local Roles: To personalise permissions on specific resources and to specific users



The following diagram gives an overview of the main DAMS use cases detailed afterwards.

Note: ***The DAMS use cases do not differ from the traditional Access Control scenarios. The implementation will however address the challenges of a preservation environment, taking the long-term impacts into account.*** In particular, the enforcement of Access Control must comply with the policies defined a long time before.


uc User management Use Cases
Create Content
User Authentcation Use Cases
Register
Log in
Access unauthenticated
Unregister
«include»
«include»
«include»
«include»
«include»
Assign / Revoke a default role
CASPAR archive system user
Assign DCProfile to User
«include»
«include»
Create / Remove / Modify User Account
«extend»
Add / Remove User from Group
«extend»
Archive administrator
Create / Modify/ Remove Group
«extend»
Assign / Revoke Role to Group
Create / Modify / Remove Role
«extend»
Assign / Revoke Permission to Role
«extend»
Link / Unlink Local Role to resource

5.6.1.1 Create/Modify/Remove UserAccount

Use Case Name	[Create/Modify/Remove]UserAccount
Actors	archive system administrator; creation is started by anonymous user
Generalisation of	
Specialisation of	

Contains	[Add/Remove]User[To/From]Group AssignDCProfileToUser
Preconditions	
Postconditions	
Description	This use case serves to change the account status, and the user will experience that change
Scenario	
Example	1. The user desires to change his DCProfile 2. He accesses the interface which allows him to select a set of options to define his knowledge base profile in a guided way 3. If the successful completion is confirmed, the user will have a personalised treatment, i.e. the content will be proposed on the basis of his DCprofile
Alternative scenario	

5.6.1.2 Create/Modify/Remove Group

Use Case Name	[Create/Modify/Remove]Group
Actors	a user with the role of users administrator
Generalisation of	
Specialisation of	
Contains	[Add/Remove]User[To/From]Group
Preconditions	
Postconditions	
Description	Changes the status of a group; a group serves to define a bucket to insert users with the same permissions
Scenario	
Example	The user administrator asks the permissions administrator to define a role with some particular set of permissions, in order to associate that role to a set of users. Then he creates the group using the interface, assigns the role to the group, and adds the users to the group.
Alternative scenario	

5.6.1.3 Create/Modify/Remove Role

Use Case Name	[Create/Modify/Remove]Role
Actors	a user with the role of permissions administrator
Generalisation of	
Specialisation of	
Contains	[Assign/Revoke]Permission[To/From]Role

	[Link/Unlink]LocalRole[To/From]Resource
Preconditions	
Postconditions	
Description	Changes the status of a role; a role serves to define a profile of permissions, which can be used for user administration.
Scenario	
Example	The role administrator is asked to define a particular role. He interacts with the user administrator to identify the permission requirements, and modifies the role accordingly.
Alternative scenario	

5.6.1.4 Add/Remove User To/From Group

Use Case Name	[Add/Remove]User[To/From]Group
Actors	a user with the role of users administrator
Generalisation of	
Specialisation of	
Contains	
Preconditions	
Postconditions	The user inherits the permissions from the roles associated to the group The user must belong to (exactly) one group
Description	The user membership to groups is changed
Scenario	
Example	At registration, the user is assigned to the ‘default registered user’ group
Alternative scenario	

5.6.1.5 Assign/Revoke Permission To/From Role

Use Case Name	[Assign/Revoke]Permission[To/From]Role
Actors	a user with the role of permissions administrator
Generalisation of	
Specialisation of	
Contains	
Preconditions	
Postconditions	All authorization performed afterwards reflect the change in the user permissions
Description	Changes the definition of the role
Scenario	1. The permissions administrator is asked to change a role 2. He uses the interface to perform the operation and commits
Example	

Alternative scenario	

5.6.1.6 Assign/Revoke Role To/From Group

Use Case Name	[Assign/Revoke]Role[To/From]Group
Actors	a user with the role of permissions administrator
Generalisation of	
Specialisation of	
Contains	
Preconditions	
Postconditions	All authorization performed afterwards reflect the change in the users'(belonging to the group) permissions
Description	Changes the status of the group
Scenario	1. The users administrator is asked to change a group 2. He uses the interface to perform the operation and commits
Example	
Alternative scenario	

5.6.1.7 Assign/Revoke DefaultRole

Use Case Name	[Assign/Revoke] DefaultRole
Actors	User authentication subcomponent
Generalisation of	
Specialisation of	
Contains	[Add/Remove]User[To/From]Group, [Add/Remove]Role[To/From]Group
Preconditions	
Postconditions	
Description	This change can affect a user account, as it modifies its status, or can just affect the behaviour of the system from the perspective of the user. Default roles are: 1. anonymous users 2. owner (who creates and owns a content item) 3. registered users 4. manager
Scenario	
Example	1. The authentication component completes the logout and assigns the 'anonymous user' role 2. The creation of a new archived item implies that the 'creator' obtains the 'owner' role.

Alternative scenario	

5.6.1.8 Link/Unlink LocalRole To/From Resource

Use Case Name	[Link/Unlink]LocalRole[To/From]Resource
Actors	a user with the role of permissions administrator
Generalisation of	
Specialisation of	
Contains	
Preconditions	
Postconditions	The local role overrides the other roles (excl. owner and manager role)
Description	When a role is linked to a resource or a set of resources, the role becomes a 'local role'; it is useful to allow a fine-grained , personalized access control
Scenario	
Example	A (set of) resource(s) should be viewable/readable/writable only to a particular community of users.
Alternative scenario	

5.6.1.9 AssignDCProfileToUser

Use Case Name	AssignDCProfileToUser
Actors	DCProfileManager, User
Generalisation of	
Specialisation of	
Contains	
Preconditions	The right DC profile to associate to the user is already defined.
Postconditions	A user is assigned to a particular Designated Community
Description	It associates the user with a DCProfile
Scenario	1. At registration, or at some later time, the user is asked to complete his own personalization information, i.e. by means of a DCProfile 2. The user is guided in this process, e.g. by choosing among a set of available DCProfiles 3. If there is no appropriate DCProfile, he may define a new profile proposal, e.g. by selecting the options in a from 4. The proposal needs to be accepted by a proper profile administrator
Example	
Alternative scenario	

5.6.1.10 getUserDCProfile

Use Case Name	getUserDCProfile
Actors	Any component that needs to retrieve the DCProfile of a user
Generalisation of	
Specialisation of	
Contains	
Preconditions	
Postconditions	
Description	It serves to retrieve the DCProfile class of the reference to it
Scenario	
Example	
Alternative scenario	

5.6.2 Authentication Manager

5.6.2.1 RegisterUser

Use Case Name	RegisterUser
Actors	**CASPAR** archive system user (involving the archive system administrator)
Generalisation of	
Specialisation of	
Contains	CreateUserAccount AssignDCProfileToUser (but not mandatory) AssignDefaultRole
Preconditions	
Postconditions	The user can access the archive system
Description	A User account is created
Scenario	1. The user register himself for accessing the archive system 2. User administrator examines the user request and decides if the user can be accepted or rejected. 3. The User administrator could proceed in defining all its profiles (permissions and its knowledge base), which might require interaction with the user
Example	
Alternative scenario	

5.6.2.2 UnregisterUser

Use Case Name	UnregisterUser
Actors	**CASPAR** archive system user

Generalisation of	
Specialisation of	
Contains	RevokeUserFromGroup
Preconditions	The user is registered
Postconditions	The user can not access the archive system; The user account is not removed, only the permissions are changed
Description	
Scenario	Either the user or the user administrator can trigger this use case
Example	
Alternative scenario	

5.6.2.3 Login

Use Case Name	Login
Actors	**CASPAR** archive system user, AuthenticationManager
Generalisation of	
Specialisation of	
Contains	RevokeDefaultRole
Preconditions	The user is not logged in
Postconditions	The user is logged in
Description	
Scenario	1. The user (client) provides its credentials to prove its identity 2. The user is authenticated from the AuthenticationManager
Example	
Alternative scenario	

5.6.2.4 Logout

Use Case Name	Logout
Actors	**CASPAR** archive system user
Generalisation of	
Specialisation of	
Contains	AssignDefaulRole
Preconditions	User is logged in
Postconditions	User is logged out,
Description	The working session is closed and the role of anonymous user is given to the user
Scenario	
Example	
Alternative scenario	

Note: The authentication manager will deal only with User Authentication. Component authentication will be implemented through low level operations at installation/configuration time, as the **CASPAR** architecture does not need to support a dynamic registration of new components.

5.6.3 Accounting Manager

If user accounting requirements emerge, they will be handled through the accounting manager. At a high level, the following use cases may be foreseen:

- *CreateResourceAccountingPolicy*
- *RegisterAccoutingPolicy*
- *QueryAccountingEvents*

Details of these Use Cases will be provided if the requirement materialises.

5.7 DIGITAL RIGHT MANAGER USE CASES

Digital Rights Management (DRM) is a complex process, and becomes even more complex in the context of long term preservation, moreover this is an area where OAIS is essentially silent. It is therefore important to introduce and define some terms analysed in this section:

- License offer template: a file containing incomplete Rights Expression Language (REL) expressions. It can be used as template to produce a *license offer document.*
- License offer document: a document where the license policy is completely specified by means of some REL language, but the actual parameters need still to be specified.
- License instance: a document that instantiates a *license offer document* by specifying all parameters, namely the date of the contract, the involved entities, etc.
- Licensed object: the resource, or a collection of resources, that are the object of a license offer. Needs to be further specified. We suppose that a licensed object corresponds exactly to the content of one AIP.

The actors involved in the use cases are:

- License manager (archive external staff member) - The license manager has the ability to create license offers, assisted by a proper GUI. He/she is responsible for managing the license offers templates and the license offer documents. He/she is external to the OAIS archive, specifically his/her role would map to the “OAIS Management” entity.

- Ingest staff member (archive internal staff member) - The Ingest staff members are responsible, among others, for generating the Archival Information Packages (AIPs) and for coordinating updates to the Archival Storage and Data Management. They are involved in the operation of binding the content of AIPs to one or more license offer documents.

- DRM manager (archive internal staff member) - The DRM manager represents several experts, who have the ability to manage and administer all aspects of the DRM sub-system. They belong to the OAIS internal archive staff; for instance they are responsible for updating the semantics of the license offers and license instances, but without updating the licenses themselves.

- Consumer - This actor includes the buyers of the licenses, namely the License Principals.

- Producer - This actor represents the Rights-holders, namely the Content Creators and the Content Providers.

- Access module - This actor represents a functional entity of the OAIS archive, which interacts with the DRM module.

5.7.1 List of Use Cases

In the following, some use cases that involve the DRM module are listed. They represent the high-level operations of the users, both internally and externally of the archive.

5.7.2 Definition of Rights

These use cases refer to the management of license offer documents and templates.

- Create license offer document (or a license offer template).
- Import license offer document (or a license offer template).
- Update license offer document (or a license offer template).
- Insert description about license offer document.
- Publish license offer document.
- Bind license offer document.
- Search for license offer document (or a license offer template).

5.7.3 Distribution of Rights

These use cases refer to the interactions of consumers with the overall archive system.

- Retrieve information about license terms.
- Get license.
- Watermark licensed object.
- Scramble content

5.7.4 Verification of Rights

These use cases refer to the interactions between the DRM sub-system and the consumer, once the latter has become the principal of a license instance. This kind of interactions may also occur after the delivery of a DIP containing the licensed object. Furthermore, the interaction may occur completely or partially transparently to the consumer.

- Check license instance.
- Unscramble content.
- Track content usage.

5.7.5 Management and preservation of DRM

These use cases model the operations of archive internal staff members, in particular, those that are responsible for preservation activities related to DRM. In fact, these operations are necessary to deal with the dynamicity of the license semantics and of the DRM technologies.

- Update license interpretation
- Update license parameter

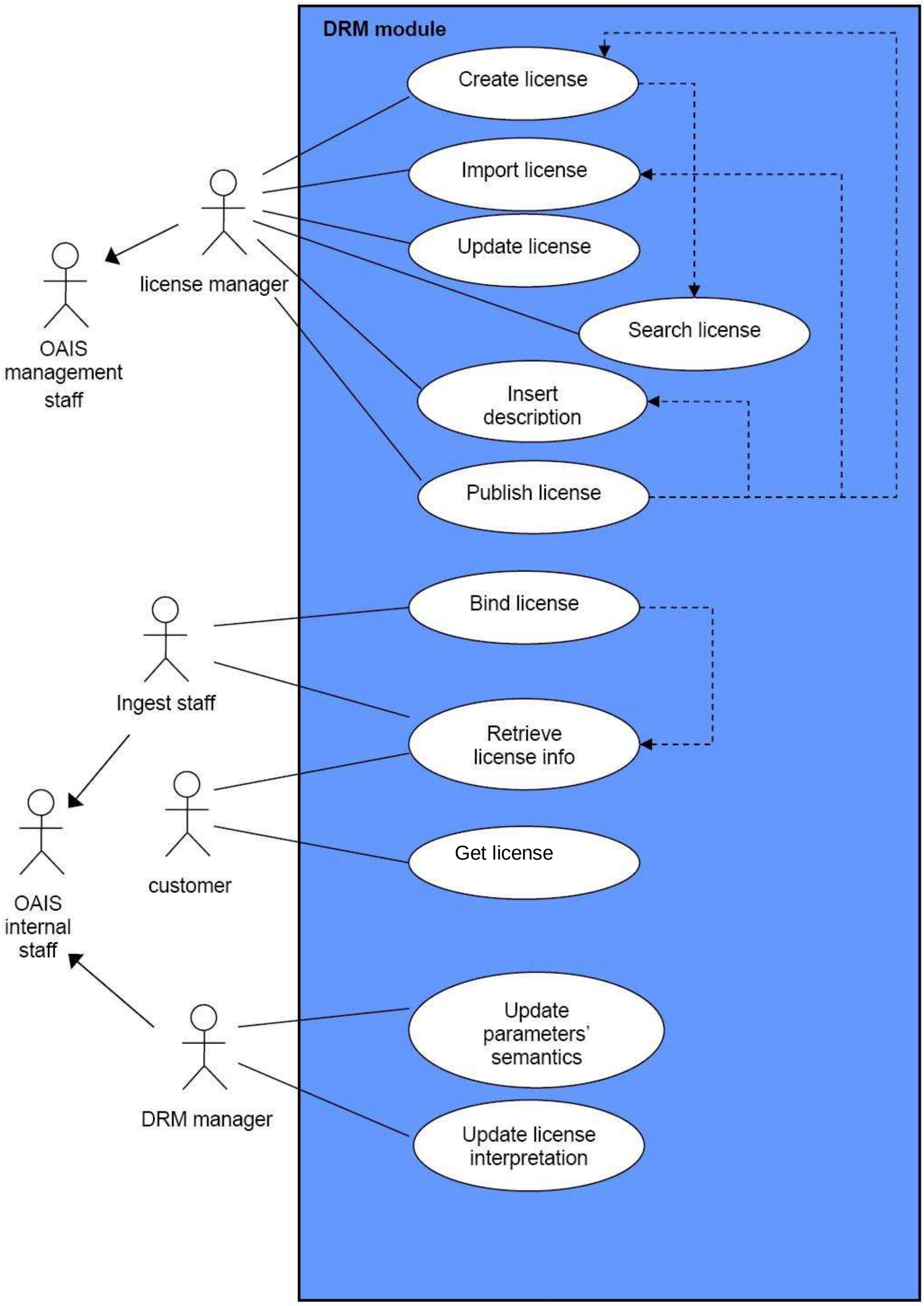


Figure 10 DRM use cases

5.7.6 Create license offer

Name	Create license offer
Actors	License manager (OAIS Management)
Generalisation of	
Specialisation of	

Contains	
Preconditions	The actor has a formal or informal document with a complete specification of the license offer to be produced.
Postconditions	A REL-syntax validated license offer template or document is stored, but still not published, i.e. ready to be bound to a licensed object.
Description	To create a new license offer document or a new license offer template. Note: The creation may occur from scratch, or may start from an existing license offer template.
Scenario	1. the actor accesses to the license management GUI 2. the actor starts composing 3. he/she opens a new file 4. he/she opens a template file 5. he/she composes a license offer following the specifications document, by selection and editing of REL expressions, parameters and values, assisted by a proper GUI. 6. he/she may complete the license offer, thus creating a license offer document, or he/she may leave the license offer incomplete, thus creating a license offer template. 7. he/she saves the file and validates it against the REL syntax
Example	
Alternative Scenario	

5.7.7 Import license offer

Name	Import license offer
Actors	License manager (OAIS Management)
Generalisation of	
Specialisation of	
Contains	
Preconditions	There exists a file containing a license offer document or a license offer template
Postconditions	A REL-syntax validated license offer template or document is stored, but still not published, i.e. ready to be bound to a licensed object.
Description	To import into the license repository an existing license offer document. Note: The source license offer document is specified as a REL document; it is envisaged to support translation, where applicable, from other REL languages into the REL used internally to the system. However, templates can not be translated.
Scenario	1. the actor accesses to the license management GUI 2. the actor selects ‘Import license offer’

	3. the actor selects the source REL language among those that are supported 4. the actor selects a file; if the file contains a template, the REL must be the same as the internally used REL 5. the actor saves the file and validates it
Example	
Alternative Scenario	

5.7.8 Update license offer

Name	Update license offer
Actors	License manager (OAIS Management)
Generalisation of	
Specialisation of	
Contains	
Preconditions	The license offer document or license offer template is stored in the licenses repository
Postconditions	A REL-syntax validated license offer template or document is stored, but still not published, i.e. ready to be bound to a licensed object.
Description	To modify a license offer (document or template) that is stored in the repository. Note: The update does not impact on the license instances that have been generated from the previous version of the license offer document.
Scenario	1. the actor accesses to the license management GUI 2. the actor selects ‘Update license offer’ and chooses the file to be updated 3. the GUI shows the content of the license offer, allowing to modify REL expressions, parameters and values, in an assisted way, i.e. not all elements are directly editable 4. the actor saves the file and validates it
Example	
Alternative Scenario	

5.7.9 Insert description about license offer

Name	Insert description about license offer
Actors	License manager (OAIS Management)
Generalisation of	
Specialisation of	
Contains	
Preconditions	The license offer document (or template) is stored in the licenses repository

Postconditions	If a license offer document has all the mandatory metadata specified, it can be published and instantiated.
Description	To provide the descriptive information about a license offer document or a license offer template. Note: It is mandatory to complete this operation before instantiating a license offer document. The information inserted with this operation will be used for searching and retrieving of licenses as well as for a description of the license terms in a human language, i.e. it must be understandable by the customer.
Scenario	1. If the use case is triggered by the system during an AIP's publication operation, the actor is notified through a message 2. the actor accesses the license management GUI 3. the actor opens the file containing a license document of template, then he/she selects 'Add metadata' 4. A form appears where all the mandatory metadata can be inserted, such as an identification of the legal framework, a category of licenses, and others. 5. the actor fills in all or some of the fields and saves.
Example	
Alternative Scenario	

5.7.10 Publish license offer document

Name	Publish license offer document
Actors	License manager (OAIS Management)
Generalisation of	
Specialisation of	
Contains	Insert description about license offer
Preconditions	The license offer document has been created or imported and is now stored in the licenses repository. Moreover it has passed the validation check and all the mandatory metadata have been inserted.
Postconditions	The state of the license is 'published'; this means that the license offer document can be used for binding it to AIPs, thus it can be instantiated, and furthermore, it becomes part of the candidate set for license searching
Description	To have a license offer document suitable to be used in the system.
Scenario	1. the actor accesses to the license management GUI and opens a license offer document 2. if the state of the license offer document is suitable for publication, the button 'Publish' is active 3. the actor selects the button 'Publish'

Example	
Alternative Scenario	

5.7.11 Bind license offer document

Name	Bind license offer document
Actors	Ingest staff member (OAIS internal staff)
Generalisation of	
Specialisation of	
Contains	
Preconditions	The license offer document is stored in the licenses repository and passed the validation check. The target licensed object corresponds exactly to the content of one AIP. The AIP is stored in the Archive Store and may be available for consumers or not.
Postconditions	From that moment the access and usage of the AIP's content are subjected to the license conditions.
Description	To bind a license offer document to a license object Note: more than one license offer documents can be associated to a licensed object.
Scenario	1. If the use case is triggered by the system during an AIP's publication operation, the actor is notified through a message 2. the actor accesses a proper toolbox for managing AIPs 3. the actor selects an AIP, then he/she selects 'Bind license offer' and chooses a file containing the license offer document
Example	
Alternative Scenario	

5.7.12 Search for license offer

Name	Search for license offer
Actors	License manager (OAIS Management)
Generalisation of	
Specialisation of	
Contains	
Preconditions	None
Postconditions	No changes in the state of the system
Description	To search for existing license offer documents or templates.
Scenario	1. the actor accesses the license management GUI 2. the actor selects 'Find license offer document' or 'Find license offer template' 3. a form appears where the actor can insert metadata values on the basis of which the search for license offers is based on

	4. a list of license offer documents or templates is shown 5. if the actor selects an item from the list, and selects ‘Open’, the file is opened in editing mode 6. if the use case was triggered by the system, for instance during the operation of binding a license offer document to (the content of) an AIP, then the actor can select a license offer document and select some other action to perform, instead of open it in editing mode
Example	
Alternative Scenario	

5.7.13 Retrieve information about license terms

Name	Retrieve information about license terms
Actors	Consumer
Generalisation of	
Specialisation of	
Contains	
Preconditions	The actor is interested in a digital content that is archived in one or more AIPs, and each one has one or more license offer documents associated to it
Postconditions	No changes in the state of the system
Description	To get all the information about the license terms before deciding to buy a license (offer document).
Scenario	1. the actor is accessing a proper toolbox and is browsing among (the index of) available digital contents (DIPs) 2. the actor selects one item of digital content (DIP) and selects ‘View license terms’ 3. a list of the licensed objects that are contained within the DIP is shown 4. the actor selects a licensed object and selects ‘View license terms’ 5. the terms within the license offer document are shown, i.e. they are expressed in a suitable form that is understandable by a human 6. the actor selects ‘Close’ and decides if he wants to proceed and buy the license
Example	
Alternative Scenario	

5.7.14 Get license

Name	Get license
Actors	Consumer

Generalisation of	
Specialisation of	
Contains	
Preconditions	The actor has viewed the license terms and does agree with them
Postconditions	The actor has all the necessary permissions and the associated medium to perform the actions granted in the bought license
Description	To obtain the license to perform the desired actions on a particular licensed object. Note: the licensed object corresponds exactly to the content of one AIP. If a DIP requires more than one license, the actor must complete the buying operation independently and for each licensed object.
Scenario	1. the actor is accessing a proper toolbox and is browsing among (the index of) available digital contents (DIPs) 2. the actor selects one item of digital content, which corresponds to a DIP, and selects 'Get license' 3. a list of the licensed objects that are contained within the DIP is shown 4. the actor selects a licensed object one at a time and selects 'Get license' 5. the terms within the license offer document are shown, i.e. they are expressed in a suitable form that is understandable by a human 6. the actor selects 'Get license' 7. if necessary, the actor is connected to a payment system and he/she completes the financial transaction 8. the actor completes the license obtainment transaction by inserting all the necessary data through the form 9. the actor receives, for instance by e-mail a confirmation, along with all the resources for performing the granted actions, such as public keys, passwords and identification to access services, and similar
Example	
Alternative Scenario	

5.7.15 Watermark licensed object

Name	Watermark licensed object
Actors	Access module
Generalisation of	
Specialisation of	
Contains	
Preconditions	The transaction of licensing the content has been completed, and the consumer has become the principal of a license instance

	related to the mentioned content. Moreover, the consumer has just submitted the request to receive the licensed object, for instance through a web-based GUI. The DRM module has already finished to check the license instance.
Postconditions	The Access module owns a digital content suitable for submission, and it can proceed with the packaging procedure.
Description	To generate a distributable version of the licensed object. Note: the licensed object corresponds exactly to the content of one AIP, so the watermark is related to the content of one AIP, even if the final DIP contains more than one AIP.
Scenario	1. The Access module requests watermarking of one AIP 2. the DRM module retrieves the information from the repositories which is required to produce the watermark; the kind of information depends on the type and function of the watermark, and may be for instance a private key, an identifier of the requesting consumer or of a rights-holder. 3. the DRM module generates the watermark and applies it to the digital content 4. the DRM module sends the watermarked content to the Access module
Example	
Alternative Scenario	

5.7.16 Scramble content

Name	Scramble content
Actors	Access module
Generalisation of	
Specialisation of	
Contains	
Preconditions	The license instance has been checked to verify the consumer's permissions
Postconditions	The content is scrambled, for instance through encryption or through watermarking to alter the content's rendering quality
Description	The content data is scrambled, e.g. through encryption or watermarking, in order to limit access operations or content quality
Scenario	1. The access module sends the request with all necessary parameters for the particular scrambling operation 2. The content file(s) are retrieved 1. A new file with the scrambled content is produced and sent to the access module
Example	
Alternative Scenario	

5.7.17 Check license instance

Name	Check license instance
Actors	Access module
Generalisation of	
Specialisation of	
Contains	
Preconditions	The consumer has agreed on the license terms and already owns a license instance that grants him the necessary permissions.
Postconditions	The access module is free to send the DIP
Description	To verify if a certain action performed on a licensed object is granted to the consumer. Note: this use case refers to the verification before completing the submission through the Access module. It could also cover the verification of rights when the licensed object has been sent and exists outside of the preservation platform.
Scenario	1. the consumer has agreed on the license terms 2. the consumer has obtained a license instance 3. the access module checks the license instance before sending any licensed object
Example	
Alternative Scenario	

5.7.18 Unscramble content

Name	Unscramble content
Actors	Basic SW on the client side or specific tools
Generalisation of	
Specialisation of	
Contains	
Preconditions	The content file is locally stored, the necessary SW to complete unscrambling is installed, the scrambling was aimed at securing content from unauthorized usage
Postconditions	The content file is ready to be rendered on the customer's side
Description	To apply some transformation to the content file(s) aimed at verifying the permissions and finally at producing a file ready to be rendered; at the basis is encryption
Scenario	1. The consumer follows the instructions to unscramble the content, or the application guides him to do so 2. The consumer inserts some data, if necessary 3. The private keys are retrieved for description

	4. The content file is ready to be opened by the rendering application
Example	
Alternative Scenario	

5.7.19 Track content usage

Name	Track content usage
Actors	Specific tools that may be installed partly on the consumers' side and partly on special servers
Generalisation of	
Specialisation of	
Contains	
Preconditions	The consumer has some content data stored locally
Postconditions	The content usage instance has been traced by some accounting system or registered persistently within the content file
Description	To monitor and control the consumer's usage instances of the content
Scenario	1. The consumer uses a particular application or some plugged-in tool to open the content file 2. The tool interacts with some server or applies some watermarking to register the usage instance within the content file 3. The content is rendered and the usage tracking is done transparently to him
Example	
Alternative Scenario	

5.7.20 Update interpretation of licenses

Name	Update interpretation of licenses
Actors	DRM Manager (OAIS internal staff member)
Generalisation of	
Specialisation of	
Contains	
Preconditions	
Postconditions	All licenses that are affected by the change and that will afterwards be interpreted by machine to enforce rights protection, will reflect the updates in the license terms semantics
Description	To update the semantics of the licenses without modifying themselves. Note: this use case is not defined in detail, and collects all possible operations required to intervene in the interpretation of some licenses where it is not feasible to update the licenses themselves, for instance to update it with changes in legal

	frameworks.
Scenario	1. The DRM Manager accesses DRM management subsystem 2. He performs the update of the license “interpretation engine” 3. He specifies the extent of the update, e.g. the licenses related to that update 4. A log is created to take trace of the semantics update, and to allow reporting and re-do if necessary
Example	
Alternative Scenario	

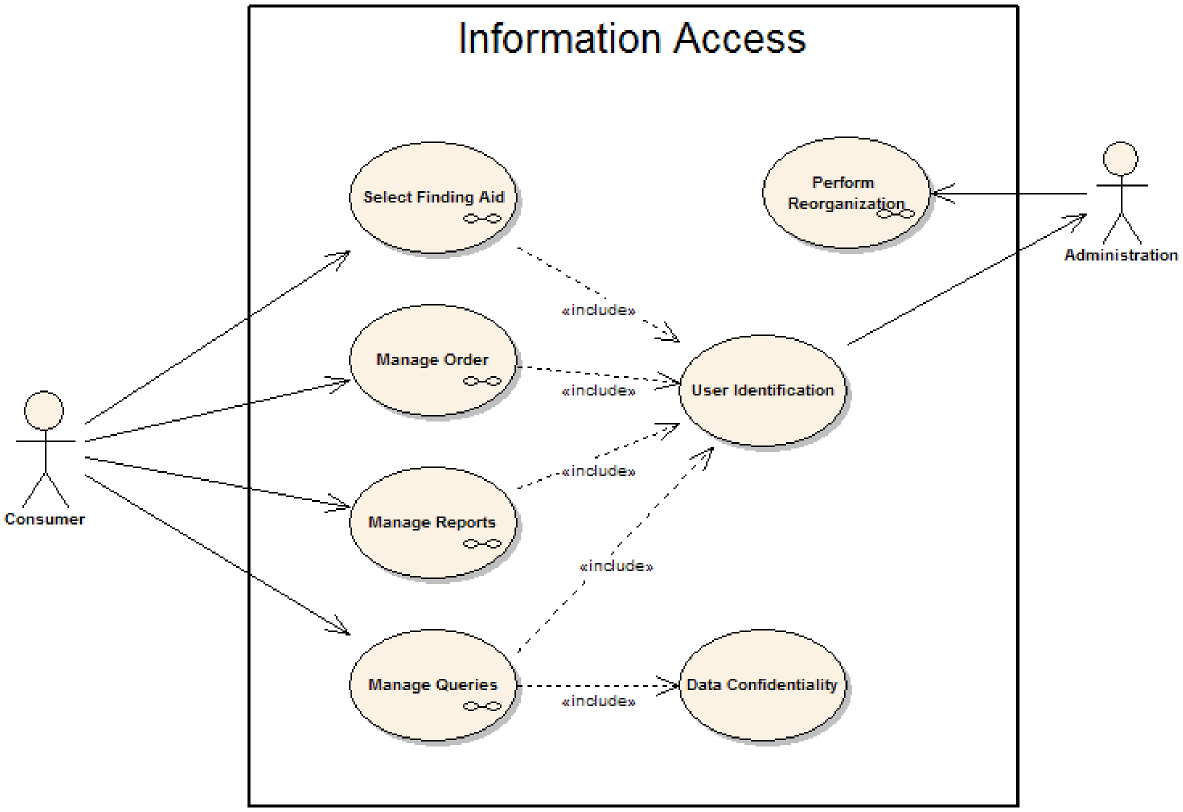
5.7.21 Update semantics of license parameters

Name	Update semantics of license parameters
Actors	DRM Manager (OAIS internal staff member)
Generalisation of	
Specialisation of	
Contains	
Preconditions	The license interpretation engine, if on the client, requires interaction with some server in order to be aligned with the updates
Postconditions	The license document (i.e. all its instances) that are affected by the change and that will afterwards be interpreted by machine to enforce rights protection, will reflect the updates in the license terms semantics
Description	To update the semantics of the licenses without modifying themselves. Note: this operation is required to perform a change that does not affect the license statements, but rather affects the ‘knowledge base’ related to the license, such as the life span of persons.
Scenario	1. The DRM Manager accesses DRM management subsystem 2. He identifies the particular license document and performs the update 3. If the interpretation must be updated also for license instances that have already been obtained by consumers, then a proper synchronization mechanism must be applied
Example	
Alternative Scenario	

5.8 FINDING AIDS USE CASES

The Finding Aids are an important part of any archive (other than perhaps a “dark archive”), and all existing archives will have their own brand of such tools, each with its own array of capabilities tailored to its customers’ needs. CASPAR does not aim to duplicate the great variety of capabilities which already exist, but rather to provide some generic, extensible, capabilities as well as a number of novel capabilities based on OAIS concepts.

The Finding Aids support the user application in locating the relevant data; the Finding Aids’ primary actors are the Consumer and the Administrator. The generic user (a Consumer, or an Administrator) can establish a Search Session with **CASPAR**. During this Search Session the user will employ the **CASPAR** Finding Aids to identify and investigate potential holdings of interest. This may be accomplished by the submission of queries and the return of result sets to the user. This searching process tends to be iterative, with a user first identifying broad criteria and then refining these criteria based on previous search results. **CASPAR** could also limit the result set returned to the user, to enforce the underlying intellectual property rights and restricting the user access exclusively to the items (s)he is entitled to see. In the case of Consumers querying the system, once they have identified the AIPs of interest, they may provide an Order Agreement that documents the identifiers of the AIPs they wishes to get, and how the DIPs will be acquired from **CASPAR**.

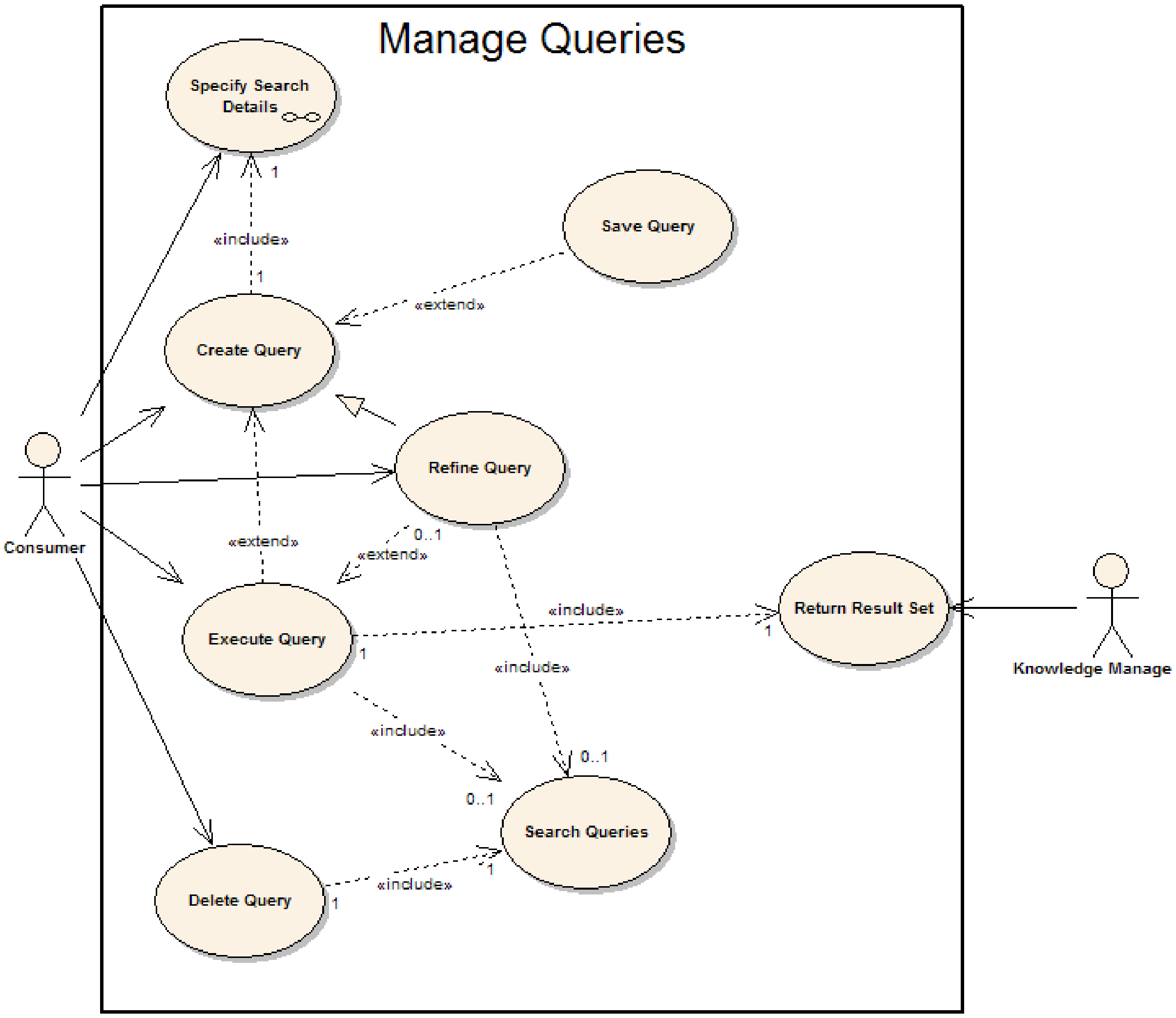


5.8.1 Select Finding Aid

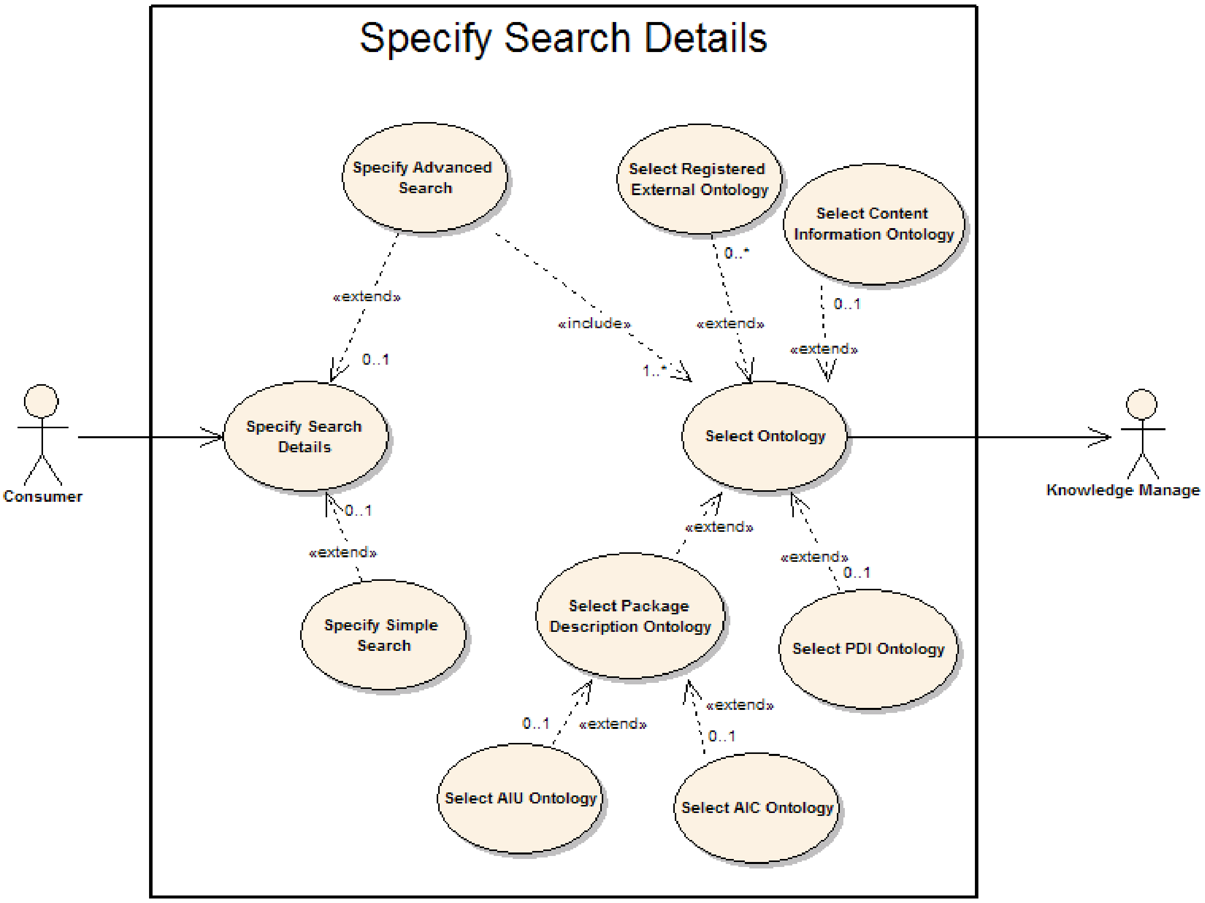
This use case is triggered when the Consumer wishes to define, save, retrieve or refine a previously saved query, report or order. The query functionality is at the core of any of these services, as reports can be seen as the composition of a query plus aggregating and formatting rules, while orders can be seen as the composition of queries plus other information related to the transformation, packaging and delivery of the information to be disseminated. Users can save and retrieve the finding aids they have defined if they like, or take advantage of the **CASPAR** predefined, published ones.

5.8.2 Queries

The basic component of Finding Aids are queries, which return sets of unique AIP locators.



Queries are available in two flavours: Simple Search and Advanced Search. In the first case the user just specifies keywords and logical operators, and the system will search all descriptive information for textual matching; for the search purpose, the descriptive information will be considered as flat text, even if properly structured. This search still remains one of the most used for its simplicity: no particular competencies are required to the issuer – which, in case of an interfacing software component, means no need for implementation changes. Being the flat text search so common in systems, this option provides great interfacing capability, at a very low price. In the Advanced Search case, instead, the user is presented with a more sophisticated interface, allowing him or her to perform a search based on one or more system-specific or external ontologies – as long as they have been previously registered into the system. This option is much powerful, letting the requester specify in a very precise manner the details of the intended results, and will generally lead to more accurate solutions. The adoption of ontologies to express the Descriptive Information is somewhat driven by the fact that ontologies may have also been chosen to represent knowledge about the PDI (Preservation Description Information) and the Representation Information, and those information may be considered in the Descriptive Information.



Use Case Name	Specify Search Details
Actors	Consumer (Primary), Knowledge Management
Generalisation of	
Specialisation of	
Contains	
Preconditions	This is called as part of the Create or Refine Query.
Postconditions	A set of search details is defined, i.e. the intentional definition of the desired Result Set.

Description	This use case is triggered when the consumer needs to specify the search details associated with a query; this interaction is initiated as part of a Create Query or Refine Query use case. Two alternatives are possible: the specification of simple (free text) search details, or the specification of complex (i.e. ontology-based) search details. In any case, those details are returned to the calling use case.
Scenario	1. The consumer wishes to specify the search details in order to be used in a particular query. 2. The consumer is given the option to perform either a simple search or a more sophisticated advanced search. Consumer's selection initiates one of the two possible mutually exclusive alternative scenarios.
Example	
Alternative scenario	3. Simple Search 3a. The consumer is asked to specify a set of keywords to the system; these will be used to make a simple free text search upon the records stored in the system. 3b. The selected keywords are returned to the Create Query or Refine Query use case that spawned the Specify Search Details use case. 4. Advanced Search 4a. In this case the consumer must provide or specify: i. system-specific or external ontologies, like: Content Information Ontology, PDI Ontology or Package Description Ontology. In addition to the above, the consumer has the option to select an arbitrary number of external ontologies upon which to base his search; such external ontologies may have to be registered first, through the appropriate interface (not a part of this use case). ii. the exact search criteria to apply on the selected ontologies.

5.8.3 Manage Order

After having identified one or more AIPs of interest, eventually with the help of the Finding Aids, the Consumer can collect them in an Order, specifying also how those AIPs are to be transformed and mapped into Dissemination Information Packages (DIPs) and how those DIPs will be packaged in a Data Dissemination Session. The order is then submitted to the **CASPAR** Administration for negotiation; when an agreement is reached, the Consumer can place the order. Then, according to the agreement, a Data Dissemination Session takes place, and the information is dispatched to the Consumer. In the case of an Event Based Order, i.e. an order regarding AIPs not yet ingested into **CASPAR**, the Consumer establishes an order agreement with **CASPAR** for information expected to be received, on the basis of some triggering event. This event may be periodic, such as a monthly distribution of any AIPs ingested from a specific Producer, or it may be a unique event such as the ingestion of a specific AIP.

5.9 VIRTUALISATION USE CASES

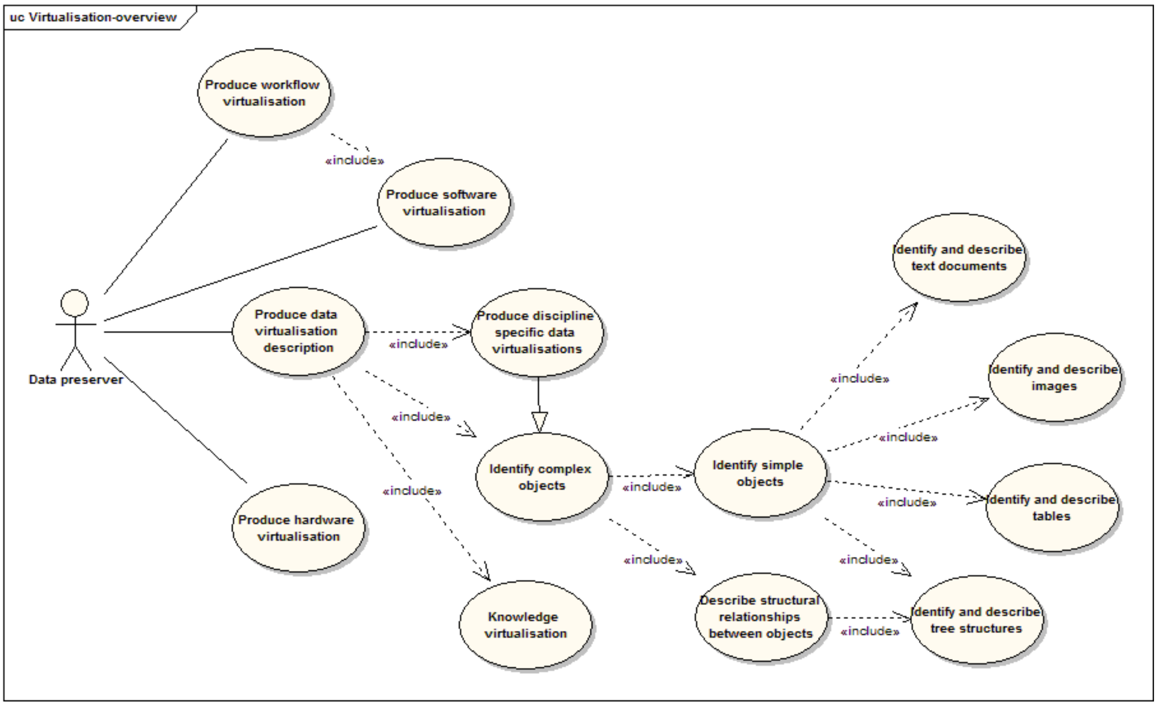

uc Virtualisation-overview
Produce workflow virtualisation
«include»
Produce software virtualisation
Data preserver
Produce data virtualisation description
«include»
Produce discipline specific data virtualisations
«include»
«include»
Identify complex objects
«include»
Identify simple objects
«include»
«include»
«include»
«include»
Produce hardware virtualisation
Knowledge virtualisation
«include»
Describe structural relationships between objects
«include»
Identify and describe text documents
Identify and describe images
Identify and describe tables
Identify and describe tree structures

Figure 11 Virtualisation Use Cases

5.9.1 Produce data virtualisation description

Use Case	Produce data virtualisation description
Actors	Data preserver
Generalisation of	none
Specialisation of	none
Contains	Produce discipline specific data virtualisations; Identify complex objects; Knowledge virtualisation
Preconditions	Actor has a particular type a data to be described and access to adequate expert knowledge about that data.
Postconditions	The user will have Representation Information which provides virtualisation of the data in question. Alternatively it may be that no techniques are currently available to produce a suitable virtualisation description.
Description	The data virtualisation tool is used by a human user to create a virtualisation description of a particular piece, or type, of data. The tool: - provides the user with a number of menus and questions about the data - depending upon the answers, a number of more specific tools are presented, each of which address specific types of virtualisation. - those more specific tools are configurable by the user and also may

	obtain information from other sources such as a Registry Repository of Representation Information. The user can save a work in progress and return to it later, and a graphical representation of the current status can be displayed. The virtualisation description may consist of a number of different types of virtualisations, from a number of different tools, using a number of different underlying techniques. The final output will be a package containing all these types of descriptions.
Scenario	The user has a particular piece of data. The tool may be accessed via a web browser (or equivalent) or installed as a stand-alone application. In the process of asking questions and presenting appropriate menus, the tool may use a local configuration file and/or one or more Registry/Repositories of Representation Information. The latter may also be a source of some of the more specific tools
Example	The user has a piece of astronomical X-ray data. This consists of a number of measurements which are taken over an extended time period. The measurements record the arrival of X-ray photons from an astronomical source on points on a a detector. This means that the data is essentially tabular i.e. it consists of a number of columns, each of which corresponds to a particular measurement, while the rows correspond to the successive sets of measurements, each row containing the values for a specific time. However by adding up the photons at each “pixel” on the detector, one can look at the data as an image made up of these pixels. The virtualisation tool presents the user with a number of options. The user first selects the option which indicates that the data is a table. There may be a selection of tools which capture descriptions of tables, selectable by the user. The selected tool for producing the table description is started and enables the user to describe the way in which the data may be viewed as a table. The user may return to the higher menu and start the process of describing the data, this time as an image. Note that these descriptions are ideally captured in some formal syntax (which of course has its own Representation Information), but may have to be captured as a text description (again with Representation Information).
Alternative scenario	

5.9.2 Produce specific type of data virtualisation description

Use Case	Produce specific type of data virtualisation description
Actors	Data preserver
Generalisation of	none

Specialisation of	none
Contains	One or more increasingly specific Use cases – see the alternative scenarios – illustrated in the Use Case diagram
Preconditions	User has been led through a number of selections to this specific type of virtualisation
Postconditions	A specific type of virtualisation description has been produced
Description	A specific tool is presented to the user. This may present the user with a number of alternatives and options. The details of how each tool then proceeds depends upon that tool.
Scenario	
Example	
Alternative scenario	- discipline specific virtualisation: the tool may be tailored according to some input from the RepInfo Gap Manager or a Registry/Repository, in order to present the user with some discipline specific tool i.e. some tool which is in the a specific discipline knowledge base - complex object virtualisation: the tool allows the user to identify a number of subcomponents, each of which can be described further - structural relationship virtualisation: the relationships between components is described using one or more techniques, including Semantic Web techniques or text descriptions - simple object virtualisation: focuses on a objects which do not naturally break down into sub-components other than primitive values (e.g. individual numbers) - table/image/tree/document/etc virtualisation: describing the specific simple sub-component type - Knowledge virtualisation: part of the Knowledge Management suite of use case

5.9.3 Produce hardware/software virtualisation description

Use Case	Produce hardware virtualisation description
Actors	Data preserver
Generalisation of	none
Specialisation of	none
Contains	Produce specific hardware virtualisations;
Preconditions	Actor has a particular type of hardware in mind to be described and access to adequate expert knowledge about that hardware.
Postconditions	The user will have Representation Information which provides virtualisation of the hardware in question. Alternatively it may be that no techniques are currently available to produce a suitable virtualisation description.

Description	The hardware virtualisation tool is used by a human user to create a virtualisation description of a particular piece, or type, of hardware. The tool: - provides the user with a number of menus and questions about the hardware - depending upon the answers, a number of more specific tools are presented, each of which address specific types of virtualisation. - those more specific tools are configurable by the user and also may obtain information from other sources such as a Registry Repository of Representation Information. The user can save a work in progress and return to it later, and a graphical representation of the current status can be displayed. The virtualisation description may consist of a number of different types of virtualisations, from a number of different tools, using a number of different underlying techniques. The final output will be a package containing all these types of descriptions.
Scenario	The user has a particular piece of hardware in mind. The tool may be accessed via a web browser (or equivalent) or installed as a stand-alone application. In the process of asking questions and presenting appropriate menus, the tool may use a local configuration file and/or one or more Registry/Repositories of Representation Information. The latter may also be a source of some of the more specific tools
Example	The user has a Macintosh computer on which some specific software runs. The tool directs the user to a number of different hardware virtualisation techniques including CPU, disk, display etc.
Alternative scenario	

5.10 PACKAGING COMPONENT USE CASES

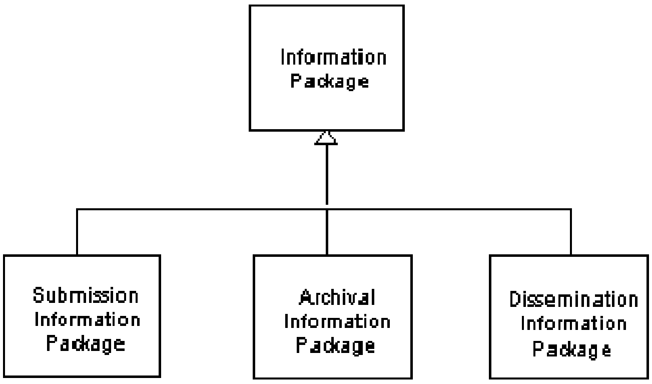

Information Package
Submission Information Package
Archival Information Package
Dissemination Information Package

Figure 12 OAIS Information Package Taxonomy

As shown in Figure 12, OAIS describes three subtypes of Information Package (IP), a Submission Information Package (SIP), the Information Package delivered by the Producer to the OAIS for use in the construction of one or more AIPs, a Dissemination Information Package (DIP), The Information Package, derived from one or more AIPs, received by the Consumer in response to a request to the OAIS.

The only one which OAIS describes in detail is the Archival Information Package (AIP), which is conceptually vital for the preservation of a digital object. According to OAIS the *AIP is defined to provide a concise way of referring to a set of information that has, in principle, all the qualities needed for permanent, or indefinite, Long Term Preservation of a designated Information Object.*

The full AIP is illustrated in Figure 13

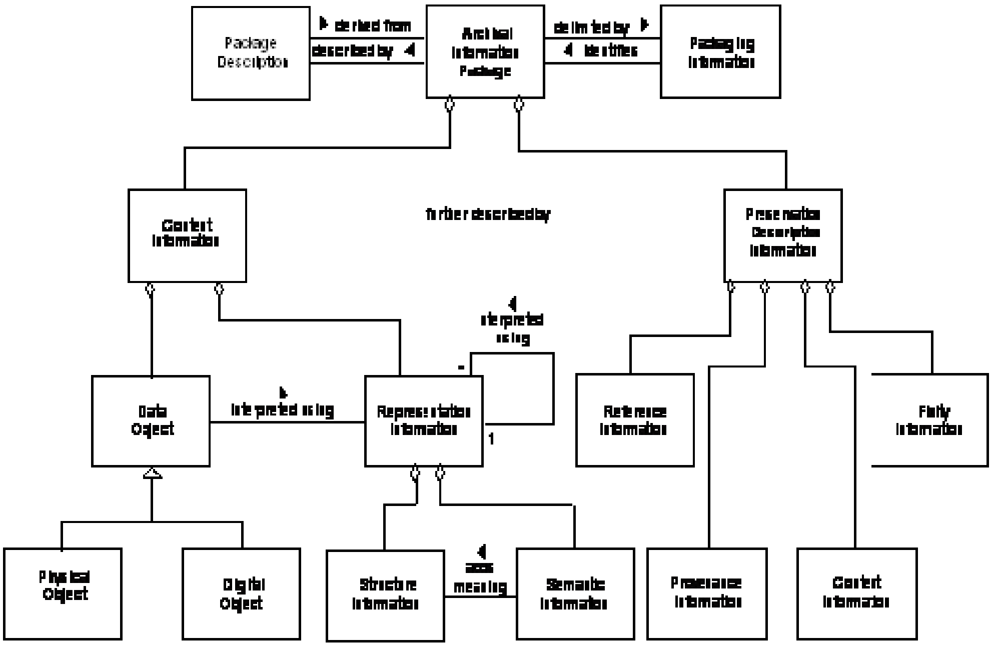


Figure 13 OAIS Archival Information Package (AIP)

Operations on Information Packages will be performed by the Package Manager toolkit interface, this may call internal operations on Information Package. Depending on the subType of Information Package a specialised operation overridden by the subType may be called. The Information Package (IP) class handles operations on the information objects stored within the IP. There are several types of Information Objects used in the OAIS. The objects are categorized by their content and function in the operation of an OAIS into Content Information objects, Preservation Description Information objects, Packaging Information objects, and Descriptive Information objects. Figure 1.2 shows a taxonomy of those Information Objects used within the OAIS.

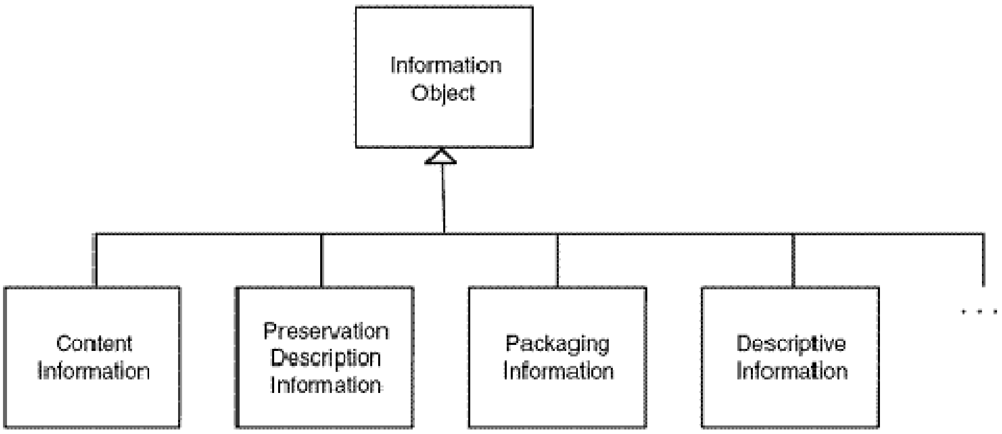


Figure 14 OAIS Information Object Taxonomy

OAIS defines as including the following types of Information Object:

Content Information: The set of information that is the original target of preservation. It is an Information Object comprised of its Content Data Object and its Representation Information.

Preservation Description Information (PDI): Information that is necessary to adequately preserve the particular Content Information with which it is associated such as Fixity, Reference, context and provenance information.

Packaging Information: Information which, either actually or logically, binds or relates the components of the package into an identifiable entity on specific media.

Descriptive Information: The Information Objects necessary to enable the Long Term Preservation function of the archive. The Information Objects described previously in this section provide the information necessary to enable the Long Term Preservation function of the archive.

Each information object added to an Information Package will be called a content unit and identified by a content unit ID. Each Information Package object will be given a unique identifier IP ID.

The packaging component includes the following high level use cases which would for the basis of the PackageManager interface.

The toolkit interface will perform operations on the Information Package (IP) class, the super type of SIP, AIP and DIP.

The logical contents of the AIP is defined in OAIS and hence a certain amount of specific things can be said about it e.g. one must be able to obtain PDI information from an AIP. The DIP and SIP one the other hand are rather general concepts which label a huge variety of instance specific variations, and hence very little can be said about these.

5.10.1 Construct IP

Use Case Name	Construct IP
Actors	Packaging
Generalisation of	• Construct SIP • Construct DIP • Construct AIP – the logical contents of an AIP is specified in OAIS and hence more details can be provided
Specialisation of	None
Contains	The use case does not contain other use cases
Preconditions	None
Postconditions	IP object will be constructed and accessible by IP id
Description	Packaging use case used to construct an IP object, constructed IP is stored by applying 'store IP' use case, this use case will apply use cases specialized in each IP subtype
Scenario	(1) PackageManager instantiates a IP object (2) IP object structure is modified by adding content units to IP (3) Each content unit given automated Content Unit ID (4) IP ID is automatically generated for referencing IP (5) IP is saved to PackageManager accessible storage, using Store IP use case

5.10.2 Unpackage IP

Use Case Name	Unpackage IP
Actors	Packaging
Generalisation of	• Unpackage SIP • Unpackage DIP • Unpackage AIP – the logical contents of an AIP is specified in OAIS

	and hence more details can be provided
Specialisation of	none
Contains	The use case does not contain other use cases
Preconditions	none
Postconditions	IP object will be constructed and accessible by IP id
Description	Packaging use case used to construct an IP object, constructed IP is stored by applying 'store IP' use case, this use case will apply use cases specialized in each IP subtype
Scenario	(1) PackageManager instantiates a IP object (2) IP object structure is modified by adding content units to IP (3) Each content unit given automated Content Unit ID (4) IP ID is automatically generated for referencing IP (5) IP is saved to PackageManager accessible storage, using Store IP use case

5.10.3 Transform IP

Use Case Name	Transform IP
Actors	Packaging
Generalisation of	• Transform SIP • Transform DIP • Transform AIP – the logical contents of an AIP is specified in OAIS and hence more details can be provided
Specialisation of	none
Contains	Validated IP , UnPackageIP, getIPbyID
Preconditions	Information Package (IP) exists and is accessible by IP ID
Postconditions	IP is transformed to a new IP
Description	Successfully Validated IP is transformed to a new IP with a new IP ID. New IP is validated as being correctly structured.
Scenario	***Transform Single IP to a new IP*** (1) PackageManager is sent transform request for a given IP ID (2) PackageManager retrieves IP identified by IP ID using use case

	'getIPbyID' (3) PackageManager applies use case 'Validate IP' to check construction of IP (4) 'UnPack IP' use case applied to IP (5) PackageManager transforms UnPacked IP into a new IP with a new ID (6) PackageManager applies use case 'Validate IP' to check construction of new IP (7) Generate IP ID for new IP (8) PackageManager returns a 'Transform Complete' status flag to the original requesting actor.
Alternative Scenario	***Transform set of multiple IPs to a new IP*** (1) PackageManager set transform request on a set of IPs (2) PackageManager retrieves IPs identified by IP ID (3) PackageManager applies use case 'Validate IP' to check construction of each IP (4) When all IPs retrieved and validated, apply transform multiple IPs to new single IP identified by new IP ID (5) Apply use case 'Validate IP' to new IP (6) Generate IP ID for new IP

5.10.4 Validate IP

Use Case Name	Validate IP
Actors	Packaging
Generalisation of	• Validate SIP • Validate DIP • Validate AIP – the logical contents of an AIP is specified in OAIS and hence validation can include such things as checking that PDI is present.
Specialisation of	none
Contains	getIPbyID
Preconditions	IP exists, is accessible by IP ID and is available for validation
Postconditions	Information Package (IP) is validated as correctly structured and formatted

	and content is checked for validity. The ‘IP is valid’ flag will be set for the IP in question
Description	Structure and content of the Information Package will be validated against some specific criteria, because the structure and contents of SIPs, AIPs and DIPs may differ greatly, this use case will be specialized by each IP subtype.
Scenario	(1) PackageManager is sent an IP id to validate (2) PackageManager retrieves IP identified by IP ID using use case ‘getIPbyID’ (3) Validation method checks content and structure of IP (4) PackageManager reports validation status to appropriate Actor

5.11 AUTHENTICITY MANAGEMENT USE CASES

Authenticity cannot be measured with a Boolean flag telling us whether a document is authentic or not. There are degrees of authenticity: the certainty about authenticity is a goal and sure cases are edge cases. We asymptotically tend to authenticity. So we have to design all the mechanism and tools that could help on this way, keeping in mind that we could have alteration, corruption, lack of significant data and so on, and we need tools, mechanisms, weights to understand their relevance and their impact on authenticity. The consequence is that "to ensure authenticity" means "providing" a proper set of evidence, and technical aspects of this can be in the form of attributes related to content and context, and "verifying/checking" (possibly against a metrics) the completeness or the alteration of this set.

Authenticity Management tools have to ensure authenticity by identifying, managing and preserving information aimed to describe and possibly evaluate its identity and integrity. These tools provide content and contextual information relevant to authenticity, i.e. to the identity and integrity profile, all along the whole preservation process**.**

A crucial point is that identity must be intended in a very wide meaning: the identity of a resource refers not only to its unique designation and/or identification. Identity refers to the whole of the characteristics of a resource that uniquely identify it and distinguish it from any other resource, so identity refers not only to the internal conceptual structure of a resource but also to its overall - administrative, legal, documentary, technological, and even social - context. From this point of view, identity is strongly related to Preservation Description Information: Context, Provenance, Fixity, and Reference Information as defined in OAIS help to understand the environment of a resource, and this information has to be gathered, maintained and interpreted altogether, as much as possible, as a set of relationships defining a complex whole, and some distinctions are not entirely clear-cut..

Context Information, for example, overlaps with Provenance Information, and Reference Information. In fact, the way things are defined and/or classified changes over time, so the reference profile changes; but from an "historical" point of view this is a matter of context too, where historical refers to the history of the resource. This history tells us a lot about the life of the object and it's something we cannot miss without losing evidence of authenticity.

The integrity of a resource refers to its wholeness. A resource has integrity when it is complete and uncorrupted in all its essential respects. The verification process should analyse and ascertain they are consistent with the inevitable changes brought about by technological obsolescence.

Authenticity Management tools monitor and manage protocols and procedures across the custody chain to ensure management and preservation systems work in coherence with authenticity requirements. Authenticity is never limited to the resource itself, but it is extended to the whole information/document/record system, thus to the concept of reliability, that is to the control over the information/document/record creation process and custody.

The most critical issues about authenticity are the right attribution of authorship, the identification of provenance in the life cycle of digital resources, the insurance of content integrity of their digital components and relevant contextual relationships, and the provision of mechanisms and information to allow future users to verify the authenticity of the preserved objects or - at least - to provide the capability of evaluating their reliability in terms of authenticity presumption.

The Authenticity Management includes the following use cases:

- Create Identity Information
- Create Integrity Information

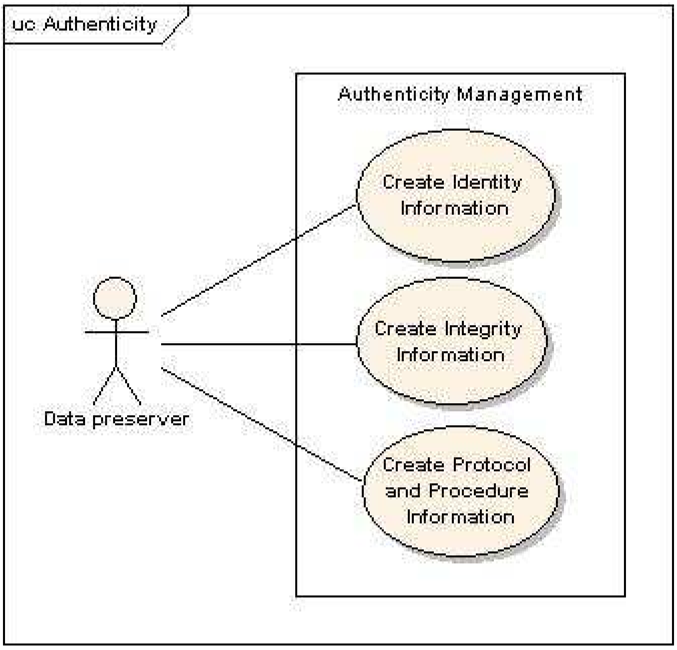
- Create Protocol and Procedure Information

The Create Identity Information refers to the identification of information elements, mainly related to:

- Provenance (with reference to archival history/chain of custody, origin or source, changes since the creation, …);
- Context (scope and content, name of action or process, archival bonds, designation, extent, medium, taxonomic systems, reference systems, registration systems, …);
- Conditions of access and use allied materials.

The Create Integrity Information refers to data integrity checks or validation/verification keys, and to procedures aimed at preventing, discovering, and correcting loss or corruption of records.

The Create Protocol and Procedure Information refers to protective strategies and/or solutions adopted in order to maintain authenticity or that affect authenticity anyway. The recursive design of information objects emphasizes the recursive nature of the problem of authenticity, so we need to manage authenticity of content/context information too, or at least define the policies for its control, for example by recording the responsibilities on creation and/or modification of content/context information.


uc Authenticity
Authenticity Management
Create Identity Information
Create Integrity Information
Create Protocol and Procedure Information
Data preserver

5.11.1 Create Identity Information

Use Case Name	Create Identity information
Actors	Data Preserver
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	Valid AIP
Postconditions	Fundamental information about provenance, context, conditions of access and use, and allied materials is created and associated to the resource

Description	Information is extracted from RepInfo, PDI, and Migration to generate a set of data covering provenance, context, conditions of access and use, and allied materials profiles. The Data Preserver evaluates the quality of information and adds any missing data needed to fulfil authenticity requirements
Scenario	1) PDS receives a "storage request" with an AIP and sends a "storage alert" to Authenticity Management. 2) Authenticity Management creates information as described above and sends an "authenticity report" to PDS, possibly requiring more qualified data. 3) When information is complete or anyway it can't be refined anymore, Authenticity Management associates it to the resource and, in case, it sends an "authenticity alert" to PDS and Administration entity

5.11.2 Create Integrity information

Use Case Name	Create Integrity information
Actors	Data Preserver
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	Valid AIP
Postconditions	Information about data integrity, and procedures aimed at preventing, discovering, and correcting loss or corruption of records is created and associated to the resource
Description	Information is extracted from Fixity to generate a set of data through which evaluate the integrity of the resource and hence its authenticity. The Data Preservers evaluate the quality of information and add any missing data needed to fulfil authenticity requirements
Scenario	1) PD receives a "storage request" with an AIP and sends a "storage alert" to Authenticity Management. 2) Authenticity Management creates information from Fixity and sends an "authenticity report" to PDS, possibly requiring more qualified data. 3) When information is complete or anyway it can't be refined anymore, Authenticity Management associates it to the resource and, in case, it sends an "authenticity alert" to PDS and Administration entity

5.11.3 Create protocol and procedure information

Use Case Name	Create protocol and procedure information
Actors	Data Preserver
Generalisation of	None
Specialisation of	None
Contains	None
Preconditions	Valid AIP
Postconditions	Information about protocols and procedures affecting authenticity is created and associated to the resource

Description	Information is extracted from RepInfo, PDI, Migration to generate a set of data aimed at express the value of preservation chain or rather the overall quality of system in terms of authenticity management. The Data Preserver may add any useful information, including responsibilities on creation and/or modification of content/context information, and evaluation or description of authenticity of content/context information.
Scenario	1) PDS receives a "storage request" with an AIP and sends a "storage alert" to Authenticity Management. 2) Authenticity Management creates information from RepInfo, PDI, Migration, and sends an "authenticity report" to PDS, possibly requiring more qualified data. 3) When information is complete or anyway it can't be refined anymore, Authenticity Management associates it to the resource and, in case, it sends an "authenticity alert" to PDS and Administration entity

6 KEY PRESERVATION COMPONENTS DESCRIPTION

This chapter describes each key preservation component in the **CASPAR** architecture.

The description of the component is given through a template indicating:

1. Name of the component;
2. Responsibilities: the purpose or the job description of the component;
3. Parts: the internal parts of the component;
4. Provided/Required Interfaces: the interfaces vs./from other components which the component provides/requires its functionalities.
5. Artefacts: the implementation of all component parts into a physical piece of software.

The Key Components and their connectivity is shown in Figure 15; details will be provided in the following sections.

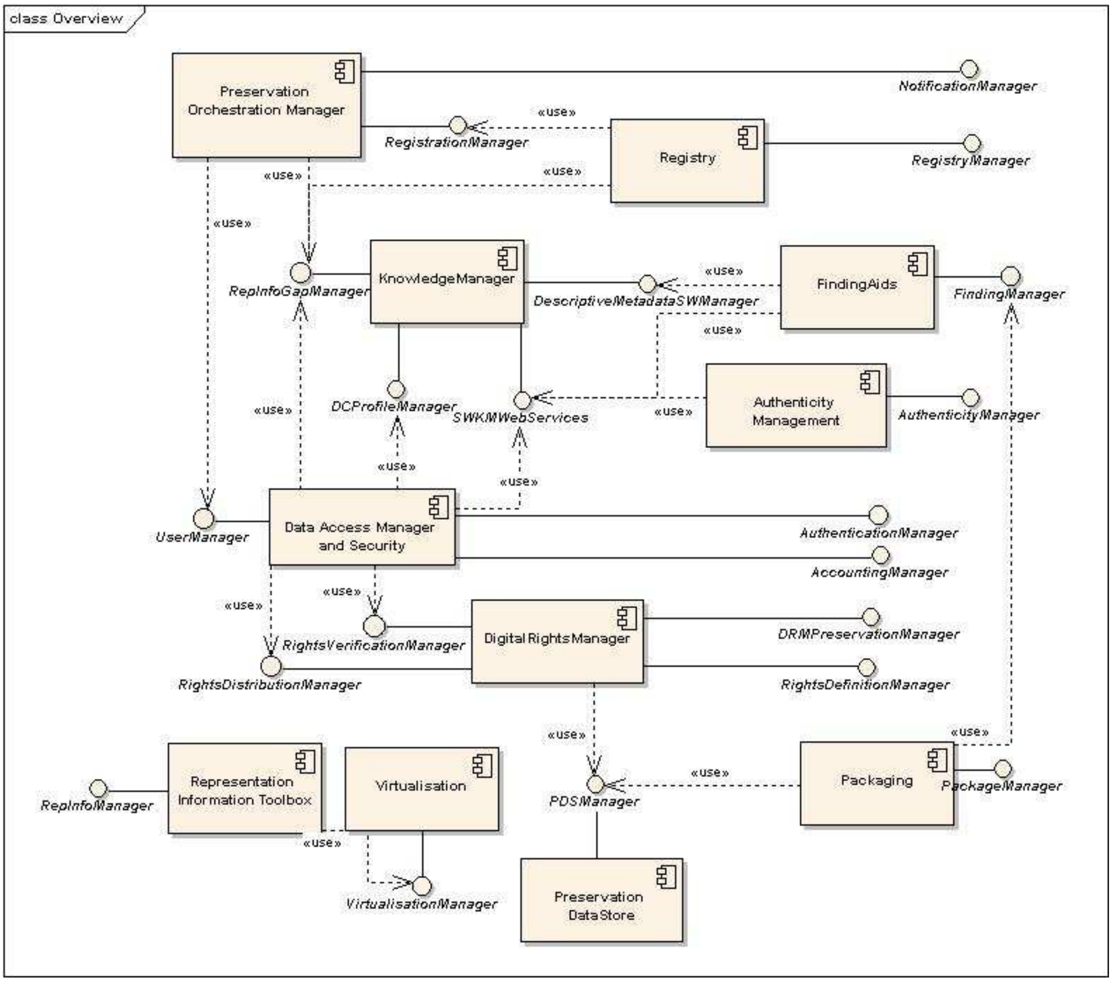


Figure 15 CASPAR component interfaces

6.1 REGISTRY

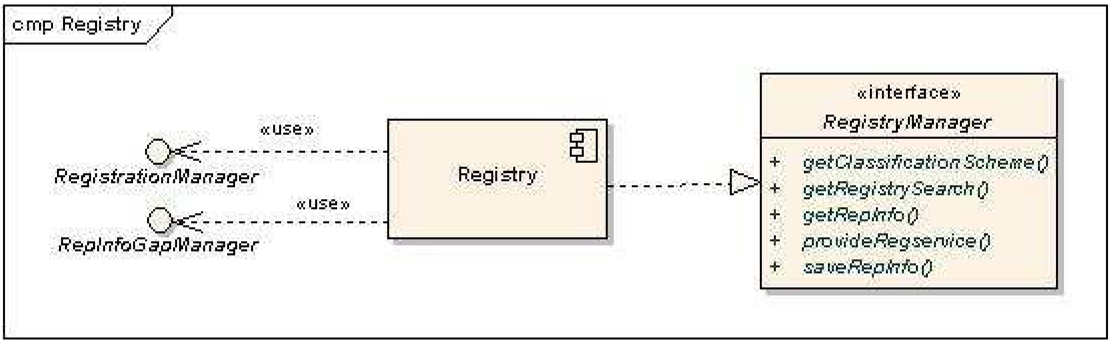
Component	Registry/Repository
Responsibilities	1. Return Representation Information, given a valid CPID 2. Ingest Representation and return a CPID 3. Return a list of Representation matching a set of selection criteria. 4. Preserve the Representation Information it holds for long-term access, as an OAIS
Parts	This modules is composed by the following parts: 1. Registry providing an o index of Representation Information o search capabilities including ▪ ability to distribute search to federated Registry/Repository 2. Repository of the Representation Information, each with an appropriate CPID, and also with relevant PDI and Access Control information. 3. Ingest mechanism to validate and store the RepInfo, and assign a unique Persistent Identifier to it. 4. Access to RepInfo contained in negotiated DIP 5. Communication negotiation 6. Backup procedures 7. RepInfo capabilities
Provided Interfaces	• RegistryManager
Required Interfaces	• RegistrationManager • RepInfoGapManager
Artefacts	Current registry implementations: • JAXR

6.1.1 Registry Models

The Registry implements the following interface:

- RegistryManager

Interface	RegistryManager
Operations	- getRepInfo - getRegistrySearch - saveRepInfo - getClassificationScheme - provideRegservice
Used by	



Registry component diagram

6.2 KNOWLEDGE MANAGER

Component	Knowledge Manager
Responsibilities	1. Capture Higher level Semantics 2. Manage Designated Community Knowledge profile 3. Identify RepInfo Gaps 4. Manage Ontologies and Metadata
Parts	It comprises two layers A) **SWKM** (Semantic Web Knowledge Manager) which is the lower layer B) **CKM** (**CASPAR** Knowledge Manager) which is the upper layer SWKM (Semantic Web Knowledge Manager) will provide a set of core services for managing Semantic Web data. At the upper layer, CKM will provide high level services based on the OAIS model aiming at offering an abstraction useful for preservation information systems. CKM can be implemented using services offered by SWKM.
Provided Interfaces	(A) **CKM** • DCProfileManager • RepInfoGapManager • DescriptiveMetadataSWManager (B) **SWKM** Services: • Query () • Update () • Import() • Export() SWMainMemoryManagement • SWMainMemoryModel
Required Interfaces	
Artefacts	The first version of the implementation of the (B)-Services, as a set of web services (based on RDFSuite), has been done. The implementation of a proof-of-concept (A)-level based on the (B)-layer is ongoing.

6.2.1 Knowledge Manager Models

The Knowledge Manager, and in particular its part CKM, implements the above specified interfaces:

- DCProfileManager
- RepInfoGapManager

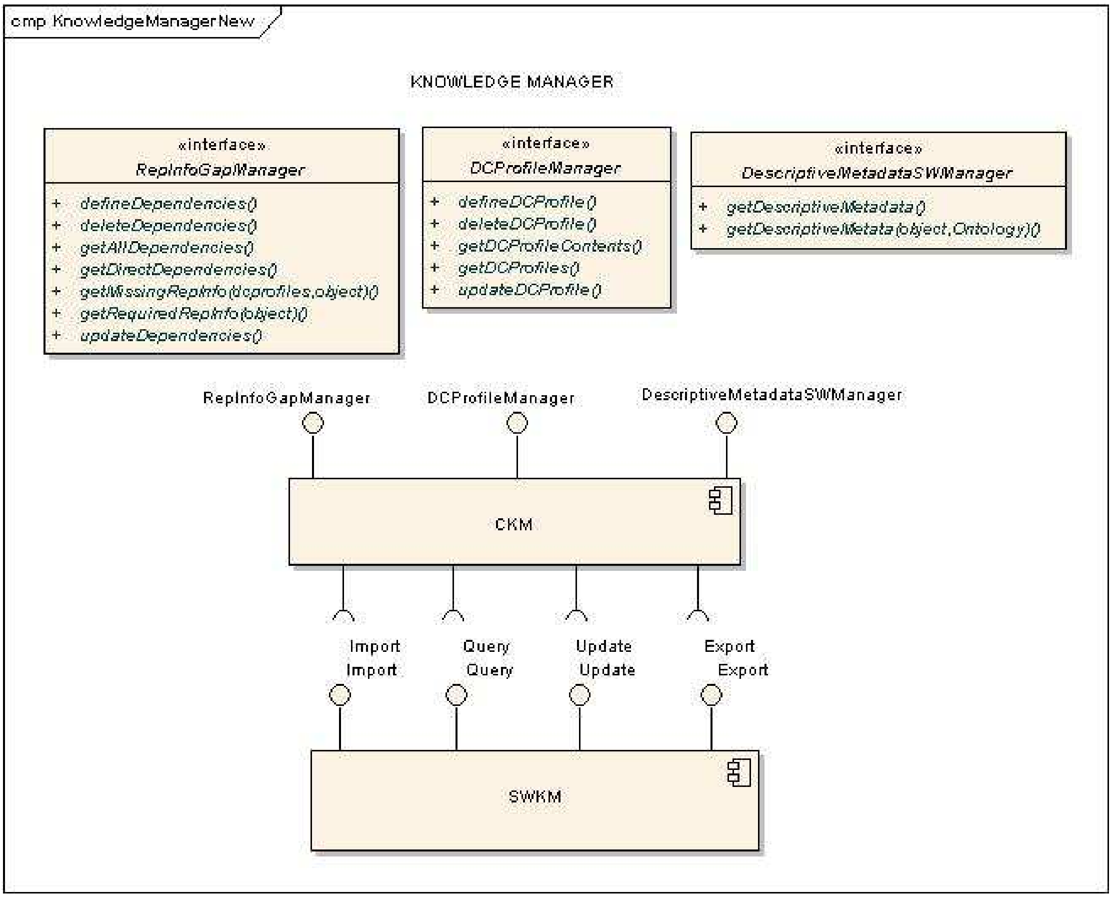
- DescriptiveMetadataSWManager

Interface	DCProfileManager
Operations	• defineDCProfile • updateDCProfile • deleteDCProfile • getDCProfiles • getDCProfileContent
Used by	• Data Access Manager and Security

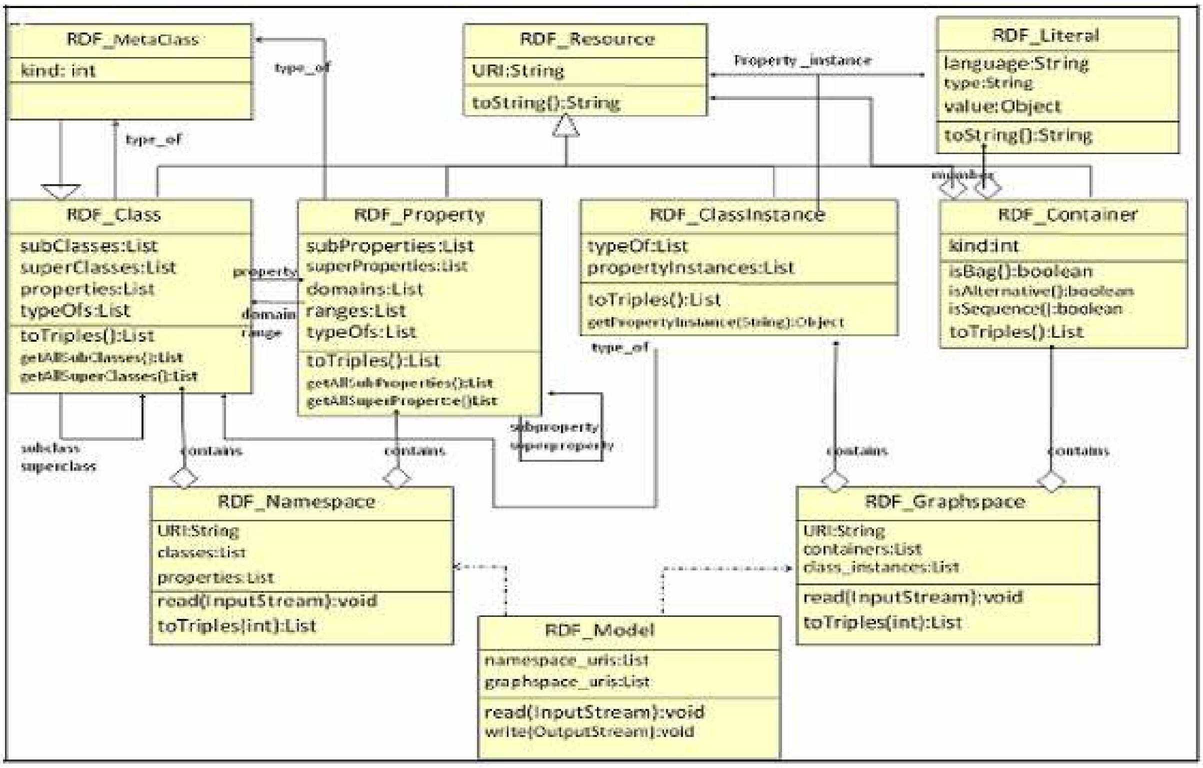
Interface	RepInfoGapManager
Operations	• defineDependencies • getDirectDependencies • getAllDependencies • updateDepenencies • deleteDependencies • getRequiredRepInfo(object) • getMissingRepInfo(dcprofiles,objects)
Used by	• Data Access Manager and Security • Registry • Preservation Orchestration Manager

Interface	DescriptiveMetadataSWManager
Operations	• getDescriptiveMetadata(object) • getDescriptiveMetadata(object, ontology)
Used by	• Finding Aids

Interface	SWKM Services
Operations	• query • update • import • export
Used by	• Data Access Manager and Security • Finding Aids



The **SWMainMemoryModel** is sketched at the following diagram. The detailed diagram is available at the EA repository.



6.3 PRESERVATION ORCHESTRATION MANAGER

Component	Preservation Orchestration Manager
Responsibilities	1. Manage Registration 2. Collect information about changes in Knowledge Base of Designated Community 3. Identify appropriate experts who are able to solve a RepInfo Gap 4. Notify experts about a RepInfo Gap
Parts	This modules is composed by the following 2 parts: • Notification Manager • Registration Manager
Provided Interfaces	• NotificationManager • RegistrationManager
Required Interfaces	• RepInfoGapManager • UserManager
Artefacts	

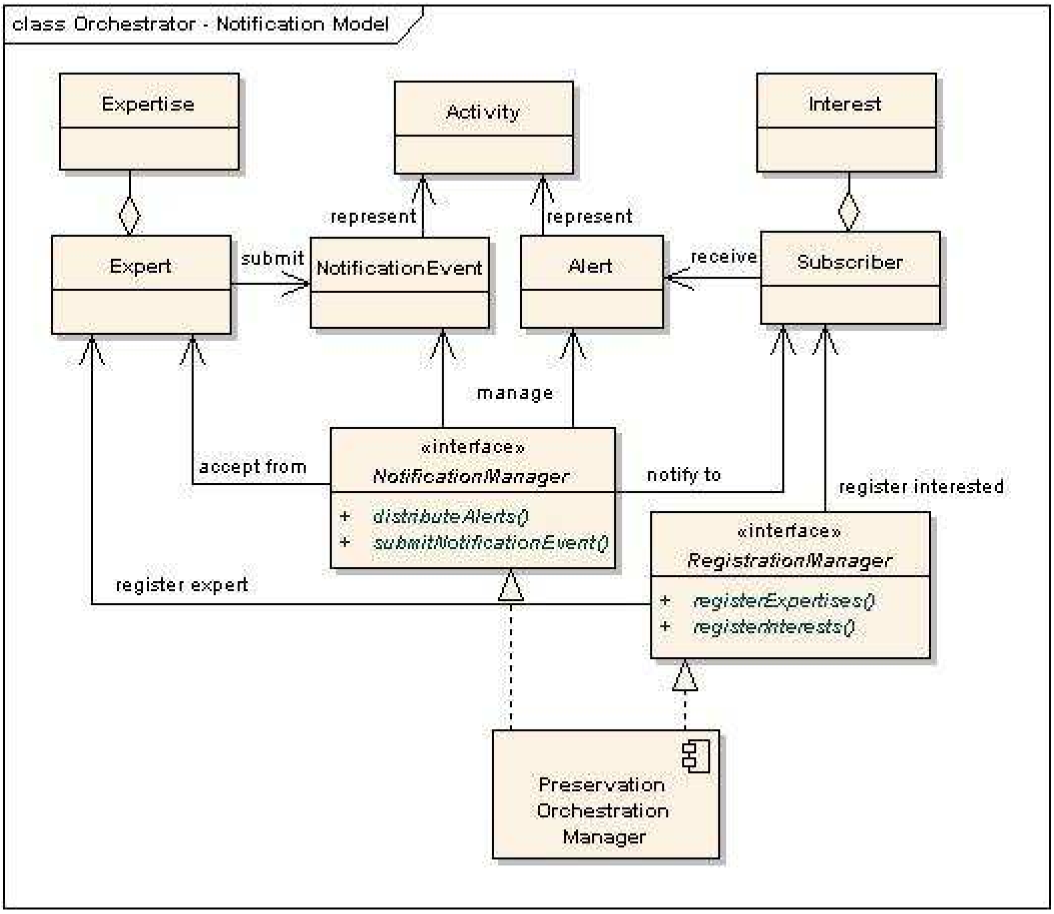
6.3.1 Preservation Orchestration Manager Models

The Preservation Orchestration Manager implements the following interface:

- NotificationManager
- RegistrationManager

Interface	NotificationManager
Operations	• submitNotificationEvent – submit a notification event • distributeAlert – distribute alerts to registered interested users
Used by	

Interface	RegistrationManager
Operations	• registerInterests - register interests of a user/system, in order to notify alerts • registerExpertises - register expertises of a user/system, in order to notify alerts
Used by	• Registry/Repository and RepInfo Gap Manager



Preservation Orchestration Manager Class Diagram

6.4 REPRESENTATION INFORMATION TOOLBOX

The Representation Information Toolbox, as specified in **CASPAR** Conceptual Model, addresses the need *"Representation Information - adequate for the Designated Community - must be created"*.

In order to do this it supports the user in the creation of Representation Information (RepInfo), by using a number of tools. These individual tools are not specified here but some are described in the *Review of the State of the Art* [**D1100**].

Representation Information includes syntactical and semantic description and also associated software and standards. The main function of the toolbox is as a portal, including a GUI.

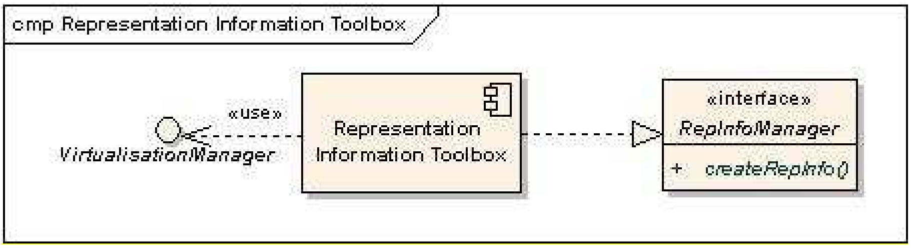
Component	Representation Information Toolbox
Responsibilities	1. Create adequate Representation Information, by helping users to use appropriate tools – acting as a portal to those tools. 2. Additional tools may be added by users
Parts	
Provided Interfaces	RepInfoManager
Required Interfaces	VirtualisationManager
Artefacts	

6.4.1 Representation Information Toolbox Models

The Representation Information Toolbox implements the following interface:

- RepInfoManager

Interface	RepInfoManager
Operations	• createRepInfo
Used by	User Interface


cmp Representation Information Toolbox
«use»
VirtualisationManager
Representation Information Toolbox
«interface»
RepInfoManager
+ createRepInfo()

Representation Information Toolbox Component Diagram

6.5 PRESERVATION DATASTORE

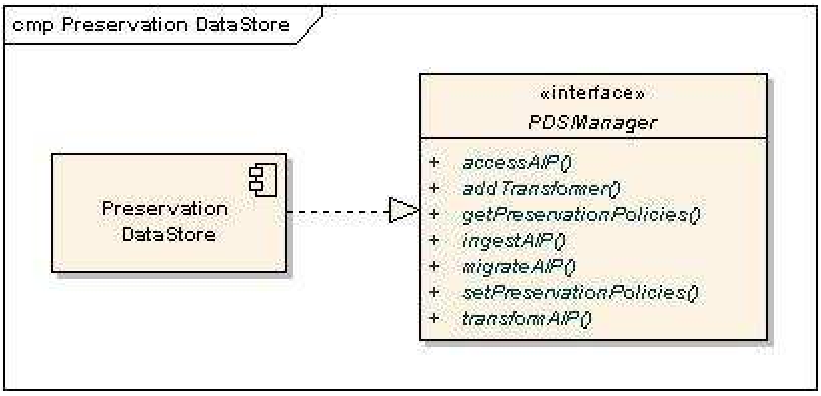
Component	Preservation DataStore
Responsibilities	Build an OAIS-based preservation aware storage with the following functionalities: 1. ingest AIP - persists the AIP into the OAIS for long term preservation 2. access AIP - retrieves an AIP stored previously within the OAIS 3. migrate AIP - migrates an AIP from an old system to a new system 4. transform AIP – transforms an AIP using a given transformer 5. add Transformer – adds a transformer to the system 6. get Preservation Policies – obtain the preservation policies which apply 7. set Preservation Policies – set the preservation policies which should apply
Parts	This module is composed by 3 layers: • OAIS Preservation Engine, including policy manager layer • XAM layer • Object layer
Provided Interfaces	• PDSManager
Required Interfaces	
Artefacts	PDS, OSD, optionally iRODS, optionally SRB

6.5.1 Preservation DataStore Models

The Preservation DataStore implements the following interface:

- PDSManager

Interface	PDSManager
Operations	• ingestAIP • accessAIP • migrateAIP • transformAIP • addTransformer • getPreservationPolicies • setPresevationPolicies
Used by	• Digital Rights Manager • Packaging



Preservation DataStore Component Diagram

6.6 DATA ACCESS MANAGER AND SECURITY

Component	Data Access Manager and Security
Responsibilities	1. manage Users / Groups / Roles / Permissions 2. manage authentication 3. manage access control to resources
Parts	This modules is composed by the following parts: • a User Management module, which allows to manage Users and Authorization • an Authentication module, which deals with user authentication • an Accounting module, which registers accounting events
Provided Interfaces	• UserManager • AuthenticationManager • AccountingManager
Required Interfaces	• DCProfileManager • RightsDistributionManager • RightsVerificationManager • RepInfoGapManager • RegistryManager • SWKMServices
Artefacts	

6.6.1 Data Access Manager and Security Models

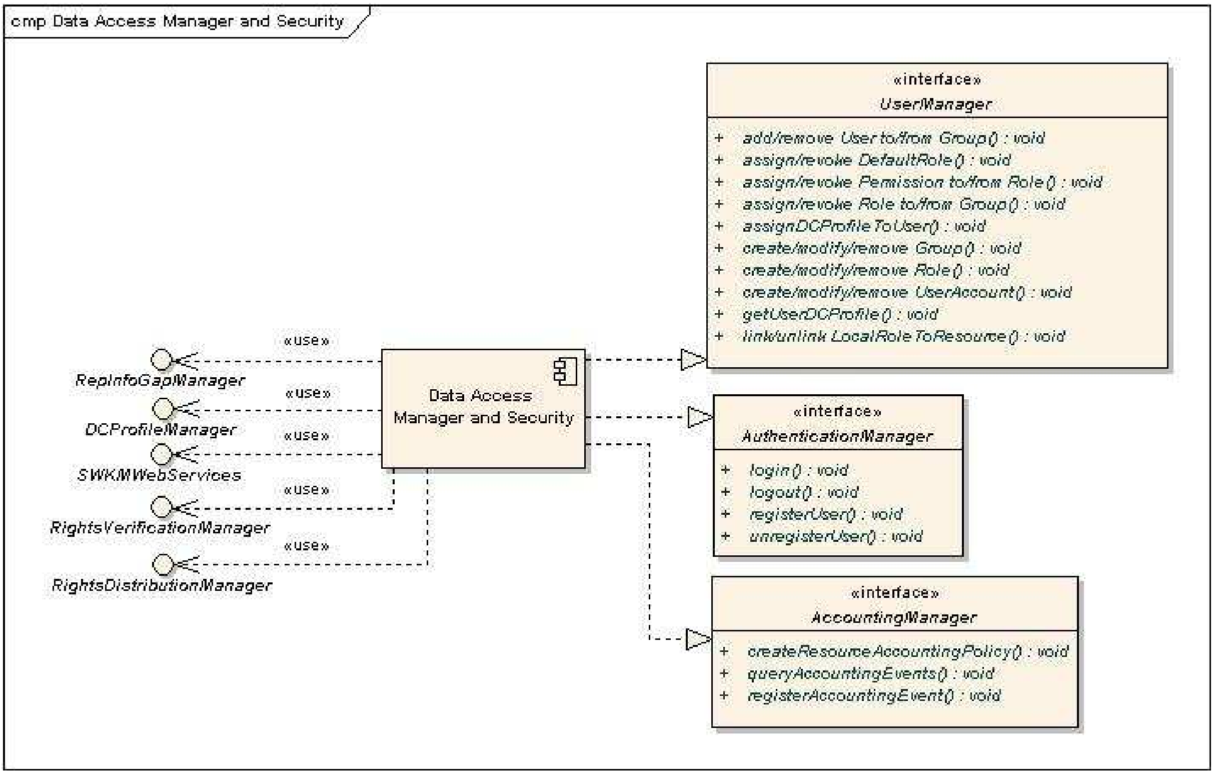
The Data Access Manager and Security implements the following interface:

- UserManager
- AuthenticationManager
- AccountingManager

Interface	UserManager
Operations	• [Add/Remove]User[To/From]Group • [Assign/Revoke]Permission[To/From]Role • [Assign/Revoke]Role[To/From]Group • [Assign/Revoke]DefaultRole • [Create/Modify/Remove]Role • [Create/Modify/Remove]Group • [Create/Modify/Remove]UserAccount • [Link/Unlink]LocalRoleToResource • AssignDCProfileToUser • getUserDCProfile
Used by	• Knowledge Manager

Interface	AuthenticationManager
Operations	• RegisterUser • UnregisterUser • Login • Logout
Used by	All user Interface

Interface	AccountingManager
Operations	• CreateResourceAccountingPolicy • RegisterAccountingEvent • QueryAccountingEvents
Used by	All user interfaces


cmp Data Access Manager and Security
«interface»
UserManager
+ add/remove User to/from Group() : void
+ assign/revoke DefaultRole() : void
+ assign/revoke Permission to/from Role() : void
+ assign/revoke Role to/from Group() : void
+ assignDCProfileToUser() : void
+ create/modify/remove Group() : void
+ create/modify/remove Role() : void
+ create/modify/remove UserAccount() : void
+ getUserDCProfile() : void
+ link/unlink LocalRoleToResource() : void
«use»
RepInfoGapManager
«use»
DCProfileManager
«use»
SWKMWebServices
«use»
RightsVerificationManager
«use»
RightsDistributionManager
Data Access Manager and Security
«interface»
AuthenticationManager
+ login() : void
+ logout() : void
+ registerUser() : void
+ unregisterUser() : void
«interface»
AccountingManager
+ createResourceAccountingPolicy() : void
+ queryAccountingEvents() : void
+ registerAccountingEvent() : void

Data Access Manager Component Diagram

6.7 DIGITAL RIGHTS MANAGER

Component	Digital Rights Manager
Responsibilities	1. definition of rights 2. distribution of rights (partly) 3. verification of rights 4. management and preservation of DRM
Parts	This module is composed by the following parts: • a Rights Definition module, which supports the definition of rights; it allows to define and associate rights to content items, and to create and manage license offers documents, templates, and license instances • a module that acts as Rights Distribution Gateway, which guides the Consumer in the process of obtaining the necessary permissions for content usage. Some parts of the process may be performed externally to the **CASPAR** system architecture, i.e. the payment of fees. Distribution of rights includes applying rights enforcement measures to content items • a Rights Verification module that is involved before allowing a specific action on a rights-protected content, e.g. to validate licenses and to resolve persistent protection • a module for DRM Management and Preservation, which allows to do maintenance on the other modules in order to align the DRM functionalities with the evolutions in technology and in the semantic aspects of rights, e.g. by registering new IPMP tools, updating rights definitions and license semantics.
Provided Interfaces	• RightsDefinitionManager • RightsDistributionManager • RightsVerificationManager • DRMPreservationManager
Required Interfaces	• PDSManager
Artefacts	REL(s)

6.7.1 Digital Rights Manager Models

The Digital Rights Manager implements the following interfaces:

- RightsDefinitionManager
- RightsDistributionManager
- RightsVerificationManager
- DRMPreservationManager

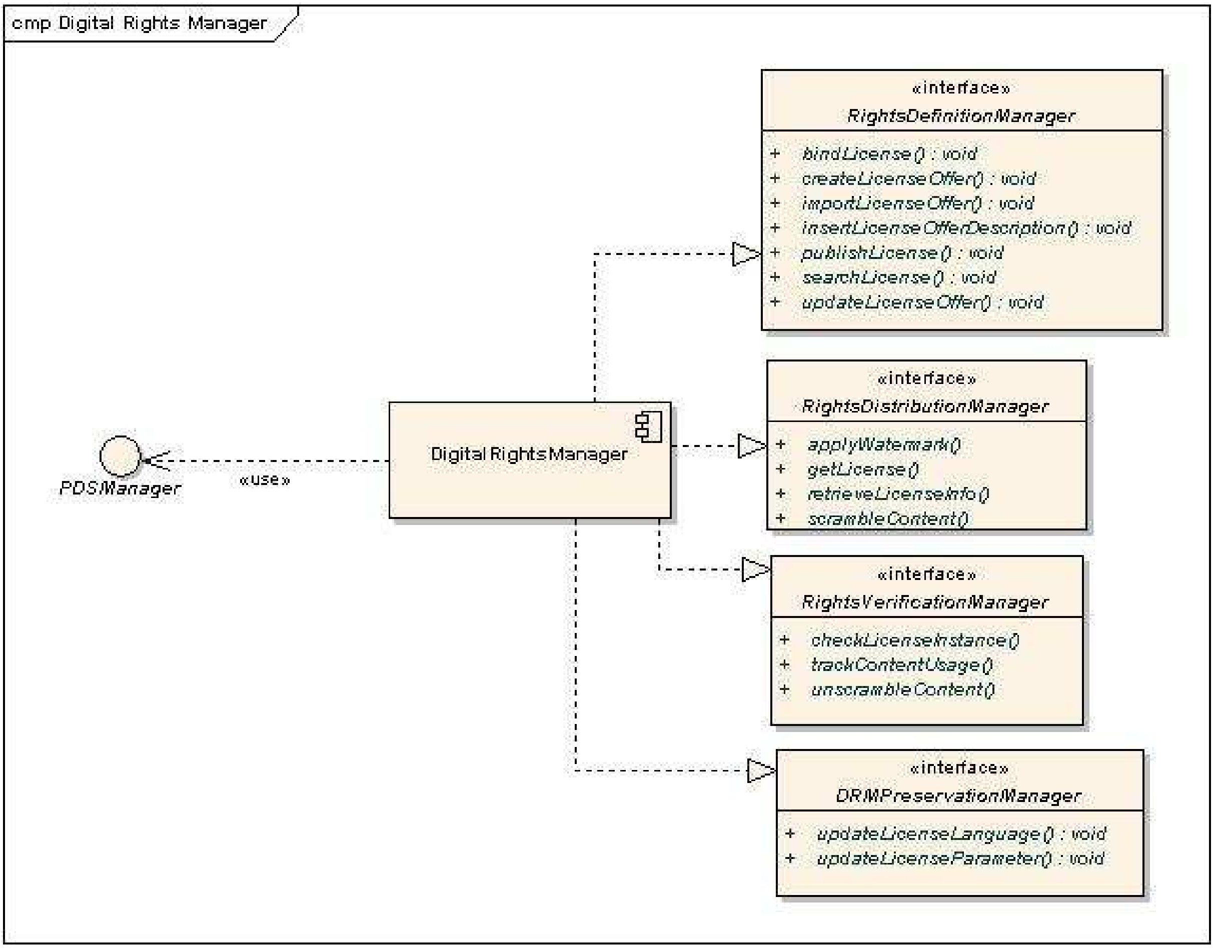
Interface	RightsDefinitionManager
Operations	• CreateLicenseOffer • UpdateLicenseOffer

	• ImportLicenseOffer • PublishLicense • SearchLicense • BindLicense • InsertLicenseOfferDescription
Used by	Rights enforcement tools external to the archival platform

Interface	RightsDistributionManager
Operations	• RetrieveLicenseInfo • GetLicense • ApplyWatermark • ScrambleContent
Used by	• Data Access Manager and Security • User Interface

Interface	RightsVerificationManager
Operations	• CheckLicenseInstance • UnscrambleContent • TrackContentUsage
Used by	Data Access Manager and Security

Interface	DRMPreservationManager
Operations	• UpdateLicenseParameter • UpdateLicenseLanguage
Used by	User Interface (DRM manager - OAIS Archive internal staff member)



Digital Rights Manager Component Diagram

6.8 FINDING AIDS

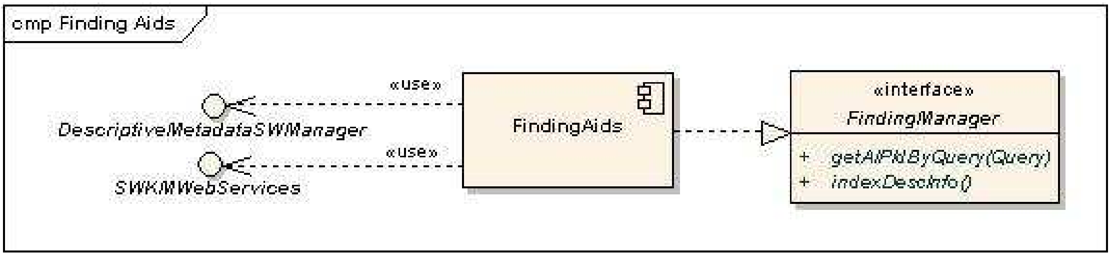
Component	Finding Aids
Responsibilities	Search AIPs, based on conditions upon the Descriptive Information. The main responsibility of the Finding Aids module is to function as the "link" between the end-user (consumer or digital archive) and the rest of the **CASPAR** system, with respect to the search and retrieval facilities.
Parts	A search AIP facility allowing the determination and retrieval of a set of AIPs whose Descriptive Information satisfies a set of conditions (query). This facility is based on the query mechanisms provided and implemented by the KM and is comprised of the following subparts: • An interface (to be used by the end-user) that allows the formulation of queries (in a specific Query Language) and the return of query results in a user-friendly way. • An interface (in the form of an API) that allows an artificial agent to formulate queries and get the results back. • A module to transform queries that were input through the interfaces into a specific Query Language syntax. • A module that reads ontology schemas and allows the dynamic creation of the relevant user interfaces.
Provided Interfaces	• FindingManager
Required Interfaces	• DescriptiveMetadataSWManager • SWKMServices
Artefacts	

6.8.1 Finding Aids Models

The Finding Aids implements the following interface:

- FindingManager

Interface	FindingManager
Operations	• getAipIdByQuery - Gets a query in its input; returns a list of AIP IDs -- to be used by external artificial agents, e.g., archives' software -- this function performs some pre-processing on the query, generates the equivalent query in a specific Query Language and forwards it to the KM for execution; then returns the results (AIP IDs) to the caller • indexDescInfo – index the Descriptive Information, which is used to support the search
Used by	• Packaging



Finding Aids Component Diagram

6.9 VIRTUALISATION TOOLBOX

This is part of the RepInfo Toolbox but the concept is sufficiently generic to warrant a dedicated, if brief, description.

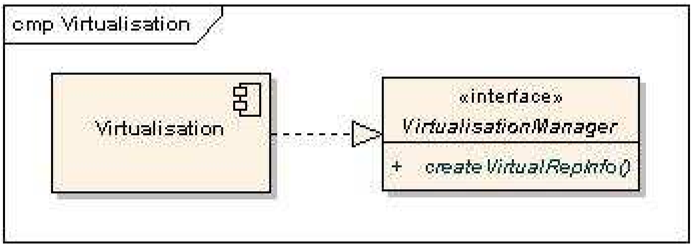
Component	Virtualisation
Responsibilities	Helps the user to identify and describe features in a piece of information (and to a lesser extent also in software and hardware) which will facilitate automated re-use and easier re-implementation.
Parts	This is a portal into a number of specific tools which deal with a variety of aspects of virtualisation, as outlined in the Use Cases.
Provided Interfaces	• VirtualisationManager
Required Interfaces	
Artefacts	This tool is a portal into many other tools, some of which are briefly described in the Review of the State of the Art.

6.9.1 Virtualisation Toolbox Models

The Virtualisation Toolbox implements the following interfaces:

- VirtualisationManager

Interface	VirtualisationManager
Operations	• createVirtualRepInfo
Used by	• Representation Information Toolbox



Virtualisation Toolbox Component Diagram

6.10 PACKAGING

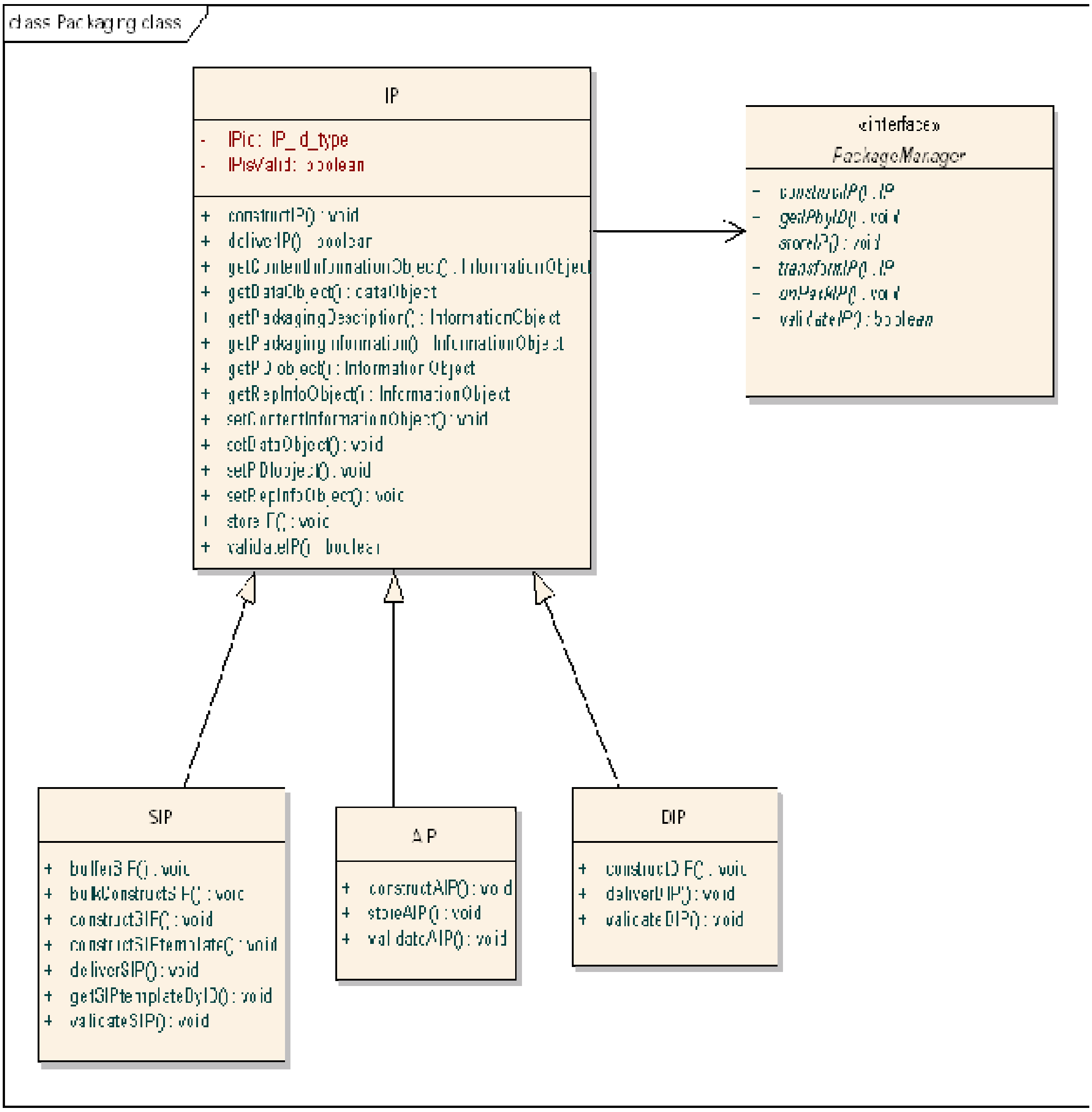
Component	Packaging
Responsibilities	1. Construction of Information Packages (IP) 2. Unpackaging of IP 3. Access Package Information Objects 4. Manipulation of Package Information Objects 5. Validation of IP 6. Delivery of IP
Parts	This module is composed of the following parts: • IP(Information Package) superType • SIP, AIP, DIP subtypes of IP
Provided Interfaces	PackageManager
Required Interfaces	• FindingManager • SWKMWebServices (import)
Artefacts	XFDU, SAFE, METS tool kits.

6.10.1 Packaging Models

The Packaging implements the following interfaces:

- PackageManager

Interface	PackageManager
Operations	• constructIP • getIPbyID • storeIP • transformIP • unpackIP • validateIP
Used by	



Packaging Component Diagram

6.11 AUTHENTICITY MANAGEMENT

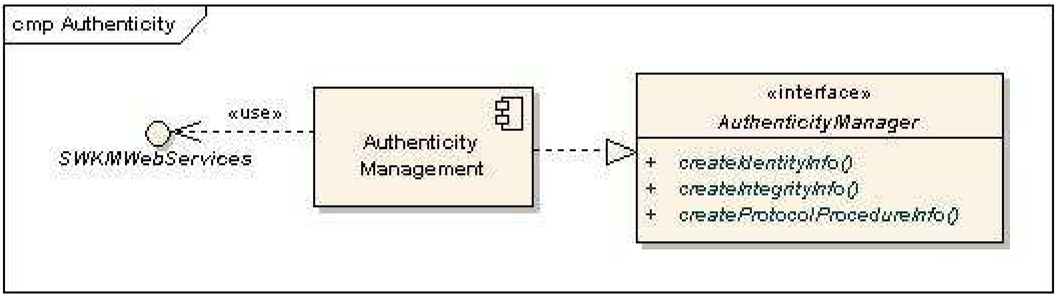
Component	Authenticity Management
Responsibilities	• Create identity information • Create integrity information • Create protocol and procedure information
Parts	
Provided Interfaces	AuthenticityManager
Required Interfaces	SWKMServices (query)
Artefacts	

6.11.1 Authenticity Management Models

The Authenticity Management implements the following interface:

- AuthenticityManager

Interface	AuthenticityManager
Operations	• createIntegrityInfo • createIdentityInfo • createProtocolProcedureInfo
Used by	



Authenticity Management Component Diagram

7 DEPLOYMENT

In the previous sections we described the **CASPAR** architectural entities, i.e. the various components, together with the respective interfaces. Here we concentrate on the way in which these components may be deployed.

CASPAR components are designed to be loosely coupled, avoid a central point of failure, be customisable, and must not demand a complete system replacement in order to be of advantage to adopters.

The following scenarios are illustrative of a broad spectrum of deployments.

Note that in order to work with existing elements in an archive, a number of relatively small pieces of software "glue" will have to be provided. Neither these nor the non-**CASPAR** components are shown in these diagrams.

7.1 THREE MAIN SCENARIOS

CASPAR is a preservation environment capable of dealing with all aspects of the digital preservation and to provide typical archive facilities, although, as noted in the Conceptual Model some components, such as INGEST and ACCESS will have limited functionality. The system is built, following the OAIS Reference Model, to cover the long-term preservation of digitally encoded information. This includes the long-term applicability of, for example, Intellectual Properties Rights (IPR), as well.

Three broad scenarios are:

- Self-contained installation
- Shared service installation
- Minimal shared service installation

7.1.1 The Self-Contained installation

In this scenario all **CASPAR** components are installed within a single organisation.

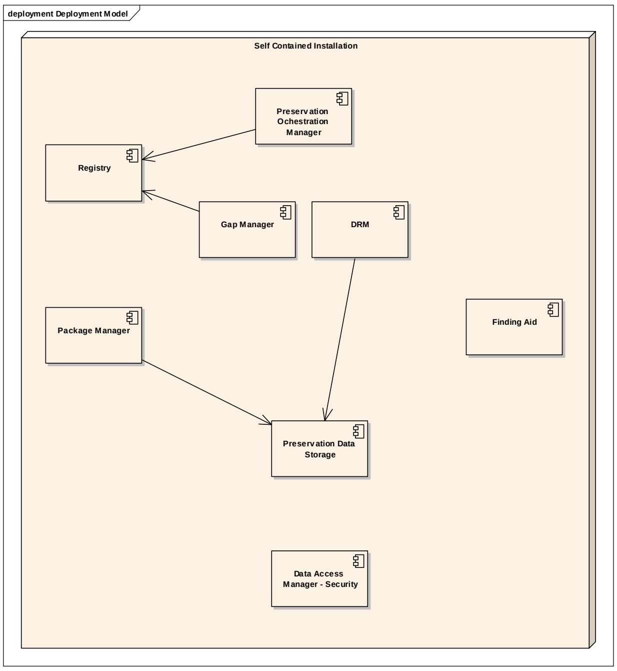

deployment Deployment Model
Self Contained Installation
Preservation Ochestration Manager
Registry
Gap Manager
DRM
Package Manager
Finding Aid
Preservation Data Storage
Data Access Manager - Security

Figure 16 Self-contained installation

One of the key aims of the **CASPAR** infrastructure is to spread the effort needed for preservation. Therefore although all components could reside within one closed system, this will somewhat defeat the aims of **CASPAR** but could share expertise within that organisation.

7.1.2 The Shared Service installation

In this scenario, which allows a sharing of the effort of preservation, the installation would omit one or more of :

- the Preservation Orchestration Manager,
- the RepInfo Gap Manager,
- the Registry

These components would be hosted on one or more other hosts and are shared between many shared service installations. The resource discovery mechanism will be discussed below.

Some selection of

- Preservation Data Store
- Access components
- Rights Management

May be installed and integrated with the existing systems in the archive. In this common scenario the archive has already invested effort and funding to provide archival services to its end users (both Producers and Consumers), and has no interest in replacing all its systems.

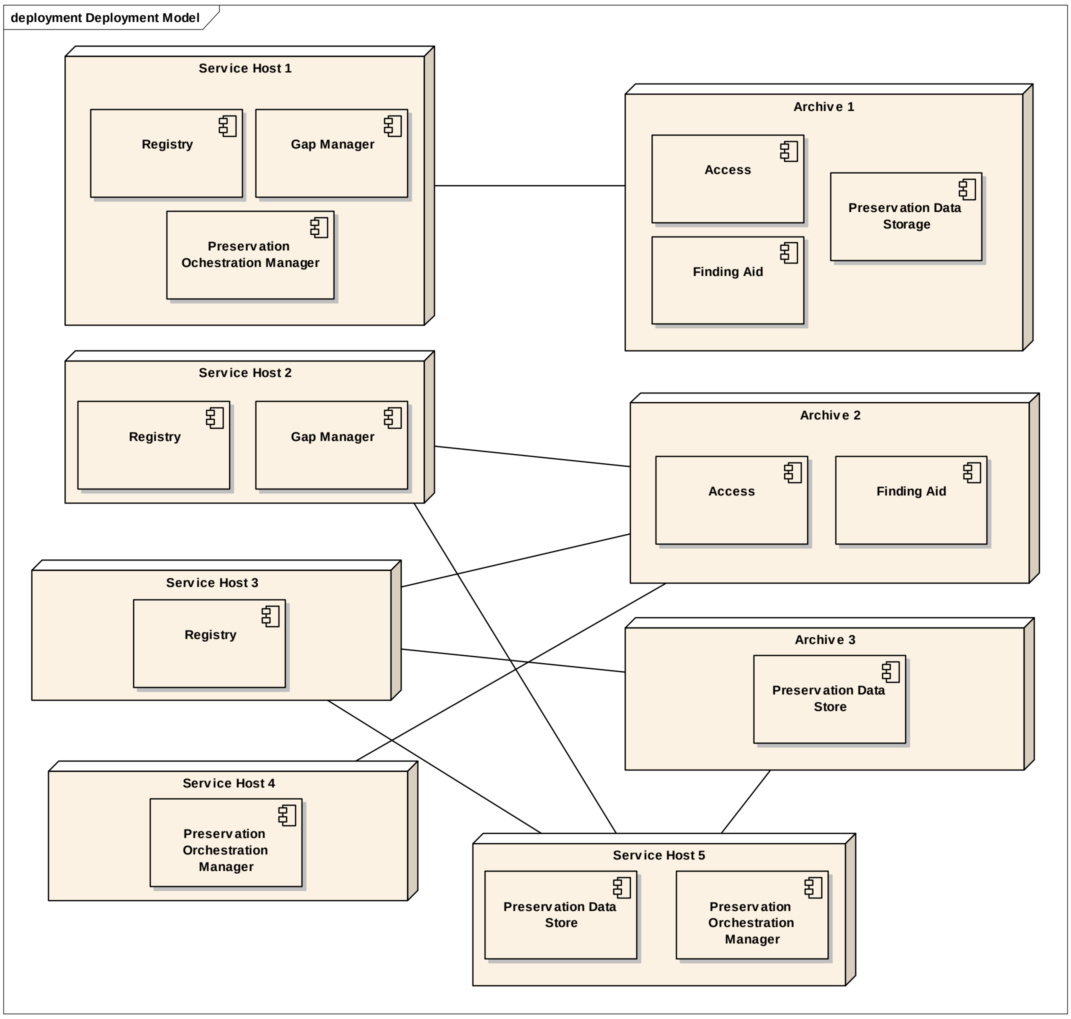


Figure 17 Shared service installation

Archive 1, Archive 2 and Archive 3 contain one or more of the **CASPAR** components, and share the services from:

Service Hosts 1 to 4 containing one or more of the shared infrastructure components.

In Service Host 5 explicitly shows that such a host may contain other **CASPAR** components, and indeed such a host would have to be a Trusted Archive, something with which adoption of **CASPAR** components would assist.

7.1.3 The Minimal installation

A minimal scenario, which is an extreme type of shared service installation, involves only the adoption of the **CASPAR** convention for Representation Information, in particular associating data objects with pointers to Representation Information contained in external registries. This minimal installation is a low cost buy-in to **CASPAR**; it involves minimal change to existing systems, but still allows sharing of the preservation effort.

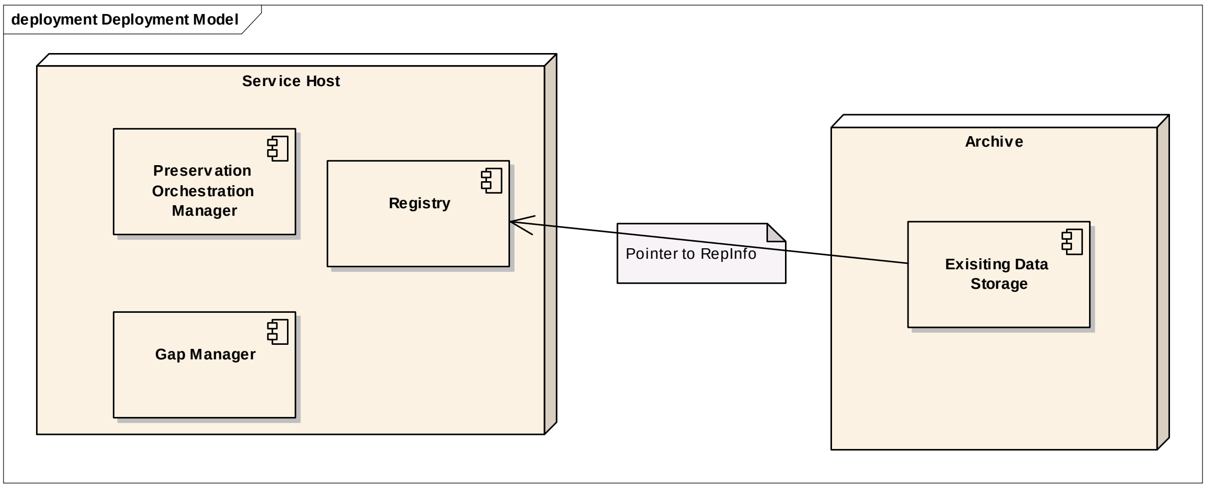


Figure 18 Minimal shared service installation

These deployment scenarios and resource discovery mechanisms are discussed next.

7.2 RESOURCE DISCOVERY MECHANISMS

The **CASPAR** components installed in any of the above scenarios must be configured so that each one knows how to address the others. The discovery mechanisms

- Cannot rely on a continuation of **CASPAR**
- Must be flexible, supporting the types of scenarios described above
- Must be configurable – allowing archives to tailor their service providers, for example depending upon, for example, which services they trust or which services they are allowed to share information with.
- Need not be all encompassing i.e. does not need to guarantee that all services are known about by all others

As with other distributed systems there are a number of discovery mechanisms:

- Local Configuration file
  - This would be needed as an initial "boot-strap" step for the other discovery methods by pointing to one or more discovery nodes.
  - It could also contain a fixed set of pointers to selected service providers
- Discovery nodes
  - These would contain pointers to service providers – note that it does not point to other non-service provider archives.
  - Service providers would publish their services to one or more discovery nodes (named in their configuration files)

Any of the service providers could also be a discovery node.

8 IMPLEMENTATION GUIDELINES

CASPAR implementations must themselves be preservable – by applying the **CASPAR** techniques themselves, for example facilitating the software against changes in hardware and software environments via Virtualisation, as indicated in Conceptual Model.

Multi-lingual support will be supported wherever possible – itself a type of Virtualisation.

In addition to the Virtualisation itself, for example by defining software interfaces, it is important to provide proof of effectiveness by implementing the interfaces on at least two underlying implementations.

8.1 RE-USE OF EXISTING SOFTWARE

The Storage interfaces may be implemented on a number of underlying systems as illustrated in Figure 24 in which the underlying implementations may range from simple file systems to the iRODS system – itself an interface to a distributed heterogeneous storage system, shown here using an Object Data Store using its XAM interface – and also to a XAM device directly.

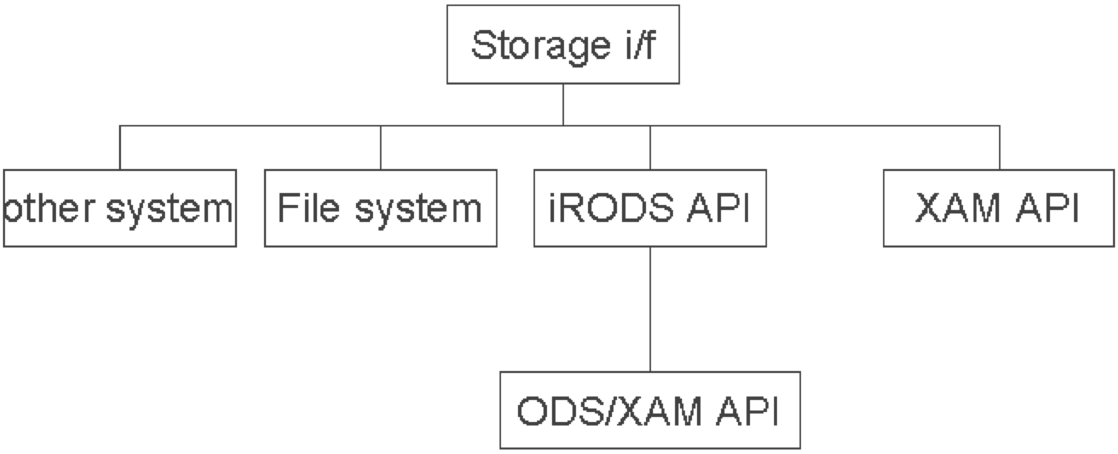


Figure 19 Storage Interface

The Registry functionality may be implemented as in Figure 20 which shows the Registry API implemented on systems including JAXR, which is itself an abstraction which can use emXML as well as UDDI registries as underlying implementations.

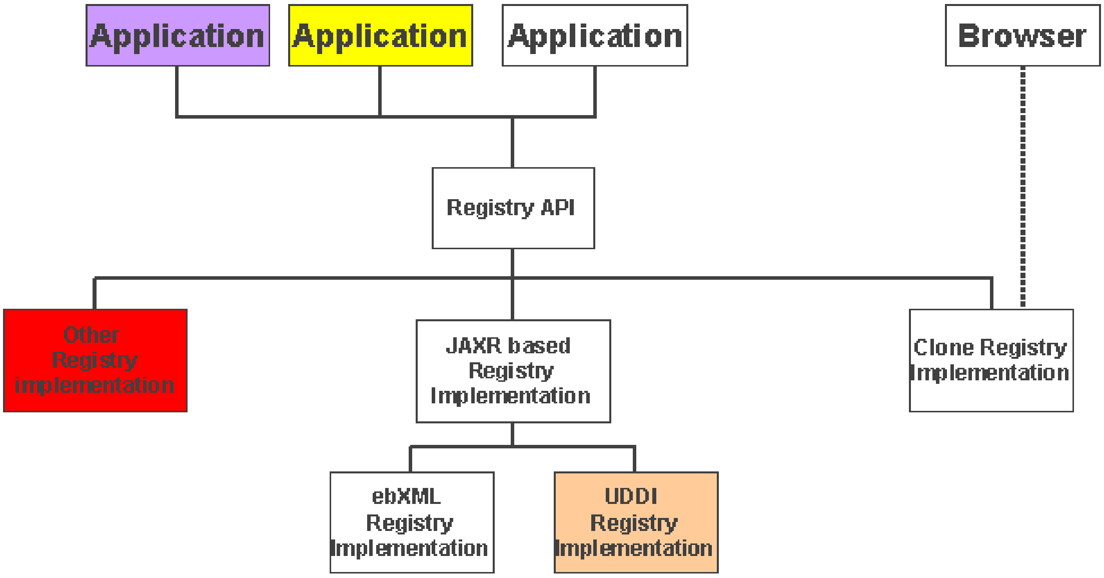

Application
Application
Application
Browser
Registry API
Other Registry implementation
JAXR based Registry Implementation
Clone Registry Implementation
ebXML Registry Implementation
UDDI Registry Implementation

Figure 20 Registry Interface

Some of the communications middleware may also be implemented on several underlying systems as in Figure 21, where the BRICKS and DILIGENT underlying systems are used. DILIGENT is itself layered on top of various GRID Web Services.

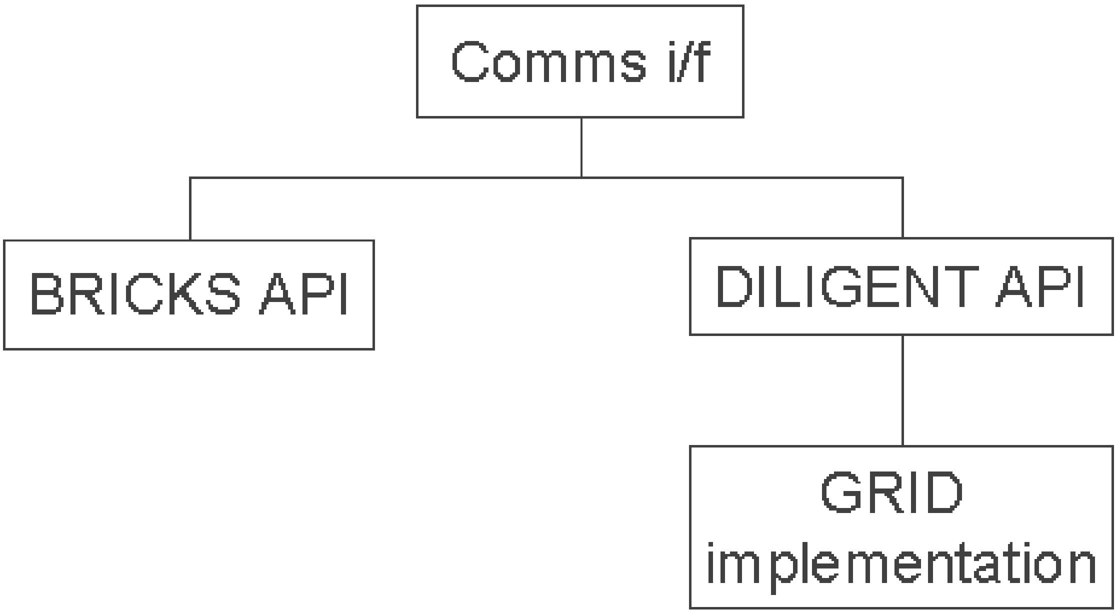


Figure 21 Communication Interface

The Knowledge Management tools should be based on a Knowledge management interface which can be implemented on a number of existing libraries:

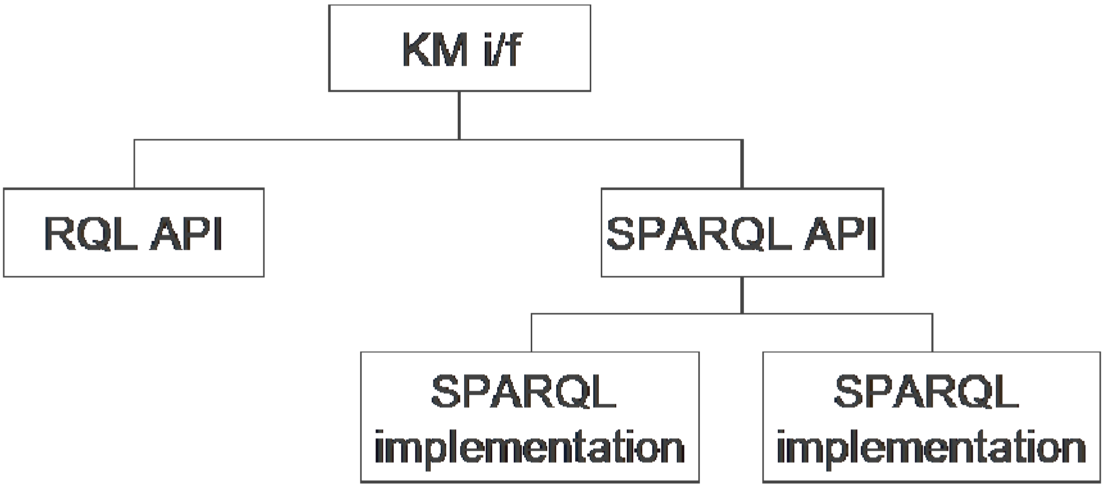


Figure 22 Knowledge Management Interface

8.2 IMPLEMENTATION PROGRAMMING LANGUAGE AND PROTOCOLS

The initial implementation of essentially all the components will be in JAVA, using Web Services for distributed computing aspects.

Other implementations may be produced for specific components. Furthermore the **CASPAR** components will be integrated with a variety of existing systems, implemented on many different systems. All this will provide some evidence about the preservability of the **CASPAR** functionality.

As noted in section 1, the use of UML as the modelling language allows us, if needed to produce a significant part of the software in other programming languages. There are limits to this capability, because one tends not to use the language independent aspects of UML to the fullest extent, and the UML tool tends to be used to create a skeleton into which code defining the processes and algorithms are inserted. Nevertheless it provides an important degree of flexibility.

Furthermore the interfaces which are defined may be implemented in such a way as to support a variety of distributed communication frameworks, such as Web Services, JAVA-RMI and CORBA.

8.2.1 Discussion of Java implementation

To attempt to program the **CASPAR** environment for the long term it is necessary to take a fairly high level view of the software operations required and then to encapsulate any particular implementation within that framework. Where functions are limited to a particular subset of possible implementations then the **CASPAR** system should be able to identify the functions that can or cannot be performed. The software system is going to be distributed over a number of disjoint, heterogeneous computer systems and this means we need a software system that is widely available and with a rich library of networking software. There will also need to be checks to identify and as necessary authenticate users in their access to the system. To make this system accessible to the users we will develop a number of graphical user interfaces, so a common library across all the platforms is required, and as out users will in general speak different languages we will internationalization of the interface.

Fortunately Java meets these requirements; its concept of a virtual machine on which to run a program matches nicely with the concepts concerning preservation already in the project and it is available on a wide range of computer systems. The language comes with as standard both a rich networking library and a rich graphical user interface system. The concept in this of layout managers handling both simple and complex objects in the interface is also one with echoes elsewhere in the project. Sun Microsystems release of Java SE, Java EE and Java ME implementations under the GNU General Public License (GPLv2), the same license as is used for Linux, is a very significant step in the long term preservation of systems built on top of Java. In developing responsive graphic interfaces across different hardware systems is challenging; the multi-threading model built into the Java language can help to make this more practical; however this has to be well understood if the action of the interface is to be preserved in the long term.

The Java serialization system can make for a good communication model between active processes [short term serialization], but needs effective Representation Information if the object store is to be maintained long term. The default protection is to fail if a class has changed; this is a step up of doing some random operation in that a least the user is aware of the problem.

An additional feature of the graphics interface library, the “swing” library, is its support for people who have some difficulty utilizing the standard user interface, providing this information has benefit in the long term since it provides extra context to the use of the system that may help in the understanding of the software in generations to come.

8.3 ITERATIVE IMPLEMENTATION PHASES

The implementation will, as described in the **CASPAR** Guidelines [**D1202**] and following software best practice, be implemented in a number of iterations. Each iteration will last approximately 4 to 6 months and will deliver an incremental level of functionality.

The initial phases are outlined in the following table, where the key components are shown, but of course a number of smaller pieces of supporting “glue” will be being developed at the same time.

Phase	Registry	Knowledge Mgt	Pres. Data Store	RepInfo (incl. Virtualisation) Tools	DRM and DAMS	Finding Aids	Packaging	Orchestration	Authenticity
1	Initial implementation with RepInfoLabel and initial load of RepInfo. This is being done in association with the UK Digital Curation Centre (DCC)	Initial implementation of the Gap Manager	Implementation on Object Storage Device and XAM interface	Low level RepInfo for Scientific data and initial Virtualisation for simple objects	Prototype DAMS system	Prototype Finding Aids based on Provenance	XFDU and METS based packaging tools	Simple notification and alert system	Prototype authenticity protocols and more detailed design
2	Trial federation capabilities	Interaction with Orchestration and Registry	Prototype policy interface	Integration of tools to view RepInfo and RepInfo creation for Cultural Heritage info	Integration of DAMS with Registry and Gap Manager	Interface to Data Management implementation (not a key component)	Packaging of Cultural Heritage data and integration with existing Performing Arts packaging	Integration with DAMS and installation of Classification schemes for registration of interests	Implementation of basic Authenticity protocols
3	Release 2 and interface to a number of other registry systems	Release 2 of Gap Manager using SPARQL and expansion of DCProfiles	Implementation of interfaces on simple file systems and simple existing archive data	Tools for capture of semantic information. Initial “expert system” for	Completion of DAMS Release 1. Prototype DRM s/w	Release 1 of Finding Aids and integration with one or more existing archives	Integration with Pres. Data Store operations	Use of simple Knowledge Management techniques for routing alerts	Release 1 of Authenticity tools

			stores.	providing suggestions					
4	Release 3 of registry with installation kit and interoperability framework	Release 3 of Gap manager with terminology evolution	Release 3 of Pre. Data Store with test implementation with 10 million objects	Virtualisation tools and guide integrated.	Integration of preservation capabilities into DRM and DAMS	Intelligent Finding Aids using PDI, Release 2	Integrate Packaging for DIP with Finding Aids and Access.	Full alert system release	Release 2 of Authenticity tools integrated with PDI production and Packaging

9 REFERENCES

Here are listed all references used during the preparation of this deliverable. More, some interesting readings are signalised (interest is evaluated with stars ★: from 5 [high interest] to 1 [low interest]).

★★★★★

Id	DesigningJ2EE
Author(s)	Inderjeet Singh, Beth Stearns
Title	Designing Enterprise Applications with J2EE Platform, 2nd Ed.
Keywords	J2EE Web Architecture
Reference	http://java.sun.com

Id	Sommerville
Author(s)	Ian Sommerville
Title	Software Engineering
Keywords	Software Engineering
Reference	

★★★★

Id	Y.Tzitzikas'2007
Author(s)	Yannis Tzitzikas
Title	Dependency Management for the Preservation of Digital Information
Keywords	Digital Information Preservation; Representation Information Gap; Knowledge Management
Reference	18th International Conference on Database and Expert Systems Applications, DEXA'2007, Regensburg, Germany, September 2007

Id	CIDOCCRM
Author(s)	ICOM/CIDOC
Title	CIDOC CRM - Introduction to object oriented modelling
Keywords	CIDOC CRM; OO Modelling
Reference	http://cidoc.ics.forth.gr/docs/introduction2oo.htm

Id	MLA
Author(s)	MLA and EERAC
Title	Report of the East of England Digital Preservation Regional Pilot Project
Keywords	Preservation; OAIS; Cost Models; Business Case
Reference	http://www.data-archive.ac.uk/news/publications/darp2006.pdf

Id	Mitcham
Author(s)	Jen Mitcham
Title	Working with OAIS
Keywords	Preservation; OAIS; Archaeology; Guideline; Architecture
Reference	http://www.dpconline.org/docs/events/060404jlm-dpc-oasis.pdf

Id	Moore02
Author(s)	Reagan W. Moore
Title	The Preservation of Data, Information and Knowledge
Keywords	Preservation Knowledge
Reference	http://www.sdsc.edu/NARA/Publications/Web/moore-rw.doc

Id	PreservationParadox
Author(s)	Su-Shing Chen
Title	The Paradox of Digital Preservation
Keywords	Preservation
Reference	http://www.gseis.ucla.edu/us-interpares/pdf/ParadoxOfDigitalPreservation.pdf

★★★

Id	Rosental05
Author(s)	D.S.H.Rosenthal, T.Robertson, T.Lipkis, V.Reich, S.Morabito
Title	Requirements for Digital Preservation Systems: A Bottom-Up Approach
Keywords	OAIS; LOCKSS; Requirements
Reference	http://arxiv.org/PS_cache/cs/pdf/0509/0509018v2.pdf

Id	IEEE00
Author(s)	IEEE Recommended Practice for Architectural Description of Software-Intensive Systems
Title	Architectural description
Keywords	Architecture Description
Reference	http://ieeexplore.ieee.org/iel5/7040/18957/00875998.pdf

Id	EUP
Author(s)	Scott W. Ambler
Title	An Overview of the Enterprise Unified Process (EUP)
Keywords	Enterprise Unified Process
Reference	http://www.enterpriseunifiedprocess.info/downloads/eupHandout.pdf

Id	ArchiMate2002
Author(s)	Arbab F., Bonsangue M., Scholten J.G., Iacob M.-E., Jonkers H., Lankhorst M., Proper E., Stam A
Title	State of the Art in Architecture Frameworks and Tools
Keywords	Architecture Frameworks
Link	http://www3new.telin.nl/index.cfm?type=doc&handle=22327&language=en

Id	MITRE2004
Author(s)	
Title	Guide to the Enterprise Architecture Body of Knowledge
Keywords	Enterprise Architecture
Link	http://www.mitre.org/tech/eabok/index.html

Id	ARCHMOD2006
Author(s)	Scott W.Ambler
Title	Architecture and Architecture Modeling Techniques
Keywords	Architecture Modelling
Link	http://www.agiledata.org/essays/enterpriseArchitectureTechniques.html

Id	AGARCH2006
Author(s)	Scott W.Ambler
Title	Agile Enterprise Architecture
Keywords	Agile Enterprise Architecture
Link	http://www.agiledata.org/essays/enterpriseArchitecture.html

Id	FederalEA
Author(s)	Federal Chief Information Officer Council
Title	A Practical Guide to Federal Enterprise Architecture
Keywords	Enterprise Architecture
Link	http://www.cio.gov/archive/bpeaguide.pdf

★★

Id	EAWhitePapers
Author(s)	Enterprise Architecture Presentations and White Papers
Title	
Keywords	Enterprise Architecture
Link	http://www.pearsonblueprint.com/

Id	EnterpriseArchitect
Author(s)	Sparx Systems
Title	Deployment of Enterprise Architect
Keywords	Enterprise Architect
Link	http://www.sparxsystems.com/downloads/whitepapers/EA_Deployment.pdf

Id	VersionControl
Author(s)	Sparx Systems
Title	Setting up Version Control
Keywords	Enterprise Architect ; Version Control
Link	http://www.sparxsystems.com/WhitePapers/Version_Control.pdf

GLOSSARY

This glossary is largely taken from the OAIS Reference Model.

A

Access Aid	A software program or document that allow Consumers to locate, analyze, and order Archival Information Packages of interest.
Access Collection	A collection of AIPs that is defined by a Collection Description but for which there is no Packaging Information for the collection in Archival Storage.
Access Method	A method for retrieving an Archival Information Package based on its name or identifier, which is available to authorized users.
Access Software	A type of software that presents part of or all of the information content of an Information Object in forms understandable to humans or systems.
Access	The CASPAR entity that contains the services and functions which make the archival information holdings and related services visible to Consumers.
Ad hoc Order	A request that is generated by a Consumer for information CASPAR has indicated is currently available.
Administration	The CASPAR entity that contains the services and functions needed to control the operation of the other CASPAR functional entities on a day-to-day basis.
Archival Information Collection (AIC)	An Archival Information Package whose Content Information is an aggregation of other Archival Information Packages.
Archival Information Package (AIP)	An Information Package, consisting of the Content Information and the associated Preservation Description Information (PDI), which is preserved within CASPAR.
Archival Information Unit (AIU)	An Archival Information Package whose Content Information is not further broken down into other Content Information components, each of which has its own complete Preservation Description Information. It can be viewed as an "atomic" AIP. An example of an AIU would be a table of numbers representing temperatures in a certain region with all the associated documentation describing how and where the temperatures were measured, what instruments were used to make the measurements, who made the measurements, why they were made, what processing has been performed on the measurements and who has had custody of these measurements since they were first created, how the measurements relate to other information, how the measurements can be uniquely referenced by others, etc.
Archival Storage	The CASPAR entity that contains the services and functions used for the storage and retrieval of Archival Information Packages.
Archive	An organization that intends to preserve information for access and use by a Designated Community.
Associated Description	The information describing the content of an Information Package from the point of view of a particular Access Aid.

C

Client	An application which exchanges information with another application

	(see also Consumer).
Collection Description	A type of Package Description that is specialized to provide information about an Archival Information Collection for use by Access Aids.
Common Services	The supporting services such as inter-process communication, name services, temporary storage allocation, exception handling, security, and directory services necessary to support CASPAR.
Consumer	The role played by those persons, or client systems, who interact with CASPAR services to find preserved information of interest and to access that information in detail. This can include other CASPARs, as well as internal CASPAR persons or systems.
Content Data Object	The Data Object, that together with associated Representation Information, is the original target of preservation.
Content Information	The set of information that is the original target of preservation. It is an Information Object comprised of its Content Data Object and its Representation Information. An example of Content Information could be a single table of numbers representing, and understandable as, temperatures, but excluding the documentation that would explain its history and origin, how it relates to other observations, etc.
Context Information	The information that documents the relationships of the Content Information to its environment. This includes why the Content Information was created and how it relates to other Content Information objects.
Context	The information that documents the relationships of the Content Information to its environment. This includes why the Content Information was created and how it relates to other Content Information objects.

D

Data Dictionary	A formal repository of terms used to describe data.
Data Dissemination Session	A delivered set of media or a single telecommunications session that provides data to a Consumer. The Data Dissemination Session format/contents is based on a data model negotiated between CASPAR and the Consumer in the Request Agreement. This data model identifies the logical constructs used by CASPAR and how they are represented on each media delivery or in the telecommunication session.
Data Management Data	The data created and stored in Data Management persistent storage that refer to operation of an archive. Some examples of this data are accounting data for Consumer billing and authorization, policy data, Event Based Order (subscription) data for repeating requests, preservation process history data, and statistical data for generating reports to archive management.
Data Management	The CASPAR entity that contains the services and functions for populating, maintaining, and accessing a wide variety of information. Some examples of this information are catalogs and inventories on what may be retrieved from Archival Storage, processing algorithms that may be run on retrieved data, Consumer access statistics, Consumer billing, Event Based Orders, security controls, and CASPAR schedules, policies, and procedures.
Data Object	Either a Physical Object or a Digital Object.

Data Submission Session	A delivered set of media or a single telecommunication session that provides Data to CASPAR. The Data Submission Session format/contents is based on a data model negotiated between CASPAR and the Producer in the Submission Agreement. This data model identifies the logical constructs used by the Producer and how they are represented on each media delivery or in the telecommunication session.
Data	A reinterpretable representation of information in a formalized manner suitable for communication, interpretation, or processing. Examples of data include a sequence of bits, a table of numbers, the characters on a page, the recording of sounds made by a person speaking, or a moon rock specimen.
Derived AIP	An AIP generated by extracting or aggregating information from one or more source AIPs.
Descriptive Information	The set of information, consisting primarily of Package Descriptions, which is provided to Data Management to support the finding, ordering, and retrieving of CASPAR information holdings by Consumers.
Designated Community	An identified group of potential Consumers who should be able to understand a particular set of information. The Designated Community may be composed of multiple user communities.
Digital Migration	The transfer of digital information, while intending to preserve it, within CASPAR. It is distinguished from transfers in general by three attributes: (a) a focus on the preservation of the full information content, (b) a perspective that the new archival implementation of the information is a replacement for the old, (c) an understanding that full control and responsibility over all aspects of the transfer resides with CASPAR.
Digital Object	An object composed of a set of bit sequences.
Dissemination Information Package (DIP)	The Information Package, derived from one or more AIPs, received by the Consumer in response to a request to CASPAR.

E

Edition	An attribute of an AIP whose information content has been increased or improved from a source AIP and is therefore a candidate to replace the source AIP.
Event Based Order	A request that is generated by a Consumer for information that is to be delivered periodically on the basis of some event or events.

F

Federated Archives	A group of archives that has agreed to provide access to their holdings via one or more common finding aids.
Finding Aid	A type of Access Aid that allows a user to search for and identify Archival Information Packages of interest.
Fixity Information	The information which documents the authentication mechanisms and provides authentication keys to ensure that the Content Information object has not been altered in an undocumented manner. An example is a Cyclical Redundancy Check (CRC) code for a file.

G

Global	An extended Consumer community, in the context of Federated

Community	Archives, that accesses the holdings of several archives via one or more common Finding Aids.

I

Independently Understandable	A characteristic of information that has sufficient documentation to allow the information to be understood and used by the Designated Community without having to resort to special resources not widely available, including named individuals.
Information Object	A Data Object together with its Representation Information.
Information Package	The Content Information and associated Preservation Description Information which is needed to aid in the preservation of the Content Information. The Information Package has associated Packaging Information used to delimit and identify the Content Information and Preservation Description Information.
Information	Any type of knowledge that can be exchanged. In an exchange, it is represented by data. An example is a string of bits (the data) accompanied by a description of how to interpret a string of bits as numbers representing temperature observations measured in degrees Celsius (the representation information).
Ingest	The CASPAR entity that contains the services and functions that accept Submission Information Packages from Producers, prepares Archival Information Packages for storage, and ensures that Archival Information Packages and their supporting Descriptive Information become established within CASPAR.

K

Knowledge Base	A set of information, incorporated by a person or system, that allows that person or system to understand received information.

L

Local Community	The original Designated Community, in the context of Federated Archives, served by an archive.
Long Term Preservation	The act of maintaining information, in a correct and Independently Understandable form, over the Long Term.
Long Term	A period of time long enough for there to be concern about the impacts of changing technologies, including support for new media and data formats, and of a changing user community, on the information being held in a repository. This period extends into the indefinite future.

M

Management	The role played by those who set overall CASPAR policy as one component in a broader policy domain.
Member Description	An Associated Description that describes a member of a collection.
Metadata	Data about other data.
Migration	Transfer of data from one medium to another, or of software from one hardware platform to another, or of servers from one operating system to another.

O

Open Archival Information System (OAIS)	An archive, consisting of an organization of people and systems, that has accepted the responsibility to preserve information and make it available for a Designated Community.
Order Agreement	An agreement between the archive and the Consumer in which the physical details of the delivery, such as media type and format of Data, are specified.
Ordering Aid	An application that assists the Consumer in discovering the cost of, and in ordering, AIPs of interest.
Overview Description	A specialization of the Collection Description that describes the collection as a whole.

P

Package Description	The information intended for use by Access Aids.
Packaging Information	The information that is used to bind and identify the components of an Information Package. For example, it may be the ISO 9660 volume and directory information used on a CD-ROM to provide the content of several files containing Content Information and Preservation Description Information.
Physical Object	An object (such as a moon rock, bio-specimen, microscope slide) with physically observable properties that represent information that is considered suitable for being adequately documented for preservation, distribution, and independent usage.
Preservation (format)	File prepared for long-term preservation
Preservation Description Information (PDI)	The information which is necessary for adequate preservation of the Content Information and which can be categorized as Provenance, Reference, Fixity, and Context information.
Preservation/technology watch	Constant review of developments that affect the long-term maintenance and functioning of a digital archive
(Preservation) Processing	Preparation of files for preservation and dissemination
Producer	The role played by those persons, or client systems, who provide the information to be preserved. This can include other CASPARs or internal CASPAR persons or systems.
Producer/Depositor/Author	Individuals, organisations or systems that transfer information for long-term preservation
Provenance Information	The information that documents the history of the Content Information. This information tells the origin or source of the Content Information, any changes that may have taken place since it was originated, and who has had custody of it since it was originated. Examples of Provenance Information are the principal investigator who recorded the data, and the information concerning its storage, handling, and migration.

R

Reception	Process under which an archive is transferred for long-term preservation
Records	Information identified upon its creation as having value as evidence in 'recording' an activity or decision and so requiring a 'lifespan' for its

	management either through to destruction or appraisal and retention as part of an archive
Reference Information	The information that identifies, and if necessary describes, one or more mechanisms used to provide assigned identifiers for the Content Information. It also provides identifiers that allow outside systems to refer, unambiguously, to a particular Content Information. An example of Reference Information is an ISBN.
Reference Model	A framework for understanding significant relationships among the entities of some environment, and for the development of consistent standards or specifications supporting that environment. A reference model is based on a small number of unifying concepts and may be used as a basis for education and explaining standards to a non-specialist.
Refreshment	A Digital Migration where the effect is to replace a media instance with a copy that is sufficiently exact that all Archival Storage hardware and software continues to run as before.
Refreshment	Systematic copying of files between used and unused media. A Digital Migration where the effect is to replace a media instance with a copy that is sufficiently exact that all Archival Storage hardware and software continues to run as before.
Repackaging	A Digital Migration in which there is an alteration in the Packaging Information of the AIP.
Replication	A Digital Migration where there is no change to the Packaging Information, the Content Information, and the PDI. The bits used to represent these Information Objects are preserved in the transfer to the same or new media instance.
Repository	Site at which digital records are stored: can be a data warehouse for records or a digital archive for archives
Representation Information	The information that maps a Data Object into more meaningful concepts. An example is the ASCII definition that describes how a sequence of bits (i.e., a Data Object) is mapped into a symbol.
Representation Network	The set of Representation Information that fully describes the meaning of a Data Object. Representation Information in digital forms needs additional Representation Information so its digital forms can be understood over the Long Term.
Representation Rendering Software	A type of software that displays Representation Information of an Information Object in forms understandable to humans.
Result Set	The set of descriptive records for those AIPs in CASPAR which match the criteria stated in a Consumer query, or other results from a search on Data Management.
Retrieval Aid	An application that allows authorized users to retrieve the Content Information and PDI described by the Package Description.
Retrieval	Recovery of files for use
Risk analysis	Assessment of effects of adverse events on long-term preservation

S

Search Session	A session initiated by the Consumer with the archive during which the Consumer will use the archive Finding Aids to identify and investigate

	potential holdings of interest.
Secure (transfer/environment)	Protected from interference by non-authorised parties
Structure Information	The information that imparts meaning about how other information is organized. For example, it maps bit streams to common computer types such as characters, numbers, and pixels and aggregations of those types such as character strings and arrays.
Submission Agreement	The agreement reached between CASPAR and the Producer that specifies a data model for the Data Submission Session. This data model identifies format/contents and the logical constructs used by the Producer and how they are represented on each media delivery or in a telecommunication session.
Submission Information Package (SIP)	An Information Package that is delivered by the Producer to CASPAR for use in the construction of one or more AIPs.

T

Transfer	Movement of files between organisations
Transformation	A Digital Migration in which there is an alteration to the Content Information or PDI of an Archival Information Package. For example, changing ASCII codes to UNICODE in a text document being preserved is a Transformation.

U

Unit Description	A type of Package Description that is specialized to provide information about an Archival Information Unit for use by Access Aids.

V

Version	An attribute of an AIP whose information content has undergone a transformation on a source AIP and is a candidate to replace the source AIP.