Bibliothèque nationale de France

direction des services et des réseaux

département des systèmes d'information
quai François Mauriac 75706 Paris Cedex 13
téléphone 33.01 53 79 46 80 - télécopie 33.01 53 79 47 30

DISTRIBUTED ARCHIVING AND PRESERVATION SYSTEM BIBLIOTHÈQUE NATIONALE DE FRANCE

SPAR ANALYSIS

TABLE OF CONTENT

1 INTRODUCTION

This document defines conceptually the core of the Distributed Archiving and Preservation System (SPAR).

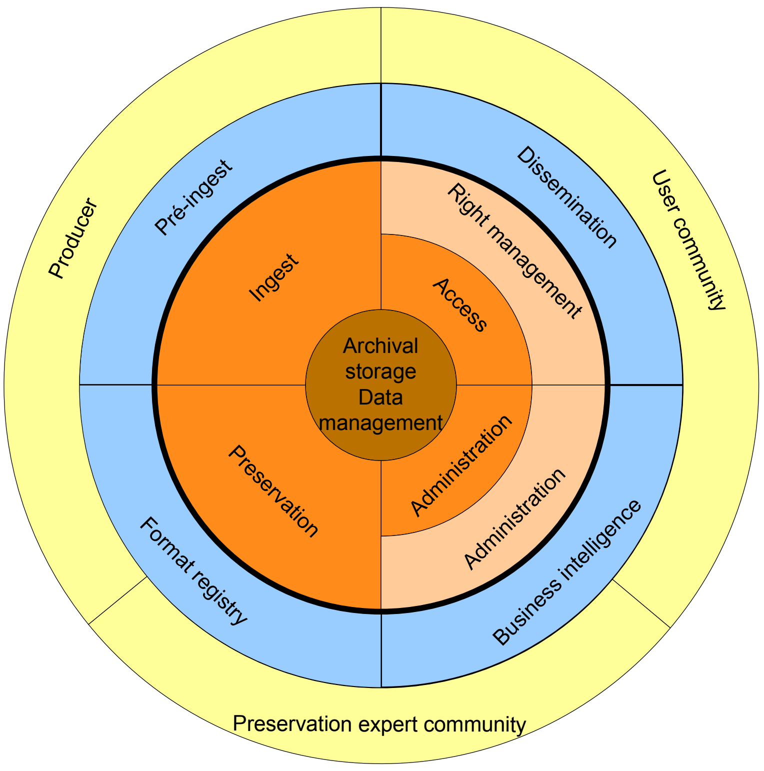
The RM-ODP model is used to realize the analysis (also know as ISO 10746). Only the enterprise and information viewpoint are detailed in this document : the other are more implementation defined.

The specificities of each channel of digitization (for example, the digitization for preservation) are described in a « Detailed Technical Book » (CTD or « Cahier Technique Détaillé »).

Note : this document is the translation of the annex 1 of the overall document “Cahier des Clauses Techniques Particulières (CCTP) du marché de réalisation du Système de Préservation et d’Archivage Réparti (SPAR)” (réf. BnF-ADM-2007-009581).

2 GENERAL CONCEPTUAL MODEL

The SPAR system can be represented conceptually by the following diagram:



This diagram shows the different layers that participate to the elaboration of an OAIS system.

It starts with the core (in orange) that is common to all the channels and which is described in detail in the following sections.

It goes then to the external modules (in blue) which are different for each channel and which lake the interface with the core.

Finally, the communities, users of the system, (in yellow) are represented by the most external layer. They never access directly to the core.

Each layer represent a different level of abstraction and interacts with the ones just up or below. These interactions are described more precisely in the document.

3 ENTERPRISE VIEWPOINT

3.1 USE CASE MODEL

As shown by I. Jacobson[1], use case models describe in a narrative fashion the behaviors of a system through its interactions with the outside: it represents the functional view of the requirements, from a user perspective as well as from the point of view of external systems with which the system will interact.. Both concepts, actors and use cases, define what is outside the system (actors) and what the system should perform (use cases).

For our specific system, since we need a modular architecture, two descriptions are provided:

- An external description which shows the interactions of the whole system with the exterior (the actors),
- An internal description which shows the interactions between the different modules of the system (each module been seen as a system per se).

3.1.1 ACTORS

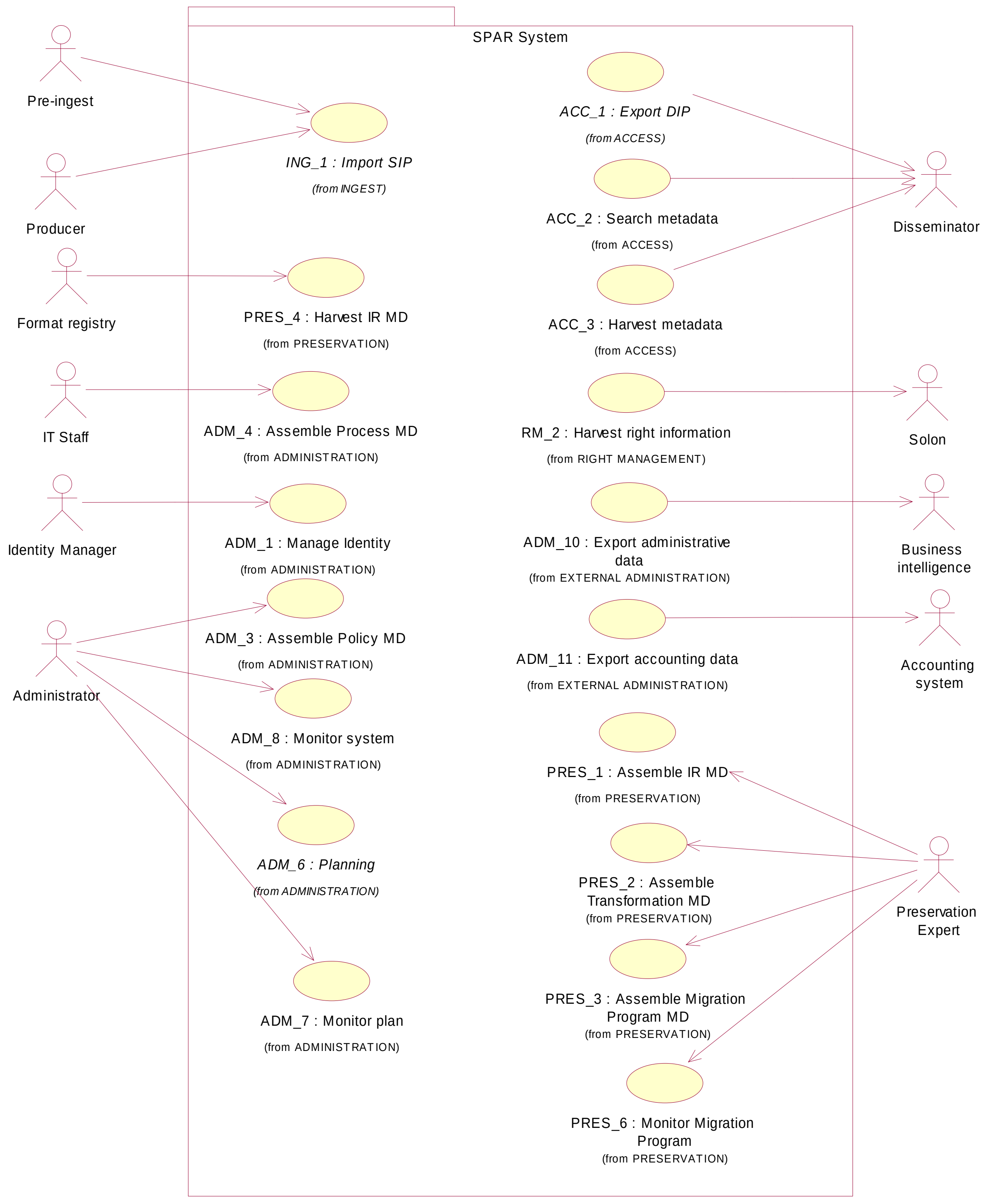
The different actors to be considered are:

- A **producer** or a **pre-ingest** process is allowed to import new packages or updates existing ones,
- A **disseminator** or dissemination system can search the metadata and export packages,
- The **business intelligence** system can export parts of the database to allow its own process (it's also a kind of disseminator)
- The **accounting system** can export parts of the database to be able to manage the financial aspect of the system (it's also a kind of disseminator)
- The **format registry** can add information representation metadata,
- An **IT department staff** defines and validates processes which metadata are imported,
- An **administrator** can add new policies (it's also a kind of disseminator),
- The **identity manager** system manages the different authorizations/authentications (users, producers, profiles, …),
- A **preservation expert** elaborates preservation programs and all the necessary elements by ensuring a technological survey (it's also a kind of disseminator),
- The **SOLON** system (cf. annex 4) provides the rights information (metadata and decision trees).

1 Ivar Jacobson, M. Christerson, P. Jonsson, G. Overgard,
Object-Oriented software engineering: A use case driven approach,
Addison Wesley Professional (ISBN 0201544350)]

3.1.2 EXTERNAL USE CASES

The following diagram shows the external view of the system with the use cases that interacts with the different actors.



As shown, the following external use cases are identified:

- Import SIP (ING_1)
- Harvest rights metadata (RM_2)
- Export DIP (ACC_1)
- Search in metadata (ACC_2)
- Harvest metadata (ACC_3)
- Manage identity (ADM_1)
- Assemble policy metadata (ADM_3)
- Assemble process metadata (ADM_4)
- Planning (ADM_6)
- Monitor a plan (ADM_7)
- Supervise system (ADM_8)
- Export administrative data (ADM_10)
- Export accounting data (ADM_11)
- Assemble Information Representation metadata (PRES_1)
- Assemble transformation package metadata (PRES_2)
- Assemble migration program metadata (PRES_3)
- Harvest Information Representation metadata (PRES_4)
- Monitor migration program (PRES_6)

We keep the relation between the use case and the module responsible for its initiation and interface with external actors.

3.1.2.1 IMPORT SIP (ING_1)

Use Case ID	**ING_1**
Description	Import of a SIP into the Ingest
Summary	This use case describes, in a generic manner, the import phase of a SIP, its transformation in AIP and its storage. This use case is abstract : its concrete instances are described in the section 3.1.3.1.
Level	*Abstract*
Actors	Producer or pre-ingest process
Assumptions	The SIP is well-formed and valid according to a specific class of SIP. A valid submission policy exists for this actor and the class of SIP. The Information Representation required is up-to-date. The processes required for the AIP generation are defined.
Pre-conditions	SIP characteristics (volume, number, submission frequency, etc..) are in accordance with the submission policy negotiated (cf. PAIMAS). Request is stored in historical database.
Primary functional path	1/ Submitter authenticates with the system. 2/ Accept SIP. 3/ Do QA (in particular, validate references to other AIP) and identify the class of SIP. 4/ Find the unique ID (generate one if new or retrieve the old one). 5/ Generate AIP from SIP. 6/ Extract metadata to be send to the “Data management” module. 7/ Ask for storage of the AIP. 8/ Deliver a receipt of the operation to the actor. 9/ Store record of the operation in historical database.
Primary result	AIP
Post-conditions	The Storage module validates the completeness of the storage transaction (including replication on different locations). The metadata are sent to the Data Management module. Outcome is stored in historical database.
Exceptional paths	1/ The SIP is not valid according to the QA : a non-conforming receipt is delivered to the actor. 2/ Failure in the storage transaction: a failure receipt is delivered and a trap is sent to the Administration.

3.1.2.2 HARVEST RIGHTS INFORMATION (RM_2)

Use Case ID	**RM_2**
Description	Harvest Rights metadata
Summary	This use case is meant for updating regularly the rights information (rights metadata, decision tree, use conditions) in behalf of Solon.
Level	
Actors	Solon
Assumptions	Frequency of update
Pre-conditions	
Primary functional path	1/ Connection to SOLON system 2/ Harvesting of new rights information 3/ If new rights metadata, update communicability status 4/ If new decision trees, invalidate communicability status
Primary result	
Post-conditions	
Exceptional paths	Connection impossible : a trap is sent to the Administration..

3.1.2.3 EXPORT DIP (ACC_1)

Use Case ID	**ACC_1**
Description	Export DIP
Summary	This use case describes the exportation of a DIP in a generic manner.. It's abstract and is specialized by two other use cases ACC_1.1 et ACC_1.2.
Level	Abstract
Actors	Disseminator
Assumptions	The actor knows the persistent identifier of the requested DIP. The requested DIP corresponds to a DIP class according to the access policy. The required Information Representation is up-to-date. The required processes (DIP transformations) are up-to-date.
Pre-conditions	Verify if DIP is not already in the cache, if not the DIP is generated from the AIP The request is stored in historical database.
Primary functional path	1/ Actor authenticates with the system 2/ Validation of the query (rights management and format requested) 3/ Use synchronous or asynchronous export (cf. specialized use case) 4/ Retrieval of the AIP 5/ Transformation or dissemination of the DIP 6/ DIP stored possibly in the cache (depending on access policy) 7/ Store record of the operation in historical database.
Primary result	DIP
Post-conditions	Cache updated with the new DIP according to the access policy. Outcome is stored in historical database.
Exceptional paths	1/ DIP not accessible to the actor => actor is provided an "access denied" 2/ The requested format is not accessible (either due to the rights or because of incompatibility with the format of the AIP) => a "bad format" is delivered to the actor with a possible alternative.

3.1.2.4 SEARCH IN METADATA (ACC_2)

Use Case ID	**ACC_2**
Description	Search in metadata
Summary	Search a persistent identifier or a collection of identifiers through a query on the metadata
Level	
Actors	Disseminator
Assumptions	The query conforms to the language defined by the interface
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Validation of the syntax of the query 3/ Validation that the query is allowed for this actor (the kind of metadata that can be queried depends on the role of the actor as well as the language used) 4/ Search the objects whose metadata match the query using only the Data management module. 5/ Filter the result set through the rights management
Primary result	Collection of persistent identifiers
Post-conditions	The query is stored in historical database.
Exceptional paths	

3.1.2.5 HARVEST METADATA (ACC_3)

Use Case ID	**ACC_3**
Description	Harvest metadata
Summary	A disseminator wants to harvest the metadata that have changed since a given date for part of the archive conforming with a particular data model (for example, the Dublin Core fields of the last ingested documents through the digitization channel in the last week)
Level	
Actors	Disseminator
Assumptions	Query in valid format: resumption date, set, data model
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Validation of the syntax of the query 3/ Search the objects whose metadata match the query using only the Data management module. 4/ Filter the result set through the rights management 5/ Elaborate the result metadata information
Primary result	Set of metadata information in appropriate format
Post-conditions	The query is stored in historical database.
Exceptional paths	

3.1.2.6 MANAGE IDENTITY (ADM_1)

Use Case ID	**ADM_1**
Description	Manage identity
Summary	This use case describes how the identities of the “users” which interact with SPAR are managed. This identities are provisioned by an external identity management system.
Level	
Actors	Identity manager system
Assumptions	The user to be recorded are those who represents actors that interact with the system: they can be human beings but more certainly programs or modules.
Pre-conditions	The provisioning is stored in the historical database.
Primary functional path	1/ The identity manager system authenticates with the system. 2/ Management of one or more identities: provisioning 3/ Store record of the operation in historical database
Primary result	Identities provisioned, or are up-to-date
Post-conditions	All the different actors can now interact with the system.
Exceptional paths	

3.1.2.7 ASSEMBLE POLICY METADATA (ADM_3)

Use Case ID	**ADM_3**
Description	Assemble policy metadata
Summary	The goal of this use case is to allow an administrator to describe a new policy with given community (producers or consumers) so that they can import or export new SIP/DIP. These policies influence the way AIP are stored and DIP are generated.
Level	
Actors	Administrator
Assumptions	The policy defined with a community (either producer or consumer) comes as a SIP. The data object can be the contract itself.
Pre-conditions	Request is stored in historical database.
Primary functional path	1/ Actors authenticates with the system. 2/ Receive the policy metadata in SIP format. 3/ Import this SIP (ING_2) 4/ Send receipt of the import to the actor. 5/ Store record of the operation in historical database
Primary result	AIP for a policy
Post-conditions	The Storage module validates the completeness of the storage transaction (including replication on different locations).. The metadata are sent to the Data Management module. Outcome is stored in historical database.
Exceptional paths	1/ The SIP is not valid according to the QA for a policy: a non-conforming receipt is delivered to the actor. 2/ Failure in the storage transaction: a failure receipt is delivered and a trap is sent to the Administration.

3.1.2.8 ASSEMBLE PROCESS METADATA (ADM_4)

Use Case ID	**ADM_4**
Description	Assemble process metadata
Summary	The process metadata are stored in the archive as AIP; so they are imported as any other SIP. The goal of this use case is to allow a IT staff to add information on available processes so that this information can be made persistent and can be referred to in the SPAR system itself.
Level	
Actors	IT staff Ingest module
Assumptions	The process metadata comes in SIP format.. The data object can be the specification of the process or even the source code of the program that realizes this process.
Pre-conditions	Request is stored in historical database.
Primary functional path	1/ Actors authenticates with the system. 2/ Receive the process metadata in SIP format. 3/ Import this SIP in the system (ING_2) 4/ Send receipt of the import to the actor : the receipt contains the external identifier of the process for a latter reference or use (transformation packages, AIP generation, DIP generation, …). 5/ Store record of the operation in historical database
Primary result	AIP of a particular process
Post-conditions	
Exceptional paths	1/ The SIP is not valid according to the QA for a process: a non-conforming receipt is delivered to the actor. 2/ Failure in the storage transaction: a failure receipt is delivered and a trap is sent to the Administration

3.1.2.9 PLANNING (ADM_6)

Use Case ID	**ADM_6**
Description	Planning
Summary	This use case describes, in a generic manner, how an administrator can launch the execution of a plan which holds for a very long period of time and which applies to a possibly important set of AIP.
Level	Abstract
Actors	Administrator
Assumptions	Choice query of AIP
Pre-conditions	
Primary functional path	1/ Building of the list of AIP to process (using the query provided) 2/ Creating a identifier for the monitoring of the execution of the plan 3/ For each ID of the list of AIP, *4/ execute the action (specified in each concrete use case)* 5/ Record the progress of the execution.
Primary result	Send the identifier of this particular plan
Post-conditions	All the list of AIP has been processed.
Exceptional paths	1/ Error in step 4 : record the error in the historical database (the error does not stop the execution of the plan)

3.1.2.10 MONITOR A PLAN (ADM_7)

Use Case ID	**ADM_7**
Description	Monitor a plan
Summary	This use case allows an administrator the monitoring of a plan previously defined (cf. ADM_6).
Level	
Actors	Administrator
Assumptions	The identifier of he execution of a particular plan is known
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system. 2/ Reception of the Identifier of the particular plan 3/ Retrieval of the status of the plan, of the number of already processed AIP, of the number of AIP still to be processed, of the time elapsed and of the estimation of the remaining time 4/ Formatting of the information and sending to the actor 5/ Store record of the operation in historical database
Primary result	Status of the plan
Post-conditions	
Exceptional paths	1/ Unknown plan 2/ Actor unauthorized to monitor plans

3.1.2.11 SYSTEM MONITORING (ADM_8)

Use Case ID	**ADM_8**
Description	System monitoring
Summary	This use case gathers together all the functions of monitoring of the system.
Level	
Actors	Administrator
Assumptions	
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system. 2/ Reception of the query (module, process … to monitor) 3/ Finding of the requested “object” and of its status 4/ Formatting of the information and sending to the actor 5/ Store record of the operation in historical database
Primary result	Status of the requested “object”
Post-conditions	
Exceptional paths	1/ Unknown or impossible to find “Object” 2/ Actor not authorized to monitor the system or the specific object

3.1.2.12 EXPORT ADMINISTRATIVE DATA (ADM_10)

Use Case ID	**ADM_10**
Description	Export administrative data
Summary	The need is to be able to build data repositories for the business intelligence system. This use case is part of the external administration module.
Level	
Actors	Business intelligence system
Assumptions	The historical database is filled up at each interaction with the system
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system. 2/ Reception of the request (temporal interval of interest, modules) 3/ Export of the right part of the historical database 4/ Store record of the operation in historical database
Primary result	Data exported
Post-conditions	
Exceptional paths	

3.1.2.13 Export accounting data (ADM_11)

Use Case ID	**ADM_12**
Description	Export accounting data
Summary	The need is to give data to the accounting system. This use case is part of the external administration module.
Level	
Actors	Accounting system
Assumptions	The historical database is filled up at each interaction with the system
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system. 2/ Reception of the request (temporal interval of interest, modules) 3/ Export of the right part of the historical database 4/ Store record of the operation in historical database
Primary result	Data exported
Post-conditions	
Exceptional paths	

3.1.2.14 Assemble Information Representation metadata (PRES_1)

Use Case ID	**PRES_1**
Description	Assemble Information Representation metadata
Summary	The goal of this use case id to allow a preservation expert to add information representation on formats (using an external format registry) so that this information is kept in perpetuity and could be referenced in the SPAR system itself.
Level	
Actors	Preservation expert Format registry Ingest module
Assumptions	The information representation metadata comes in the form of a SIP. The data-object is made of the description of the information representation.
Pre-conditions	Request stored in historical database.
Primary functional path	1/ Actor authenticates with the system. 2/ Elaboration of a SIP with the information coming from the format registry 3/ Ingest of this SIP in the system (ING_2). 4/ Receipt of the external identifier of this information representation for latter use (policy, process, …).
Primary result	AIP of the information representation metadata
Post-conditions	
Exceptional paths	1/ The SIP is not valid according to the QA process : a non-conforming receipt is sent to the actor. 2/ Failure in the storage transaction : a failure receipt is sent to the actor and a trap is sent to the administration.

3.1.2.15 ASSEMBLE TRANSFORMATION PACKAGE METADATA (PRES_2)

Use Case ID	**PRES_2**
Description	Assemble transformation package metadata
Summary	The transformation package metadata are stored in SPAR in the form of AIP and so are imported as any SIP. The goal of this use case is to allow a preservation expert to describe a transformation package in a formal way : choice request of candidate AIP, input information representation, output information representation, process
Level	
Actors	Preservation expert Ingest module Access module
Assumptions	The transformation package metadata comes in the form of a SIP. The data-object is made of the formal description of the transformation package.
Pre-conditions	Request stored in historical database.
Primary functional path	1/ Actor authenticates with the system. 2/ Elaboration of the SIP of the description of the transformation package : search the identifiers of information representation metadata search the identifier of the process write the query 3/ Ingest of this SIP in the system (ING_2). 4/ Receipt of the external identifier of this transformation package for latter use (migration program, …).
Primary result	AIP of the transformation package metadata
Post-conditions	
Exceptional paths	1/ The SIP is not valid according to the QA process : a non-conforming receipt is sent to the actor. 2/ Failure in the storage transaction : a failure receipt is sent to the actor and a trap is sent to the administration.

3.1.2.16 ASSEMBLE MIGRATION PROGRAM METADATA (PRES_3)

Use Case ID	**PRES_3**
Description	Assemble migration program metadata
Summary	The migration program metadata are stored in SPAR in the form of AIP and so are imported as any SIP. The goal of this use case is to allow a preservation expert to describe a The migration program in a formal way : context of the program, temporal constraints, transformation packages. An important aspect to keep in mind is that this AIP is only descriptive : the real implementation is defined, executed and monitored through the Administration module.
Level	
Actors	Preservation expert
Assumptions	The migration program metadata comes in the form of a SIP. The data-object can be the report that explains with the choice of such program, describes which AIP are selected and which transformations are applied.
Pre-conditions	Request stored in historical database.
Primary functional path	1/ Actor authenticates with the system. 2/ Elaboration of the SIP of the description of the migration program 3/ Ingest of this SIP in the system (ING_2). 4/ Receipt of the external identifier of this migration program for latter use.
Primary result	AIP of the migration program metadata
Post-conditions	
Exceptional paths	1/ The SIP is not valid according to the QA process : a non-conforming receipt is sent to the actor. 2/ Failure in the storage transaction : a failure receipt is sent to the actor and a trap is sent to the administration.

3.1.2.17 HARVEST INFORMATION REPRESENTATION METADATA (PRES_4)

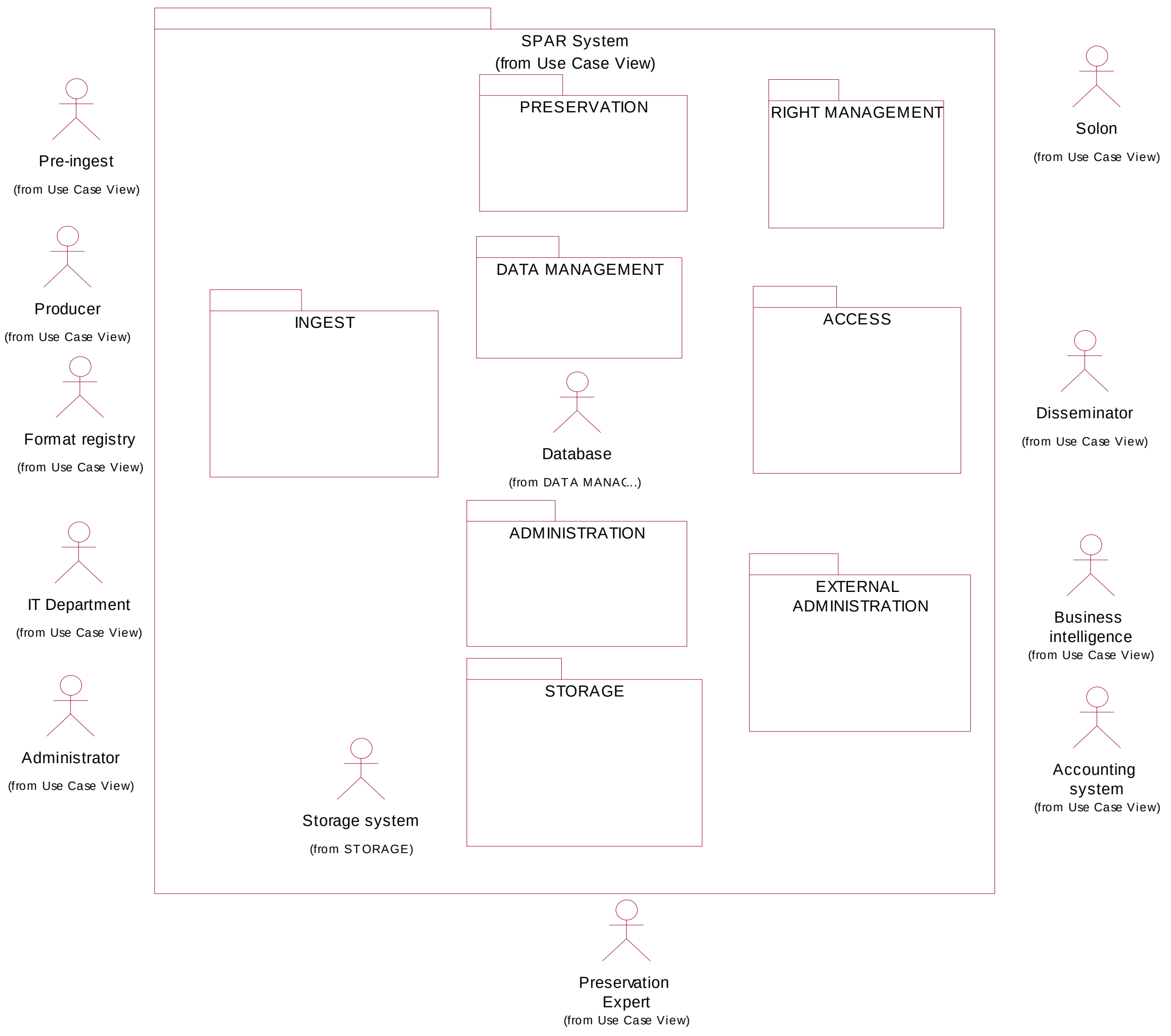
Use Case ID	**PRES_4**
Description	Harvest Information Representation metadata
Summary	This use case allows the collect in a regular basis of information representation metadata from a format registry.
Level	
Actors	Format registry
Assumptions	Frequency of the collect
Pre-conditions	The AIP of the information representation already exists.
Primary functional path	1/ Connect to the format registry 2/ Collect the new information representation metadata from the format registry. 3/ Lookup for the corresponding AIP 4/ For each new format with a corresponding modified, 5/ Build a SIP 6/ Submit the updated SIP (ING_2)
Primary result	Information representation AIP updated
Post-conditions	
Exceptional paths	1/ SIP refused : send a rap to the administration.

3.1.2.18 MONITOR MIGRATION PROGRAM (PRES_6)

Use Case ID	**PRES_6**
Description	Monitor migration program
Summary	This use case allows a preservation expert to monitor the execution of a migration program previously defined and launched by an administrator.
Level	
Actors	Preservation expert
Assumptions	The description of the migration program has previously been ingested (cf. use case PRES_3).
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system. 2/ Receipt of the identifier of the requested migration program 3/ Retrieval from the Administration of the status of the migration program, of the number of already processed AIP, of the number of AIP still to be processed, of the time elapsed and of the estimation of the remaining time 4/ Format of the information and send to the actor 5/ Store record of the operation in historical database
Primary result	Status of the migration program
Post-conditions	
Exceptional paths	1/ Unknown migration program 2/ Actor unauthorized to monitor a migration program

3.1.3 INTERNAL USE CASES

The following diagram is the function decomposition of the interns of the system :



The following sections describe the interns of each module with the use cases that explicit the interaction between them.

Two internal actors are invoked:

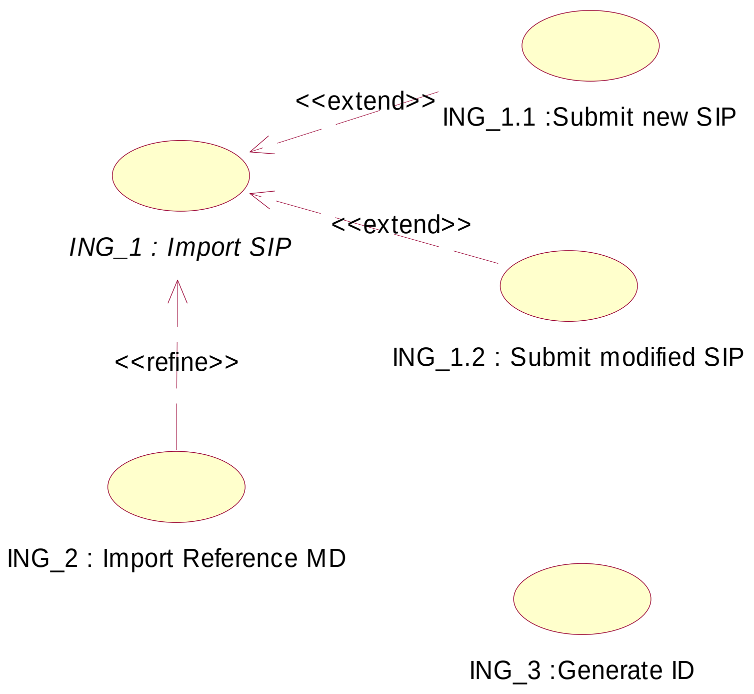
- The **Database** [2] : the system that stores all the metadata and gives access to it,
- The **Storage System** : the system that manages the infrastructure (in particular, it stores all the AIP and give access to them). This system is abstracted by means of the « Storage Abstract System » (SAS) described in the computational viewpoint.

2 By Database, we don't simply mean a Relational Database System, we speak about the whole of the systems which contribute to the management of the metadata (for example, it can be the association of mySQL and Kowari MetaStore)

3.1.3.1 Ingest Module

The ingest module is involved in two main kind of use cases :

- import SIP (ING_1),
- import reference metadata (ING_2)



The second one is a "simple" refinement of the first one (specialized in metadata SIP).

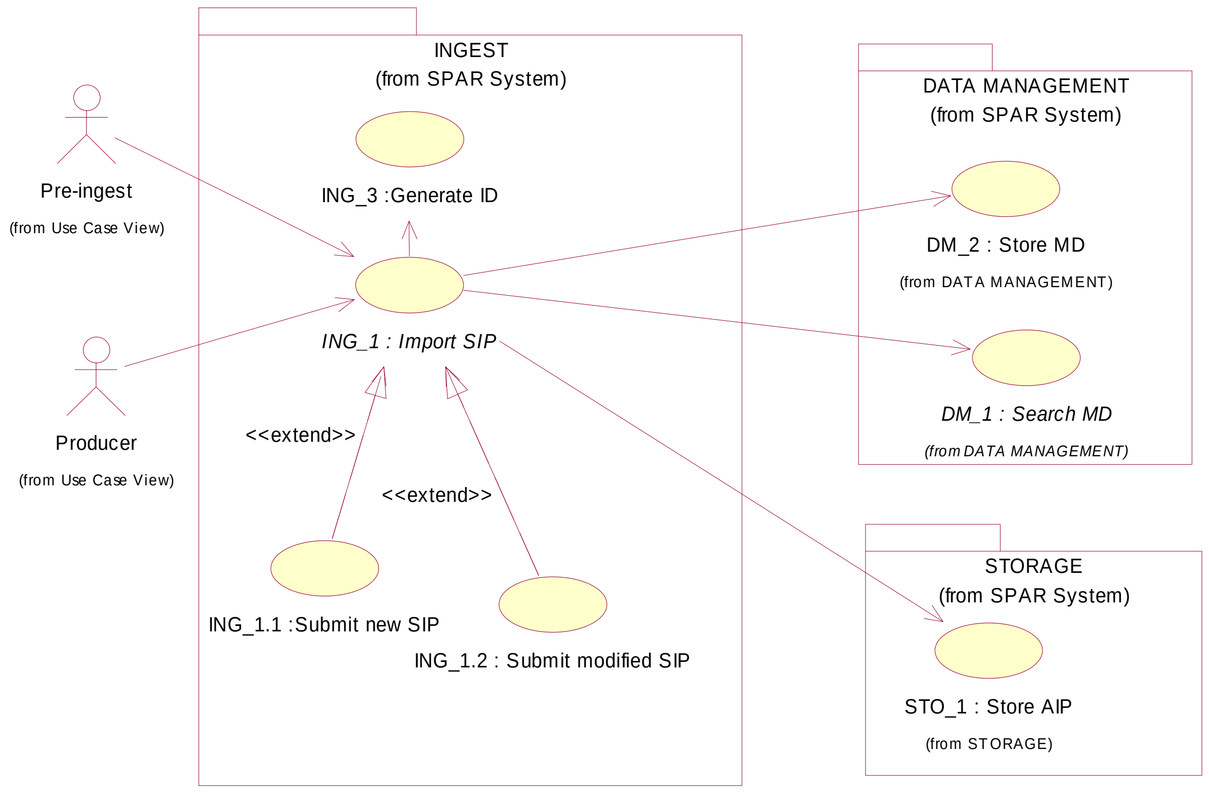
The first use case can be further decomposed in two sub cases:

- Submit new SIP (ING_1.1),
- Submit modified SIP (ING_1.2).

A last internal use case is defined:

- Generate ID (ING_3)

The interactions with the other modules are defined in the following diagram:



3.1.3.1.1 Submit new SIP (ING_1.1)

Use Case ID	**ING_1.1**
Description	Submit new SIP
Summary	This use case describes the ingest of a new information package, which means a package with a unique identifier from the producer perspective.
Level	Specialization of ING_1
Actors	Producer or Pre-Ingest
Assumptions	Idem ING_1
Pre-conditions	Idem ING_1 + The package has never been submitted before (the identifier given by the producer is not already known).
Primary functional path	Step 4/ is modified by: 4/ Generate a unique ID (ING_3)
Primary result	AIP
Post-conditions	Idem ING_1
Exceptional paths	Idem ING_1

3.1.3.1.2 Submit modified SIP (ING_1.2)

Use Case ID	**ING_1.2**
Description	Submit modified SIP
Summary	This use case describes how an existing AIP is updated with the providing of a SIP. The identifier given to get back the already ingested package can be: - The previously given production identifier (external ID) - The external access identifier (coming for example from the receipt given back during the previous ingest).
Level	Specialization of ING_1
Actors	Producer or Pre-Ingest
Assumptions	Idem ING_1
Pre-conditions	Idem ING_1 + The package has already been ingested (an ID to identified it is given)
Primary functional path	Step 4/ is modified by: 4.1/ Retrieve last version/edition of the Pn package. 4.2/ Update the Pn package according to the content of the SIP 4.3/ Identify the kind of modification to generate a new version or a new edition (cf. AIP lifecycle)
Primary result	AIP
Post-conditions	Idem ING_1
Exceptional paths	Idem ING_1 3/ No previous version/edition : a “new package” receipt is sent

3.1.3.1.3 Import reference MD (ING_2)

Use Case ID	**ING_2**
Description	Import reference MD
Summary	This use case allows the ingest of any reference description. Those descriptions define the loosely coupled metadata (cf. information viewpoint) for the others AIP.
Level	Refinement of ING_1
Actors	Producer or Pre-Ingest Preservation module
Assumptions	Idem ING_1
Pre-conditions	Idem ING_1 + The package has only reference information.
Primary functional path	Step 6/ is modified by: 6a/ Extract the metadata from the data object 6b/ Send these metadata to the Data Management module for immediate retrieval.
Primary result	Reference description AIP
Post-conditions	Idem ING_1
Exceptional paths	Idem ING_1

3.1.3.1.4 Generate ID (ING_3)

Use Case ID	**ING_3**
Description	Generate ID
Summary	This internal use case has been put aside to enhance its specificity and its importance in a distributed environment. Indeed it has to generate a *unique* identifier.
Level	
Actors	Ingest module
Assumptions	
Pre-conditions	Actor authenticated with the system The actor asks either for an internal identifier or an access one.
Primary functional path	1/ Generate a unique identifier: - The uniqueness MUST be global for the system (inside SPAR), - It CAN be made universal (through the add of a unique prefix; for example, the BnF prefix is ark:/12148).
Primary result	ID
Post-conditions	
Exceptional paths	1/ Impossibility of generating a globally unique ID

The identifiers of the AIP can be of 3 kinds:

- The **external** IDs (IDs generated by other systems: ISSN, record number, production identifier, …) which are only stored to keep track in the history of the package : their format depends on their generation system,
- The **internal** IDs which identifies in the SPAR system each instance of AIP in a unique manner: their format are unspecified and depends on the implementation,
- The **access** IDs which allows an external actor to reference an AIP : their format respects the ARK format defined by BnF.

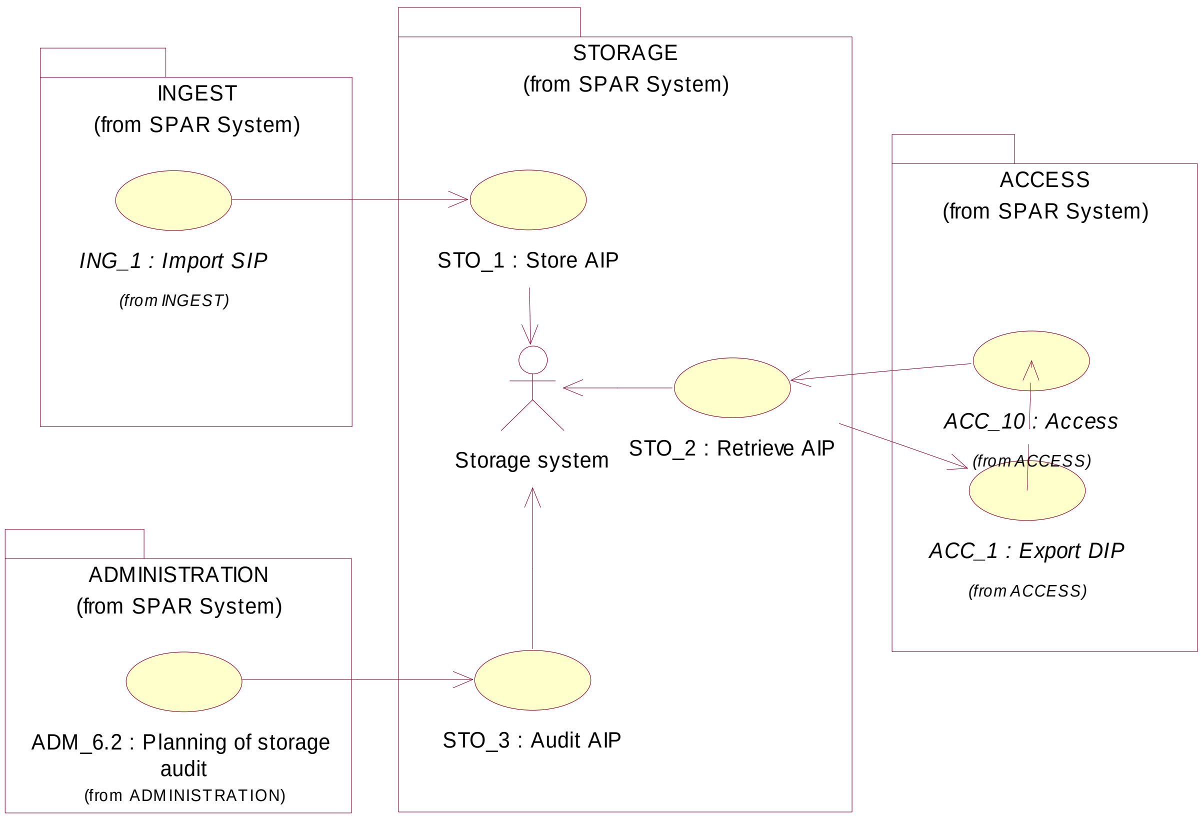
For a more in depth discussion on identifiers, see the information viewpoint.

3.1.3.2 STORAGE MODULE

The storage module is an internal module with no interaction with an external actor. Its main goal is to manage the AIP packages by interacting with the storage system. Three principal use cases are defined:

- Store AIP (STO_1),
- Retrieve AIP (STO_2),
- Audit AIP (STO_3).

The interactions with the other modules are defined in the following diagram:



3.1.3.2.1 Store AIP (STO_1)

Use Case ID	**STO_1**
Description	Store AIP
Summary	This use case describes the storage of an AIP. In the response, a token is given so that the exact status of the storage step (2) can be asked on a regular basis until the completion of the preservation policy is reached.
Level	
Actors	Ingest module Storage System
Assumptions	AIP well formed and containing its preservation policy
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Supply the AIP with its preservation policy (number of records, kind of storage, hierarchy of storage) to the Storage System 3/ Reach, from the Storage System, a basic level of security for the AIP (it exists at least one record in a non volatile[3] storage volume) 4/ Response to the actor with a token for this operation. 5/ Store record of the operation in historical database.
Primary result	Token for the responsibility assumption of the AIP by the storage system
Post-conditions	AIP stored in the storage system and available for a retrieval. The storage system takes care of the AIP according to the preservation policy.
Exceptional paths	1/ Error in the supply : a “Storage failure” receipt is sent, a critical trap is sent to the administration

3 A **non volatile** memory is a memory which keeps its content even with power supply.

3.1.3.2.2 Retrieve AIP (STO_2)

Use Case ID	**STO_2**
Description	Retrieve AIP
Summary	This use case describes the retrieval of a AIP. The retrieval of an AIP is asynchronous; it allows the monitoring of the different steps of the retrieval with respect to the storage system (error controls, usage of secondary records, …)
Level	
Actors	Access module Storage system
Assumptions	ID of the AIP is known.
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Retrieval query by ID of the AIP in behalf of the Storage System 3/ Since the execution time can not be predicted, a token (with a resumption time) is given in return, to allow the monitoring of the steps of the retrieval and the final retrieving of the AIP 4/ Record of the operation in historical database.
Primary result	AIP
Post-conditions	
Exceptional paths	1/ Unknown ID : a “Unknown package” receipt is sent 2/ The Storage System is unreachable : a “Storage Error” is sent, a trap is sent to the Administration 3/ The resumption time of the token is reached: a “timeout” is sent, a trap is sent to the administration 4/ The Storage System can’t give any valid AIP record: a critical trap is sent to the administration

3.1.3.2.3 Audit AIP (STO_3)

Use Case ID	**STO_3**
Description	Audit AIP
Summary	This case describes the audit of an AIP. To make an audit of an AIP, a request for an audit is sent to the Storage System and an internal validation of the content of the information package is made.
Level	
Actors	Administration module Storage system
Assumptions	ID of the AIP is known
Pre-conditions	
Primary functional path	1/ Retrieval through audit of the AIP from the Storage System 2/ Make internal validation of the content of the AIP. 3/ Record of the operation in historical database.
Primary result	Validation of the AIP
Post-conditions	
Exceptional paths	1/ Unknown ID : a “Unknown package” receipt is sent 2/ The Storage System is unreachable : a “Storage Error” is sent, a trap is sent to the Administration 3/ The Storage System can’t give any valid AIP record: a critical trap is sent to the administration 4/ The internal validation finds an error or an inconsistency: a critical trap is sent to the administration

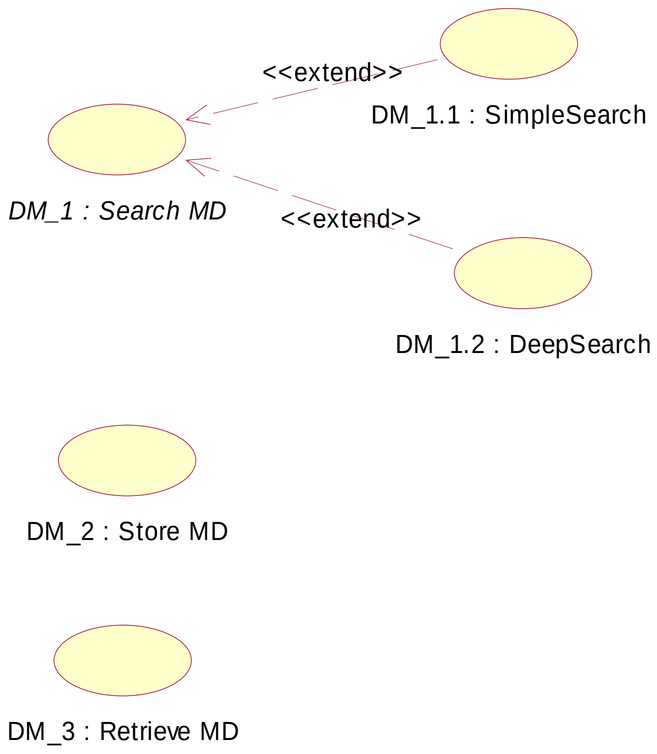
3.1.3.3 DATA MANAGEMENT MODULE

The data management module is an internal module with no interaction with any external actor. Its main goal is to manage the metadata associated with the AIP to allow efficient search. Its reaches its goal by interacting with the Database (which is not necessarily monolithic). Three principal use cases are defined:

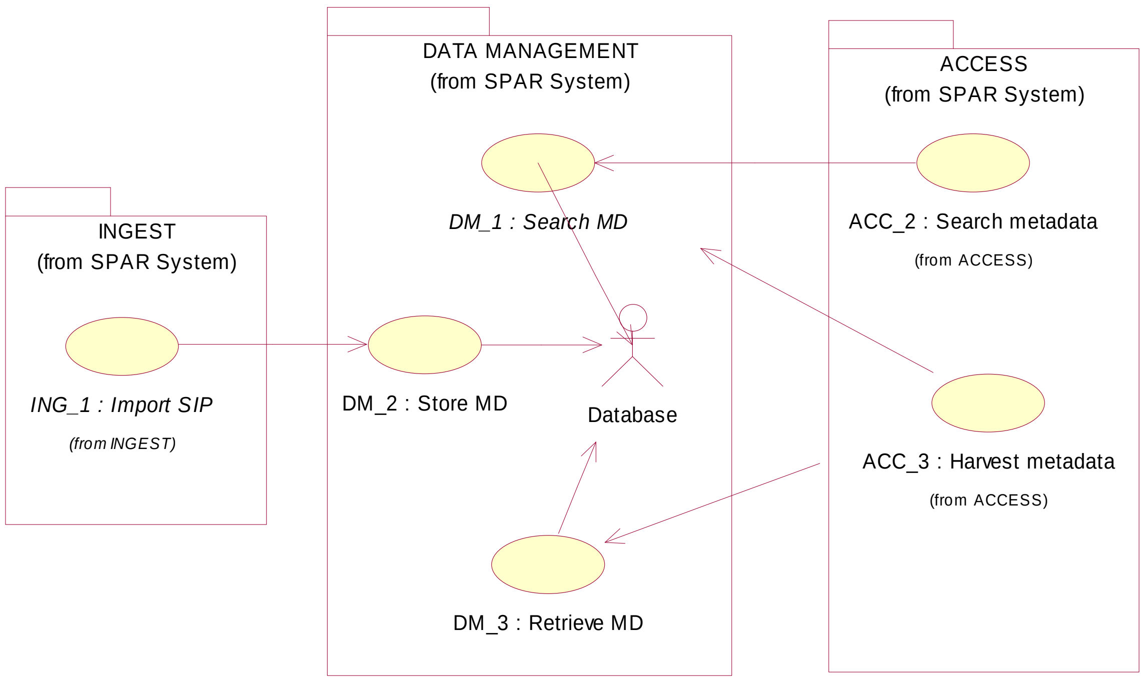
- Search the metadata (DM_1),
- Store metadata (DM_2),
- Retrieve metadata (DM_3).

The first use case exists in two instantiations:

- Simple search (DM_1.1)
- Deep search (DM_1.2)



The interactions with the other modules are defined in the following diagram:



3.1.3.3.1 Search the metadata (DM_1)

Use Case ID	**DM_1**
Description	Search the metadata
Summary	This case describes in a generic manner the search in the metadata stored in the Data Management module.
Level	*Abstract*
Actors	Access module Database
Assumptions	The query respects the language of the interface.
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Validation of the syntax of the query 3/ Run the query 5/ Extract the access IDs in the result of the query
Primary result	Collection of access identifiers
Post-conditions	Query store in historical database.
Exceptional paths	

3.1.3.3.2 Simple search (DM_1.1)

Use Case ID	**DM_1.1**
Description	Simple search
Summary	This case describes the simple search in the metadata : which means search in the indexed metadata. This search is synchronous.
Level	
Actors	Access module Database
Assumptions	Idem DM_1 + The query is limited to certain predefined (by the indexes) combinations (for example, paths in XPath)
Pre-conditions	
Primary functional path	Idem DM_1
Primary result	Collection of access identifiers
Post-conditions	Idem DM_1
Exceptional paths	The service exceeds assigned time (technical parameter) : a « timeout » is sent, a trap is sent to the administration

3.1.3.3.3 Deep search (DM_1.2)

Use Case ID	**DM_1.2**
Description	Deep search
Summary	This case describes the deep search in the metadata : this search can query any of the stored metadata. Since this search can possibly take a long time, it's made asynchronous.
Level	
Actors	Access module Database
Assumptions	Idem DM_1. The query can imply all the stored metadata.
Pre-conditions	
Primary functional path	Idem DM_1 except for the step 3: 3/ Since the execution time is not predictable, a token (with a resumption time) is given in return, which allows to request for the status of the query and eventually to retrieve the result.
Primary result	Collection of access identifiers
Post-conditions	Idem DM_1
Exceptional paths	1/ The resumption date is expired : a "timeout" is sent, a trap is sent to the administration.

3.1.3.3.4 Store metadata (DM_2)

Use Case ID	**DM_2**
Description	Store metadata
Summary	This case describes the store of the metadata : it can be an insertion or an update.
Level	
Actors	Ingest module Database
Assumptions	Metadata in proper format
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Store the metadata in the Database 3/ Update indexes, if necessary. 4/ Acknowledge the operation to the actor. 5/ Record of the operation in historical database.
Primary result	
Post-conditions	Metadata stored in Database and accessible to query.
Exceptional paths	1/ Failure in storing : a failure acknowledgement is delivered

3.1.3.3.5 Retrieve metadata (DM_3)

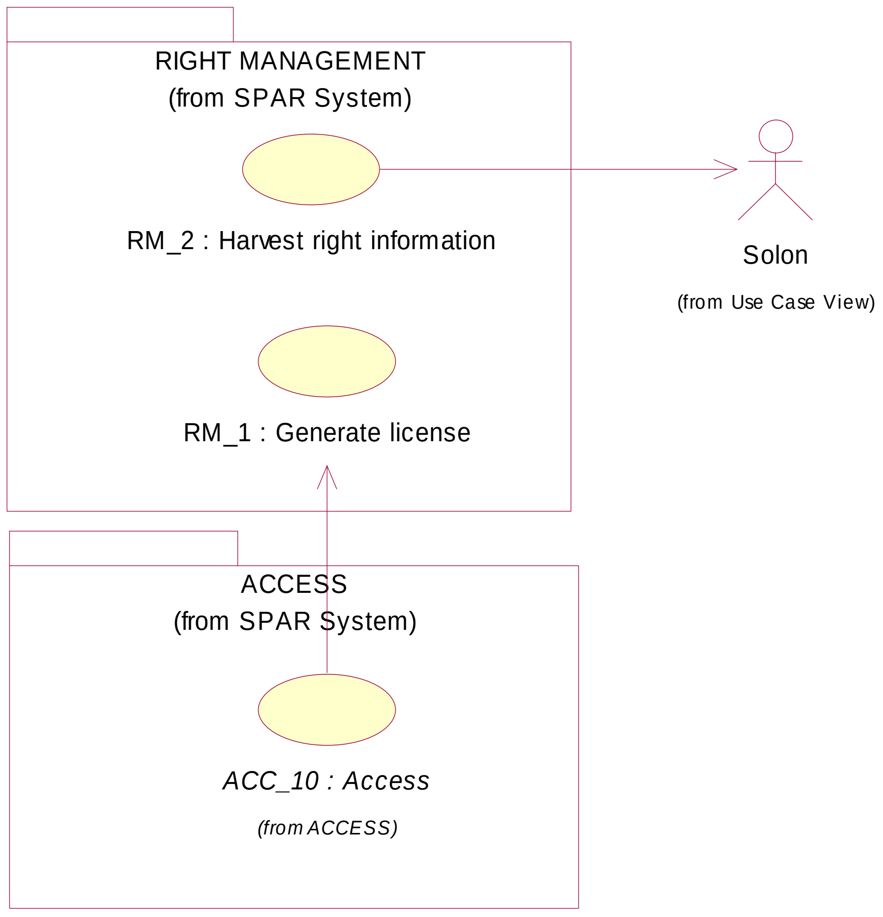
Use Case ID	**DM_3**
Description	Retrieve metadata
Summary	This use case describes the retrieval of metadata.
Level	
Actors	Access module Database
Assumptions	ID of the AIP represented by the metadata is known A transformation (defining which parts of the metadata is needed) is provided.
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Retrieve the metadata from the Database by ID 3/ Apply the transformation asked for 4/ Record of the operation in historical database.
Primary result	Transformed metadata
Post-conditions	
Exceptional paths	1/ Unknown ID : a “Unknown package” receipt is sent 2/ Transformation failure : a “transformation failure” receipt is sent

3.1.3.4 RIGHT MANAGEMENT MODULE

The right management module is a module in charge of managing the right information associated with each output DIP. It's fed by the Solon system (cf. annex 4) to get the rights metadata[4] as well as the decision tree. It "executes" those decision tree depending on the final users to associate a license to each output DIP. This filter being systematically applied to every output information, it's quite important to be able to precalculate the maximum number of information to avoid penalties in the output. Two use cases are defined:

- Generate a license (RM_1)
- Harvest rights metadata (RM_2)

The interactions with the other modules or external actors are defined in the following diagram:



4 Non persistent rights information on each AIP

3.1.3.4.1 Generate a license (RM_1)

Use Case ID	**RM_1**
Description	Generate a license
Summary	This case describes the generation of a license for a particular DIP. Given the access constraints, it is probable that the steps 2, 3 and 4 need to be bypass by the installation of a precalculation system and of a cache.
Level	
Actors	Access module
Assumptions	The actor knows the access identifier of the DIP
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Retrieval of the rights metadata for the DIP 3/ Selection of the appropriate decision tree or agreement 4/ Traversal of the tree, if necessary, and generation of the accessibility status 5/ Apply the usage context 6/ Generate the license
Primary result	License
Post-conditions	
Exceptional paths	1/ Impossible calculation : generate a license without any right, send a trap to the administration

3.1.3.5 ACCESS MODULE

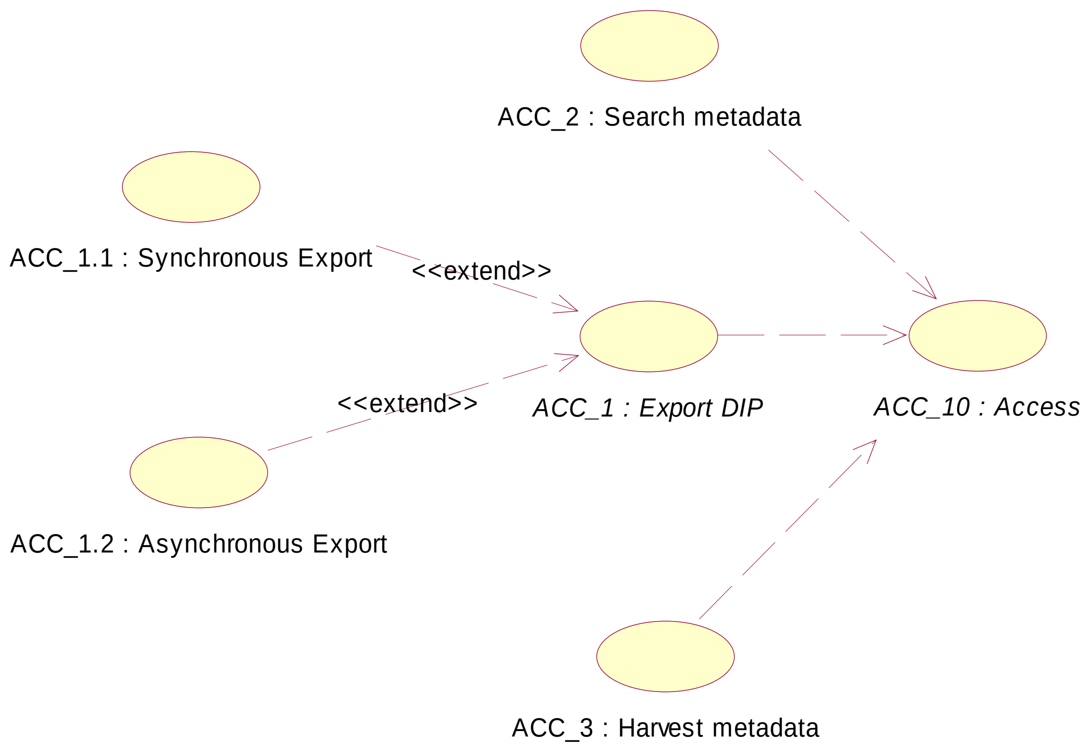
The access module is involved in three main kinds of use cases:

- Export DIP (ACC_1),
- Search metadata (ACC_2),
- Harvest metadata(ACC_3).

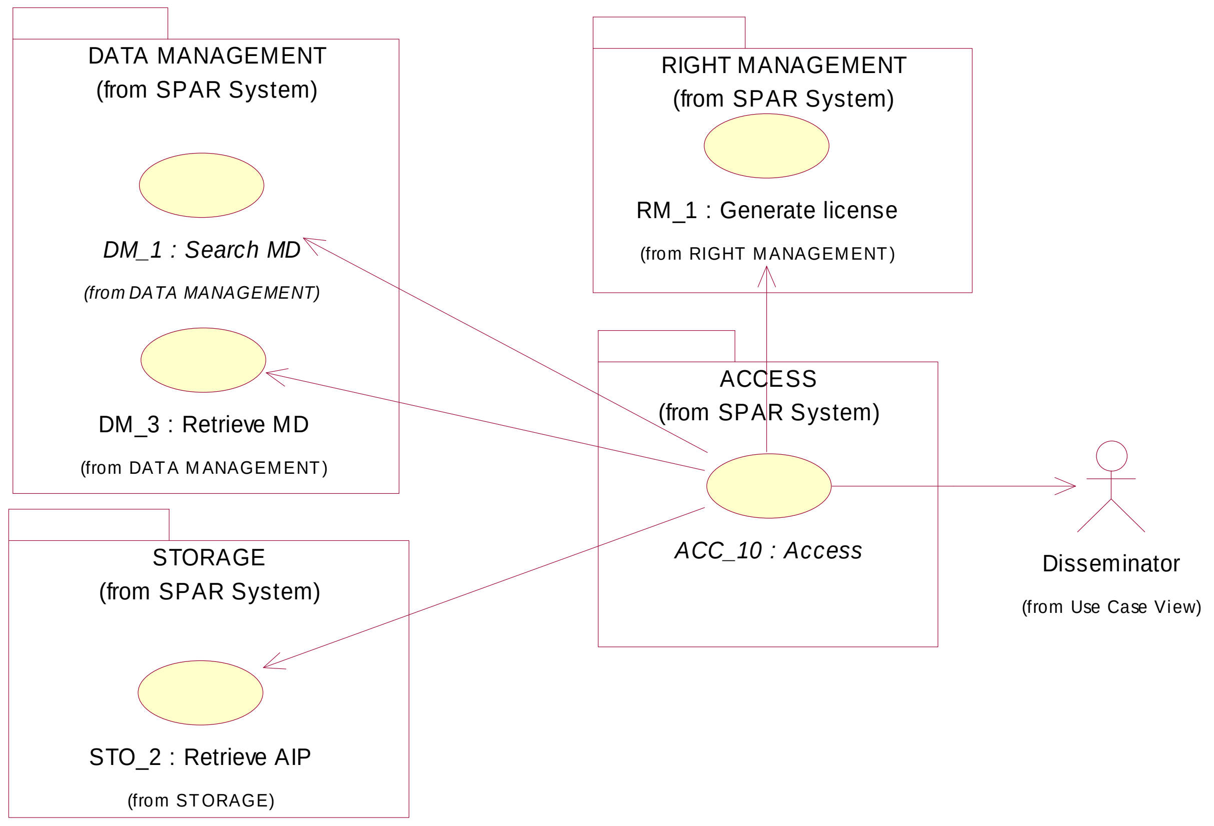
The first use case exists in several instantiations:

- Synchronous export of DIP (ACC_1.1)
- Asynchronous export of DIP (ACC_1.2)

Moreover, all the use cases are refinements of the *Access* abstract use case that mutualizes all the common operations (authentication and rights filtering).



The interactions with the other modules or external actors are defined in the following diagram:



3.1.3.5.1 Access (ACC_10)

Use Case ID	**ACC_10**
Description	Access
Summary	This abstract use case shares all the common behaviors of all the use cases of this module: - **each** access is made through an authenticated actor, - **all** the results are always filtered through the Right management module.
Level	*Abstract*
Actors	Disseminator
Assumptions	The actor is registered
Pre-conditions	
Primary functional path	1/ Actor authenticates with the system 2/ Validation of the access action 3/ Filter the result according to the rights management
Primary result	DIP
Post-conditions	Outcome is stored in historical database
Exceptional paths	1/ The actor is unknown or cannot access to the system => a denied access is notified.

3.1.3.5.2 Synchronous export of DIP (ACC_1.1)

Use Case ID	**ACC_1.1**
Description	Synchronous export of DIP
Summary	This use case describes the synchronous export of a DIP. It's a specialization of the case ACC_1.
Level	Specialization of ACC_1
Actors	Disseminator
Assumptions	Idem ACC_1
Pre-conditions	Idem ACC_1
Primary functional path	Idem ACC_1 except for the step 3: 3/ Synchronous export: the actor waits until the DIP is available.
Primary result	DIP
Post-conditions	Idem ACC_1
Exceptional paths	Idem ACC_1 + 3/ The service exceeds assigned time defined by the access policy relative to the actor: a « timeout » is sent, a trap is sent to the administration.

3.1.3.5.3 Asynchronous export of DIP (ACC_1.2)

Use Case ID	**ACC_1.2**
Description	Asynchronous export of DIP
Summary	This use case describes the synchronous export of a DIP. It's a specialization of the case ACC_1.
Level	Specialization de ACC_1
Actors	Disseminator
Assumptions	Idem ACC_1
Pre-conditions	Idem ACC_1
Primary functional path	Idem ACC_1 except for the step 3: 3/ Asynchronous export: a token (with a resumption time) is given in return, which allows to request for the status of the availability of the DIP and eventually to retrieve it.
Primary result	DIP
Post-conditions	Idem ACC_1
Exceptional paths	Idem ACC_1 + 3/ The resumption date is expired : a "timeout" is sent, a trap is sent to the administration.

3.1.3.6 ADMINISTRATION MODULE

The administration module has many roles:

- it's the interface of advanced users (in particular, the administrators),
- it's a mean for specialized systems (business intelligence or accounting systems) to retrieve global information about the system,
- but it's also an internal module which ensures the cohesion of the whole system and a monitoring function.

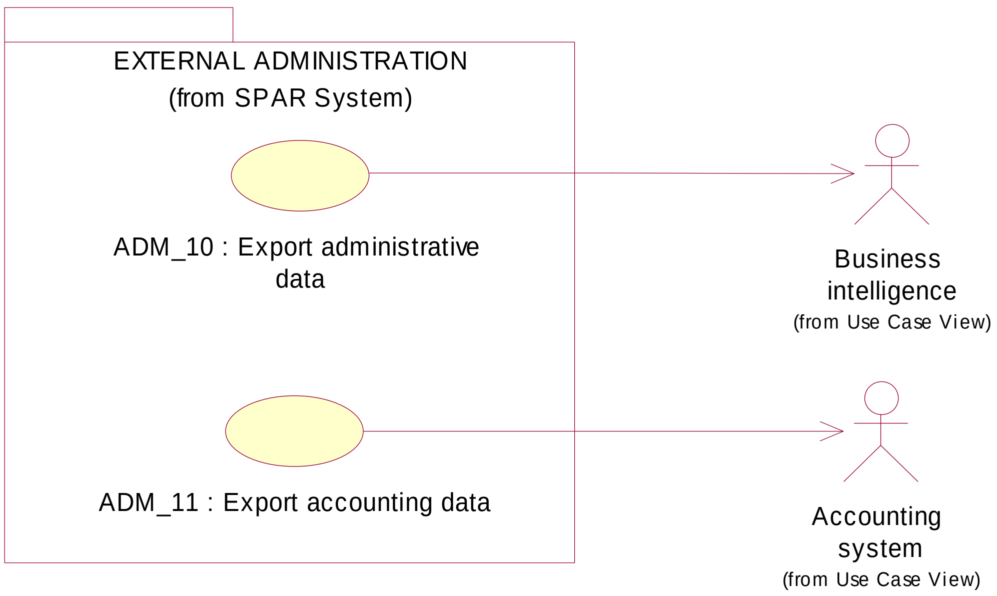
The external use cases are divided in two groups.

1/ The use cases in interaction with advanced users:

- Manage identity (ADM_1),
- Assemble policy metadata (ADM_3),
- Assemble process metadata (ADM_4),
- Planning (ADM_6),
- Monitor a plan (ADM_7),
- System monitoring (ADM_8).

2/ The use cases made for specialized systems form a specific sub-module (the External Administration module):

- Export administrative data (ADM_10),
- Export accounting data (ADM_11),

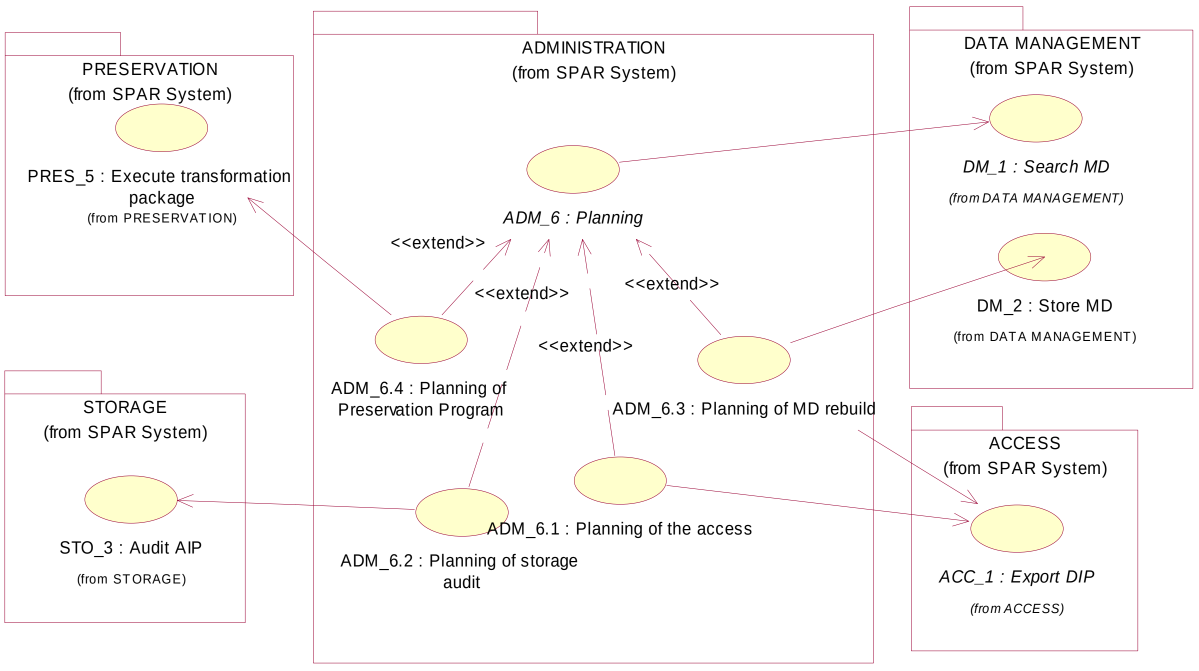


The internal use case are:

- Identity federation (ADM_2),
- Planning of the access (ADM_6.1),
- Planning of storage audit (ADM_6.2),
- Planning of metadata rebuild (ADM_6.3),

- Planning of migration program (ADM_6.4).

The following diagram shows the interactions with the other modules of the concrete use cases of the planning:



3.1.3.6.1 Identity federation (ADM_2)

Use Case ID	**ADM_2**
Description	Identity federation
Summary	Each module manages in an autonomous manner its authentication function. The consistency of the whole is ensured by the Administration. This use case reflects the reality of this consistency through a federation.
Level	
Actors	Any module
Assumptions	The users that authenticated are those that represents actors that need to interact with the system : they can be real persons but more commonly they are modules or programs.
Pre-conditions	Request stored in historical database
Primary functional path	1/ Connection to each module 2/ Update of the identity information (following an update made by ADM_1)
Primary result	Identities are distributed over all the modules
Post-conditions	The user can interact with the system
Exceptional paths	

3.1.3.6.2 Planning of the access (ADM_6.1)

Use Case ID	**ADM_6.1**
Description	Planning of the access
Summary	This use case describes how the system manages to reach the access policies.
Level	Specialization of ADM_6
Actors	Administrator Data management module Access module
Assumptions	
Pre-conditions	Definition of an access policy (ADM_3)
Primary functional path	Idem ADM_6 but Step 1 : 1/ Search the metadata (DM_1) to retrieve the new AIP corresponding to the access policy : this defines the list of AIP to process. Step 4 : 4/ Export DIP (ACC_1) so that the format(s) required by the access policy are in the cache.
Primary result	Idem ADM_6
Post-conditions	Needed DIP in the cache
Exceptional paths	Idem ADM_6 + 2/ DIP not generated : send a trap to the administration.

3.1.3.6.3 Planning of storage audit (ADM_6.2)

Use Case ID	**ADM_6.2**
Description	Planning of storage audit
Summary	This use case describes how the system manages to be sure that the audit constraints defined by the preservation policy are reached.
Level	Specialization of ADM_6
Actors	Administrator Data management module Storage module
Assumptions	
Pre-conditions	Definition of a preservation policy (ADM_3)
Primary functional path	Idem ADM_6 but Step 1 : 1/ Make of the list of AIP to audit with the levels of service (audit frequency) and the historical database. Step 4 : 4a/ Ask the Storage module to audit the AIP (cf. STO_3). 4b/ Update the audit report in the administrative data (journal).
Primary result	Idem ADM_6
Post-conditions	Audit report
Exceptional paths	Idem ADM_6 + 2/ Failure in audit : sent a critical trap to the administration.

3.1.3.6.4 Planning of metadata rebuild (ADM_6.3)

Use Case ID	**ADM_6.3**
Description	Planning of metadata rebuild
Summary	This use cases describes how the system manages to rebuild the metadata stored in the Data Management module from the AIP.
Level	Specialization of ADM_6
Actors	Administrator Data management module Access module
Assumptions	
Pre-conditions	
Primary functional path	Idem ADM_6 but Step 1 : 1/ Made of the list of AIP to process by the Administrator or by harvesting (ACC_3). Step 4 : 4a/ Retrieval of a DIP (ACC_1) to extract the metadata 4b/ Store this metadata (DM_2).
Primary result	Idem ADM_6
Post-conditions	Metadata rebuild
Exceptional paths	Idem ADM_6 + 2/ Failure in transformation : send a trap to the administration.

3.1.3.6.5 Planning of migration program (ADM_6.4)

Use Case ID	**ADM_6.4**
Description	Planning of migration program
Summary	This use case describes how the system manages to complete migration in order to preserve AIP.
Level	Specialization of ADM_6
Actors	Administrator Data management module Preservation module
Assumptions	
Pre-conditions	Definition of a migration program (PRES_3)
Primary functional path	Idem ADM_6 but Step 1 : 1/ Made of the list of AIP to process based on the query given in the migration program. Step 4 : 4/ Ask the preservation module to transform the AIP according to the transformation package (cf. PRES_5).
Primary result	Idem ADM_6
Post-conditions	AIP updated
Exceptional paths	Idem ADM_6 + 2/ Failure in the transformation : send a trap to the administration.

3.1.3.7 PRESERVATION MODULE

The preservation module is a module which allows the definition and the monitoring of the formats and the standards used by the SPAR system.

It's mainly made of tools to build:

- formal specifications of process,
- of migration programs,
- of formats (information representation).

It is fed by information coming from the business intelligence System as well as the registry format so that it can follow changes in formats or plan evolutions in storage or policy (preservation and access policy).

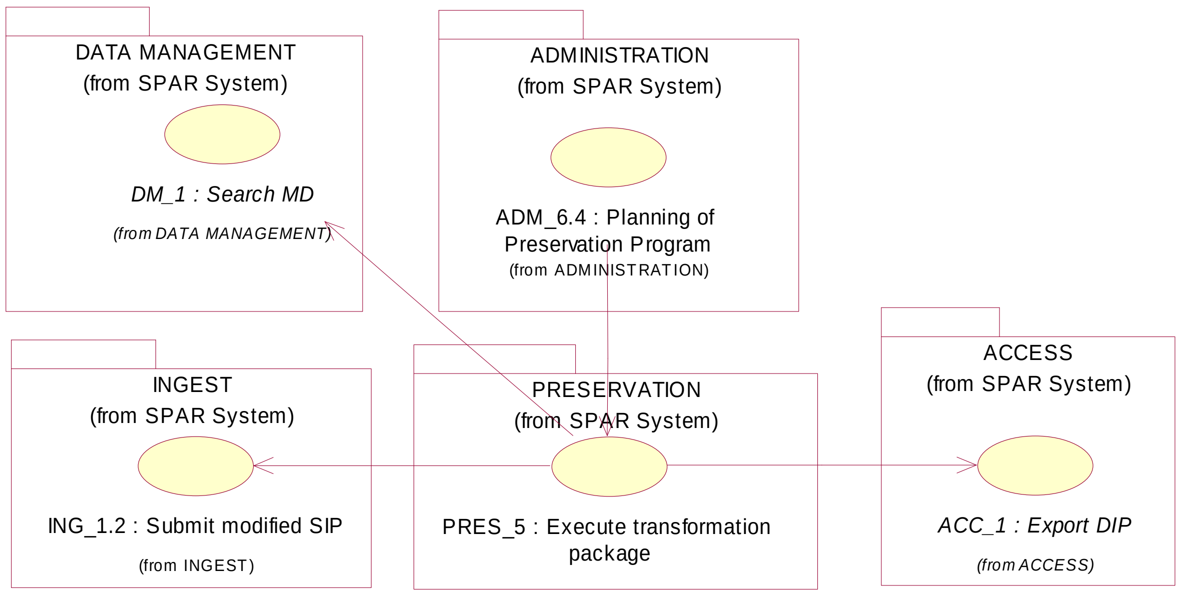
The Preservation module is mainly involved in the following use cases :

- Assemble Information Representation metadata (PRES_1)
- Assemble transformation package metadata (PRES_2)
- Assemble migration program metadata (PRES_3)
- Harvest Information Representation metadata (PRES_4)
- Monitor migration program (PRES_6)

It has only one internal use case:

- Execute transformation package (PRES_5)

The interactions with the other modules are defined in the following diagram:



3.1.3.7.1 Execute transformation package (PRES_5)

Use Case ID	PRES_5
Description	Execute transformation package
Summary	This use case describes how the preservation retrieves a DIP for a migration and deposits it back for its update.
Level	
Actors	Administration module Access module Ingest module
Assumptions	
Pre-conditions	ID of the AIP known Input Information representation of the transformation
Primary functional path	1/ Retrieval of the DIP for migration with the correct input IR (ACC_1) 2/ Transformation of the DIP to a correct SIP 3/ Ingest modified SIP (ING_1.2) 4/ Record of the operation in historical database.
Primary result	AIP updated
Post-conditions	
Exceptional paths	1/ Unknown DIP : a “ID unknown” receipt is sent to the actor 2/ DIP not given: send a trap to the administration. 2/ Failure of the ingest of the SIP: send a trap to the administration

3.2 POLICIES

This section describes the different policies of the SPAR system. As a reminder, those policies are ingested by the system (ADM_3) and they bring a formal definition for the behavior of some modules. The system uses then as needed (cf. computational viewpoint).

3.2.1 INGEST POLICY

This policy defines the services and the quality of service given by SPAR for the ingestion of the SIP. A policy is negotiated by channel in the context of the negotiation between the producer and the SPAR manager. This negotiation is made in the PAIMAS context. The outcome of a negotiation is a formal commitment made in a Ingest Quality of Service Agreement (I-QSA).

The terms of an I-QSA include:

- The SIP structure, including a "METS profile" and a complete description (see the "Detailed Technical Book"),
- The frequency of the delivery,
- The minimum volume of a delivery in bytes,
- The maximum volume of a delivery in bytes,
- The average volume of a delivery in bytes (as an indication),
- The maximum total volume,
- The minimum time to take into account a delivery,
- The maximum time to take into account a delivery,
- The average time to take into account a delivery (as an indication),
- The maximum number of data-object,
- The level of security(authentication),
- The eventual coding algorithm of the transfer,
- The availability of the service: time range of available service (for example, from 2AM to 8PM , from Monday to Friday).

3.2.2 PRESERVATION POLICY

This policy defines the services and the quality of service given by SPAR for the preservation of the AIP. A policy is negotiated by channel in the context of the negotiation between the producer and the SPAR manager. This negotiation is made in the PAIMAS context. The outcome of a negotiation is a formal commitment made in a Preservation Quality of Service Agreement (P-QSA)

The terms of an P-QSA include:

- Retention time (possibly forever);
- Preservation level:
  - Bit stream preservation,
  - Format monitoring (known format required), ,
  - Preservation of the representation of the data-objects (managed format required),
- Assurance level: number of copies, localization of the copies (local or remote),
- Audit frequency.

3.2.3 ACCESS POLICY

This policy defines the services and the quality of service given by SPAR for the dissemination of the SIP. A policy is negotiated by user community in the context of the negotiation between a defined user community and the SPAR manager. The outcome of a negotiation is a formal commitment made in a Dissemination Quality of Service Agreement (D-QSA)

The terms of an P-QSA include:

- The DIP structure, including a "METS profile" and a complete description (see the "Detailed Technical Book"),
- The access context (see annex 4 on G2D),
- The eventual coding algorithm of the transfer,
- The eventual signature of the transferred data,
- The minimum delivery time,
- The maximum delivery time,
- The average delivery time(as an indication),
- The minimum number of request of delivery in a day,
- The maximum number of request of delivery in a day,
- The average number of request of delivery in a day (as an indication),
- The availability of the service: time range of available service (for example, 24hours per day and 7 days/week).

3.2.4 POLICY IMPLEMENTATION

The Quality of Service Agreement define the necessary requirements for the implementation of the policies in SPAR. The administrators of SPAR implement the infrastructure and the necessary parameters of the SPAR modules in order to comply with such requirements. Thus, the performances of the functions in the critical path MUST be configurable. The means they dispose of can be, for instance:

- Configuration of the planning of the transformations,
- Configuration of the priority of the transformations,
- Definition of the levels of service,
- Implementation of the storage units,
- …

So the construction of an ingest channel requires the analysis of every element concerned by the operations to be made. This performance analysis brings to the definition of the different classes of service.

4 INFORMATIONAL VIEWPOINT

4.1 GENERAL DEFINITION OF THE INFORMATION PACKAGES

The information model chosen for SPAR comes from the OAIS norm (see chapter 6). According to the OAIS, the archived information is managed in the system as Information Package.

The received SIP and the given DIP MUST be in the METS format for SPAR. However, the obligation is not restrictive to this only format, other schemas can also be provided.

The structure and the content are described in the "Detailed Technical Books for each channel" as illustrated by the annex 3.

From a physical viewpoint, a Information Package is made of 3 blocks:

- The **Data** part is made of all the data files or data-objects that form the archived digital object,.
- The **Metadata** part follows a XML formalism in which the information describing the archived digital object are written,
- The **Packaging** part wraps the metadata and the data, making what is called a Information Package..

4.1.1.1 THE DATA-OBJECTS

The data-objects, targets of the archival, are digital files or bit-streams.

4.1.1.2 THE METADATAS

In the Information package, the data-objects are described by metadata, needed for their identification and the right application of the preservation transformations.

In SPAR, these metadata are contained in a XML file, conforming to the METS schema (version 1.6). This file is called **METS manifest**.

In conformance with the OAIS norm, we distinguish between two kinds of metadata :

- **The Representation Information:** the goal of the Representation Information is to map a data object made of a bit stream to an meaningful form. The elements given by the Representation Information apply to the structuring of the bit stream or data format. They give also indication at the semantic level, as the language of a text. The association of the data object with its Representation Information makes the conceptual level of the Information Object. In the METS manifest, the Representation Information about the structuring of the bit streams are contained in one or many technical metadata blocks (in the `<amdSec>` section). The language information is contained in the descriptive metadata block (in the `<dmdSec>` section).
- **The Preservation Description Information:** the Preservation Description Information is the one necessary for the good preservation of the Information Object. It gathers the information of reference (identification), provenance, context and integrity. In the METS manifest, the reference information are contained in the descriptive metadata blocks `<dmdSec>`. The provenance information is contained in one or many technical metadata blocks (section `<amdSec>`, subsection `<digiprovMD>`).

4.1.1.3 THE PACKAGING

The packaging information is the information that, in real or logically, gathers the parts of the Information Package into a identifiable entity for storage or transmission goals. It connects the many parts (metadata and data objects) and allows the reconstruction of the physical or logical structure of an archived object.

In SPAR, the packaging information is contained in the METS manifest.

4.1.1.4 THE CONTAINER FORMAT

The container format binds the other formats of data. The chosen format for the binding of the metadata of the Information Package is METS (Metadata Encoding and Transmission Standard).

The choice of a format for the binding of the data-objects as well as the METS manifest itself is left to the contractor who will explain its choice in his technical report.

4.2 THE GRANULARITY LEVELS

At BnF, **four** levels have been defined for the granularity of the digital objects. They allow to categorize the logical structure of the Data Objects contained in the Information Packages. In the METS manifest, the logical structure is defined in the metadata block `<structMap>`. Each block is associated with one of these levels..

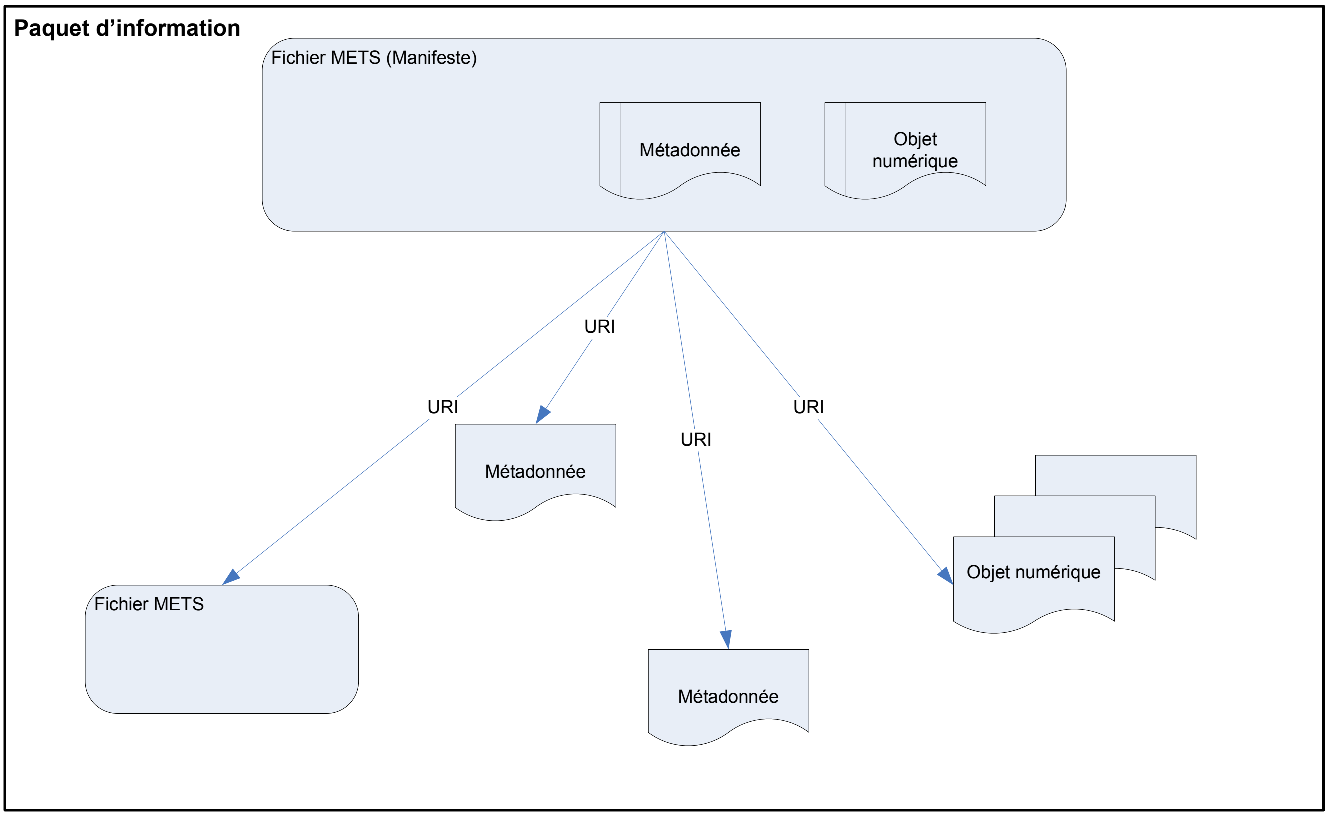
- **Set** : the "Set" entity is used for the Packages of collections of digital objects, for example a whole collection of serials or a multimedia kit. If the metadata corresponding to the Set level are included in an other Package, of AIC type, the reference to these metadata will be made through a link (weakly-linked metadata).
- **Group** : the "Group" entity is used for the Digital Object kept in the Package. It can be a monograph, an issue of a serial, a batch of images, a particular movie..
- **Object** : the "Object" entity is used for an element of a digital object as an particular image in a batch of images or a specific page of a textual document, a sound track for a audio CD.
- **File** : the "File" entity corresponds to the data-object (digital file or bit-stream).

4.3 BUILD OF THE INFORMATION PACKAGE

An Information Package is made of a whole of data objects stored in one or more files and a whole of administrative, descriptive and technical metadata..

An Information Package is characterized by a manifest which contains the description of the structure of the Information Package. It contains the description of the packaging information as defined by the OAIS model. It can also contain content information as well as preservation description information (PDI).

The format retained for the manifest of the SIP and DIP Information packages is the METS format. The contractor can propose other formats in addition. If necessary, it will provide complete documentation (format, schema, description, data dictionary, etc.) of the proposed formats.



Some metadata have to be stored with the data objects (see. § « concept of weakly/strongly linked metadata). In this event, the METS format offers two ways of implementation:

- either use an URI referencing a local resource (ex : file:///monfichier.ext) : in this case, the path MUST be relative;
- either include the metadata in the METS file (encapsulation)

The contractor will choose in its response the more appropriate solution in its opinion.

4.3.1 DEFINITION OF THE FORMATS OF THE PACKAGES FOR THE "INGEST" MODULE

The SIP accepted by SPAR MUST be at least in METS format.

The description of the SIP will be based on a "METS profile" as defined at the following address «http://www.loc.gov/standards/mets ».

A METS profile defines in a narrative way the structure and the content of METS files. It will be a sum up of the specifications defined in the present report. Several METS profiles could be given:

- a general profile defining the structure and the content of all the SIP.
- a specific profile for each channel adding to the general profile the specificities of the channel. It will be directly derived from the Detailed Technical Books.

4.3.2 DEFINITION OF THE FORMATS OF THE PACKAGES FOR THE "ACCESS" MODULE

The DIP given by SPAR MUST be at least in METS format.

The description of the DIP will be based on a "METS profile" as defined at the following address «http://www.loc.gov/standards/mets ».

A METS profile defines in a narrative way the structure and the content of METS files. It will be a sum up of the specifications defined in the present report. Several METS profiles could be given:

- a general profile defining the structure and the content of all the DIP.
- a specific profile for each channel adding to the general profile the specificities of the channel. It will be directly derived from the Detailed Technical Books.

4.3.3 DEFINITION OF THE FORMATS OF THE ARCHIVED PACKAGES

The AIP managed by SPAR are preferably in the METS format. However, the contractor can propose another formalism, for example FOXML from Fedora.

The description of the AIP will be based on a "METS profile" as defined at the following address «http://www.loc.gov/standards/mets ».

A METS profile defines in a narrative way the structure and the content of METS files. It will be a sum up of the specifications defined in the present report.

If the contractor doesn't choose METS as AIP format, he will describe the correspondence between the description of the AIP and the format he chooses. Moreover, he will give a tool allowing the extraction of the AIP in the METS format.

The detailed definition of the AIP is described for each channel in its Detailed Technical Book.

4.4 DEFINITION OF THE DATA MODEL FOR THE METADATA

4.4.1 DATA MODEL FOR THE REPRESENTATION INFORMATION

The data model of representation defines the representation information and its network of representation information of the OAIS model. Mainly, it's the most complete description of the components of a format:

- Structure: description of the organization, definition of the records, structure of the data, etc.
- Format of the data: primitives used for the structures of the data,
- Algorithm of compression/decompression of the data
- Bit orientation

It contains also the description of the relations with other formats (dependency, inclusion, inheritance, …) and the software and hardware environment necessary for the use of the described format. These information is used for:

- Selection, i.e. to search a format able for a particular use,
- Identify and characterize, i.e. to find the format and to give all its characteristics,
- Validation, i.e. to verify that the characteristics of a format correspond to those awaited for a particular use,
- Evaluation, i.e. to determine the risks of the use of a format in a given context to select or process them, ,
- Process, i.e. to realize the transformations or emulations in order to perpetuate or disseminate.

The model currently chosen is the model made by the GDFR network. The reference documentation is GDFR-Data-Model-5_0_5.rtf . the chosen version is the version *Data Model*, v.5.0.5 [October 25, 2006][5].

4.4.2 DATA MODEL FOR RIGHTS MANAGEMENT

4.4.2.1 ACCESS AND USAGE RIGHT

4.4.2.1.1 Rights metadata

To each digital object, will be associated a file of rights metadata containing factual information necessary to compute in a automatic way the communicability status of a document.

The chosen model is a XML model which schema will be given by the BnF at the beginning of the contract.

An example of such rights metadata file is given in appendix 3.

4.4.2.1.2 The context of access

The context of access is given by a *physical localization* (access in search rooms, in study rooms, etc) and by an *access profile* (known searcher, professional, etc.)

4.4.2.1.3 The decision trees

The data model for the decision tree defined a path to follow depending on the presence or not of certain metadata as well as the values of those metadata.

An example of such decision tree is given in Appendix 3.

[5] The applicable version of this model will be given with the emission of the purchase order of the Preservation module. The reference documentation given here allows the measurement of the importance and the coverage of this model.

The model chosen for the decision trees is a XML based model. The contractor, in its response, describes how these decision trees are defined.

4.4.2.1.4 The usage conditions

The files of usage conditions contain information related to the authorized actions on the document, depending on the context of access. It can be, for example, actions allowed on digital objects in "Research" context of access.

The model chosen for these usage conditions is ODRL (see bellow).

4.4.2.1.5 The agreements

The agreements contain the information related to the authorized actions on collections of digital objects for which it exists an agreement between the BnF and the holder of the rights. For example, the BnF signed an agreement with CNRS editions to allow the dissemination of their digitized documents inside the library.

The link between the agreement and the digital object is made through the rights metadata.

The model chosen for the agreements is ODRL (see below).

4.4.2.1.6 The license

The license is the response of the "Rights management" module to the request it receives. It comes from:

- an agreement

or (exclusive)

- a computation made on the rights metadata according to the rules defined in a decision tree, including the usage conditions.

The model chosen for the License is the model given by ODRL. The reference documentation is available at the following: http://odrl.net/. The chosen version is 1.1 version.

The license defines the rights <o-ex:permission> for a digital object or part of a digital object < o-ex:asset> in a context of access < o-ex:party>. A right <o-ex:permission> is declined in one or several elements categorized in the « Data Dictionary », in particular:

- <o-dd:display> for display
- <o-dd:print> for printing
- <o-dd:save> for recording

Moreover, constraints <o-ex :constraint> categorized by the « Data Dictionary » can be added, for example to constraint to 5 printings, the following declaration is needed:

```
<o-dd:print>
         <o-ex:constraint>
               <o-dd:count>5</o-dd:count>
         </o-ex:constraint>
</o-dd:print>
```

An example of an ODRL license is given in Appendix 3.

4.4.2.2 AUTHENTICATION

The model for authentication defines the access control, i.e. the rights of handling given to the possible actors on the system. It applies to the administrators of the system, the user profiles of the different modules : ingest, storage, access, preservation, etc.

The model chosen is the model given by XACML from OASIS (`http://www.oasis-open.org/committees/xacml/`).

4.4.3 DATA MODEL OF PRESERVATION

The data model of preservation defines the preservation description information (PDI) of the OAIS model. These information are related to the lifecycle of the digital object:

- the reference information gives identification data for the digital object,
- the context information gives data on the environment for the creation/update/deletion of the digital object : agent responsible of the operation, credentials for the operation, links to other digital objects, …
- the provenance information gives data on the sources of the information contents (digital object + representation information): how and from what the objects have been created/harvested/updated, logs of all the operations made on the digital objects.
- the fixity information give data ensuring to the information contents have not been faded : digital stamp, checksum, signature, …

The common model chosen is the model given by `PREMIS`, to which are added specific data model per kind of data-objects, as specified by the Detailed Technical Books.

For example, in the annex 3, we use `MIX` for image data-objects and `textMD` for textual data-objects.

4.4.4 MODEL FOR DESCRIPTIVE DATA

The model for descriptive data defines the package description of the OAIS model. These information cover bibliographic information as title, author, editor, edition data, etc.

The chosen model is the Qualified Dublin Core model. The reference documentation is available at the following URL: http://dublincore.org/

The descriptive data for each Information Package are contained in the METS manifest, in a block of the `<dmdSec>` section, for each of the entities described in the `StructMap`.

4.4.5 MODEL OF MANAGEMENT FOR THE TRACEABILITY OF THE ACTIONS

The confidence given to an OAIS model by users and managers of patrimonial collections and archives rests mainly on the transparency given by all the preservation actions, i.e. either the actions carried on the media either those carried on the AIP. It is thus appropriate to log systematically all these actions. Let us note that the actions carried on the AIP are also logged in the PDI of each AIP (context and provenance information). The actions carried on the media MUST be logged in the « Storage » module.

4.4.6 DATA MODEL FOR THE RESULT SETS

The result sets are those in response to a search for digital objects or in response to an harvesting of metadata through the OAI-PMH protocol. The OAI-PMH protocol is used in version 2. The reference documentation can be found at the following URL: « http://www.openarchives.org/ ».

In every case, the result set is given in a response conforming to the OAI-PMH protocol. For example, the search for the documents written by an author gives a response of <ListRecord> type. The format MUST be at least un non qualified Dublin Core, as defined by the OAI-PMH protocol. However, after a negotiation with the use community, other complementary formats can be added.

The result set contains descriptive information (title, author, etc.) specific to the search or harvesting made. However, it MUST always contain an identifier `dc:identifier`. This identifier MUST allow the access to the DIP corresponding to this record.

4.4.7 DATA MODEL FOR THE LEVELS OF SERVICE

4.4.7.1 LEVEL OF SERVICE FOR THE INGEST

The data model for the levels of service for the ingest defines:

- Minimal time to taking in account
- Maximum time to taking in account
- Minimal size of a SIP
- Maximum size of a SIP
- Maximum number of data-objects in a SIP
- Security level (authentication)
- Time slices of service availability
- Ingest frequency
- Encryption (allowed or prohibited)
- Kind of encryption (used algorithm, length of coding)

The above model should give enough information to allow the contractor the measurements and coverage of the model. He must explicit in his response how he will implement it.

4.4.7.2 LEVEL OF SERVICE FOR THE PRESERVATION

The data model for the levels of service for the preservation defines:

- Time of retention
- Maximum time of archival or availability
- Number of copies
- Maximum size of an AIP
- Availability in percentage over a year (ex : 90%) and in maximum time interval (ex : no more than 10 hours per month)
- Audit frequency
- Hardware compression (allowed or prohibited)
- Encryption (allowed or prohibited)
- Kind of encryption (used algorithm, length of coding)
- Kind of writing (RW or WORM)
- Kind of media (hard disk, tape, optical disk, MO, etc.)[6]

The above model should give enough information to allow the contractor the measurements and coverage of the model. He must explicit in his response how he will implement it.

4.4.7.3 LEVEL OF SERVICE FOR THE ACCESS

The data model for the levels of service for the access defines:

- Maximum time of supply of a DIP
- Deadline of generation of the DIPS (in hours)[7]

---

6 Some standards impose the use of a particular kind of media (in particular standard NF Z42-013 imposes the use of WORM optical disk).

7 For example, obtaining the master version implies fresh generated DIP.

- Availability in percentage over a year (ex : 96%) and in maximum time interval (ex : no more than 10 hours per month)
- Signature (mandatory, optional,…)[8]
- Encryption (allowed or prohibited)
- Kind of encryption (used algorithm, length of coding)

The above model should give enough information to allow the contractor the measurements and coverage of the model. He must explicit in his response how he will implement it.

[8] This information is directly linked with the third party archival. I twill thus be implemented during the realization of this particular channel.

4.5 DEFINITION OF THE IDENTIFIERS

The system manages three sorts of identifiers:

1. the internal identifiers which identifies in the SPAR system itself the information packages,
2. the access identifiers,
3. the external identifiers generated by other systems..

4.5.1 THE INTERNAL IDENTIFIERS OF THE INFORMATION PACKAGES

Each information package owns a unique identifier. It concerns the identification of the packages by SPAR itself. Each version of a package has its own identifier. The means of generation MUST allow distant and distributed components to create identifiers without the need to test for duplicates. Their format is not specified and depend on the specific implementation.

However, we proposed to use the Universal Unique IDentifier UUID as defined by the RFC 4122. The definition is a hexadecimal sequence of 36 digits, for example :

```
6ba7b810-9dad-11d1-80b4-00c04fd430c8
```

To minimize the risk of collisions, it can be used the « name-based » version based on the digital digest (MD5 or SHA1) of the information package.

The contractor will explicit the chosen mechanism in its proposal.

4.5.2 THE ACCESS IDENTIFIERS

The access identifiers guarantee the presence and the access to a digital object in SPAR. They allow thus to an external actor to make reference to a digital object in a permanent way. Each AIP has a permanent identifier. The change of edition or version doesn't make any change on the ARL identifier.

The kind of permanent access identifier chosen is the ARK format, as said in chapter 6. The reference documentation is available at the following URL « http://www.cdlib.org/inside/diglib/ark/arkspec.pdf ».

4.5.3 THE EXTERNAL IDENTIFIERS

The external identifiers are the identifiers generated by systems external to SPAR which are kept for historical reasons.

We can hold up mainly the identifiers defined by the production process, as well as ISSN, ISBN or record number from BnF catalog. Their format is not controlled.

The production identifiers are given by the producers themselves. They are used to prevent the ingest of SIP in double.

Each ingest channel has an identifier of URI kind (see. RFC 2396), defined in the appropriate Detailed Technical Book..

4.6 SPECIFIC DEFINITIONS FOR THE AIP

4.6.1 LIFECYCLE OF THE AIP

The version of an AIP created for the first time is called 'V0'. This version is NEVER destroyed[9]. Then during the lifetime of an AIP changes may occur. We mean by "change" any addition, update or deletion. These changes imply either the creation of a new edition or the creation of a new version. We shall notice that this is the same definition found in the OAIS model under « Distinguishing AIP versions, editions and derived ». It can be referred to this chapter for a better understanding. In our system, the case of derived AIP is not taken into consideration.

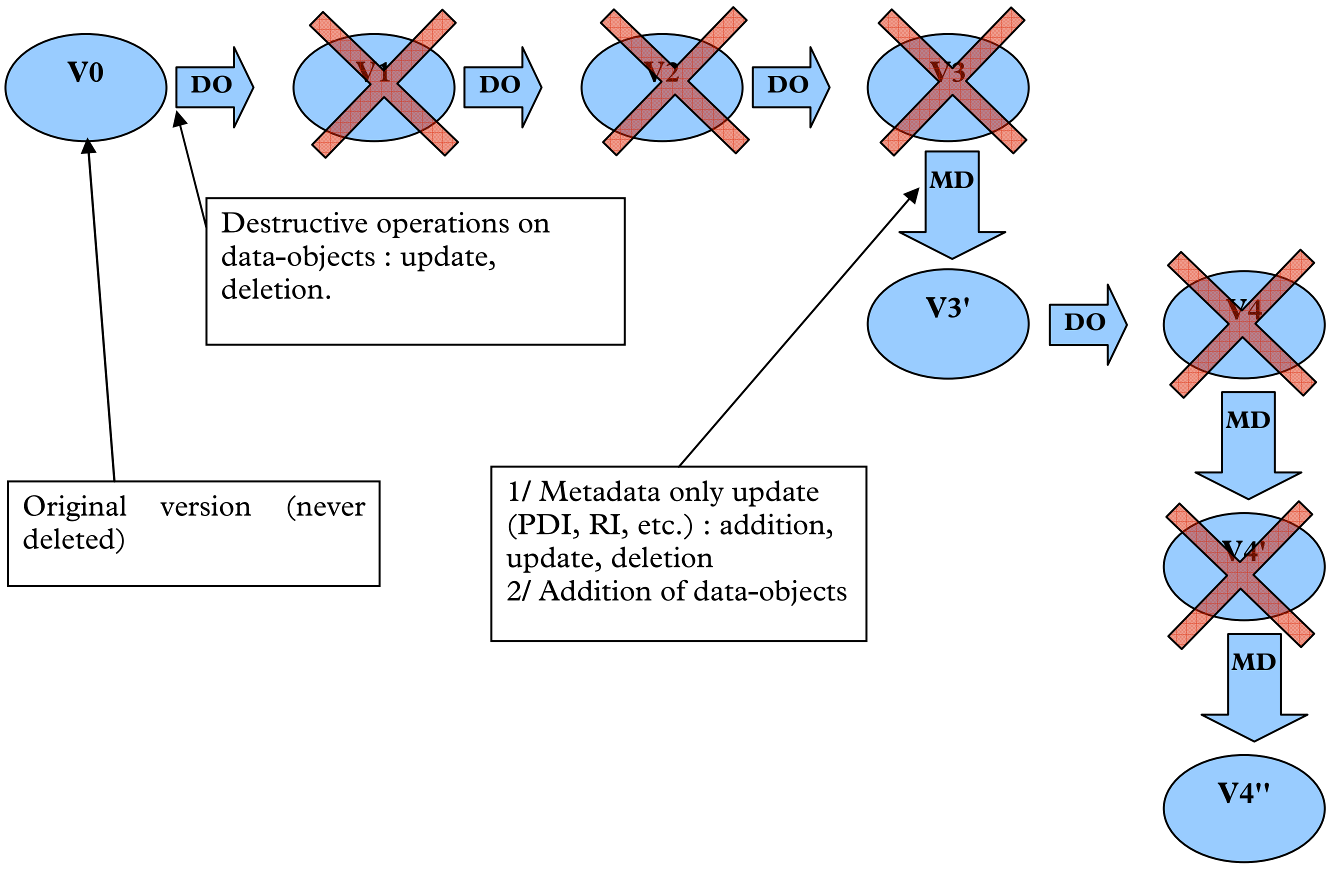
The editions cancel and replace the previous edition. A new edition is created when a change affects metadata or in the case of an addition of data-objects. The metadata gather the representation information, the PDI and the descriptive information.

The two last version are kept in addition to the original V0 version. A new version is created each time an update or a deletion is made on at least one data-object.

In certain circumstances – administrative archival or third party archival – the deletion of an entire AIP is required. An AIP is never completely deleted from SPAR, only the information content (representation information and data-objects) are deleted, the PDI is preserved.

All the changes are logged.

As an example, the following diagram show an AIP with multiple versions and editions:


V0
DO
V1
DO
V2
DO
V3
MD
V3'
DO
V4
MD
V4'
MD
V4''
Destructive operations on data-objects : update, deletion.
Original version (never deleted)
1/ Metadata only update (PDI, RI, etc.) : addition, update, deletion
2/ Addition of data-objects

[9] Except in the case of the withdrawal of an AIP.

4.6.2 AIP HIERARCHY

The OAIS model, in the chapter « Specialization of the AIP and Package Descriptions », presents the notions of AIC and AIU. In our case, the "set" level implies the creation of an AIC and the "group" level defines an AIU.

It should be noticed that the "set" level is characterized with descriptive information about its granularity level, for example the title of the collection, the editor of the collection, etc. It usually doesn't contain data-objects per se. Thus, in the "Preservation digitization" channel, with the METS format, an AIC contains only external references to other METS.

Moreover, usually the elaboration of the collection of AIU depending on an AIC is made in several times, i.e. coming from the successive ingests of multiple SIP. The way of building them is described in each Detailed Technical Book.

4.6.3 NOTION OF STRONGLY/WEAKLY LINKED METADATA

Conceptually, the metadata are all completely part of the information package. However, the implementation of this principle can bring difficulties as loss of storage space by excessive redundancy of information, complication in the update on many different objects located in various storage spaces, growth in search time, etc. For all these reasons, it's interesting from an implementation point of view to gather up some of these metadata.

Thus, the metadata that can gathered don't need to be stored with the digital objects but can be defined by reference using the link mechanism provided by the packaging format. In the case of METS packages, the link is build through an URI to a METS file which is not directly present in the same information package. These metadata are called "weakly linked".

On the other hand, some metadata – considering their criticality or because gather up is of no use – must be stored with the digital objects Here by "store with" we mean store in the same directory, in the same TAR file or ZIP file, i.e. so that the metadata are directly accessible without using any search mechanism. Those metadata are called "strongly linked".

Thus, an AIP is made of:

- strongly linked metadata which are contained in the AIP. They are provided by the SIP,
- weakly linked metadata which are referenced by an ARK identifier. They are provided, in the SIP, as reference metadata,
- data-objects contained in the AIP.

5 COMPUTATIONAL VIEWPOINT

5.1 GENERAL DESCRIPTION OF SPAR MODULES

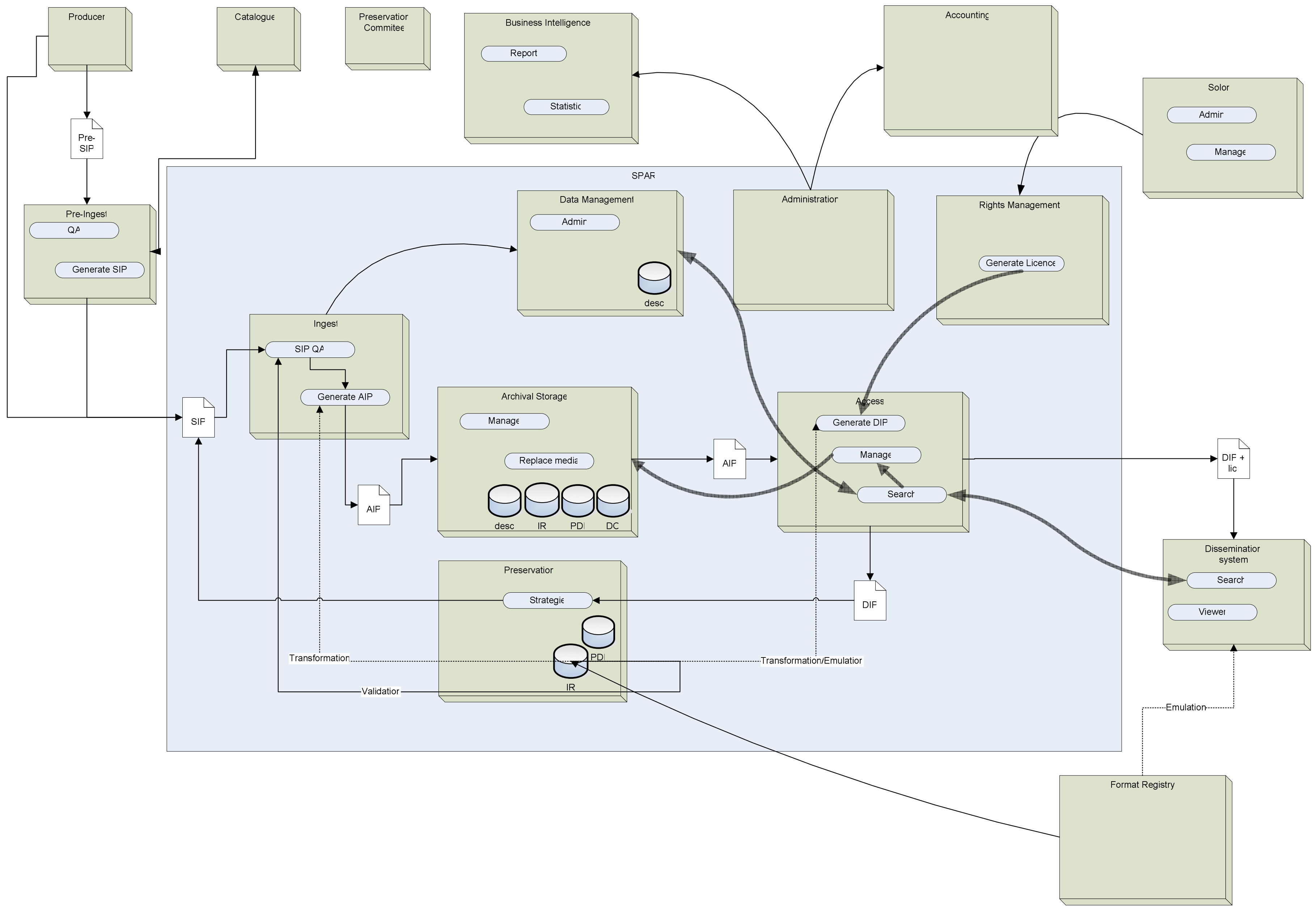
The architecture is based on a distributed model. It's broken up into 7 principal modules:

- Ingest
- Archival storage
- Data management
- Rights management (G2D)
- Access
- Administration
- Preservation

Each module is instantiated on one or several independent nodes which communicate with the others through interfaces. Each module is itself broken up in consistent functionalities. The goal of this part is to describe completely the expected functionalities by each of the modules and the interfaces of communication between the modules. The conception made by the contractor must guarantee the correct behavior of the services and interfaces of a module independently from the other modules.

Moreover, in order to answer the perpetuity demands, a storage abstraction service is defined to ensure the independence of the system against the Storage Infrastructure.

The diagram in the next page shows the modules and their interactions.

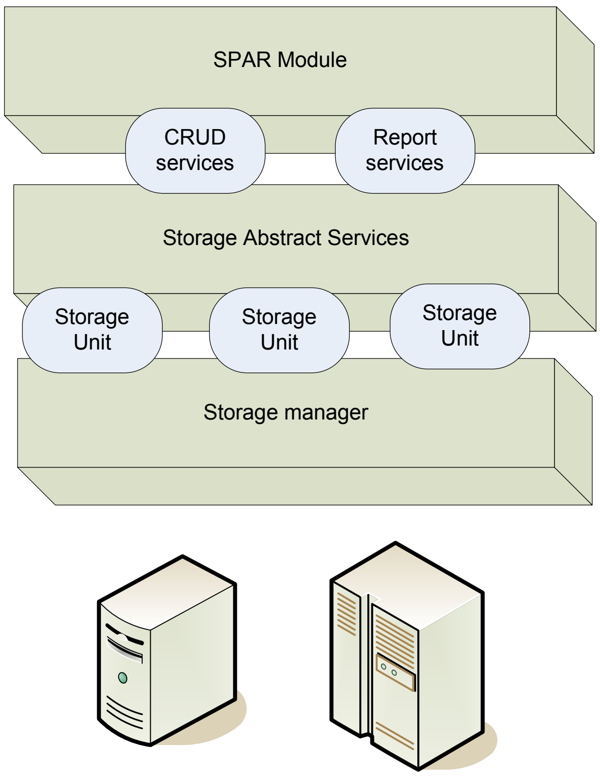

Producer
Catalogue
Preservation Commitee
Business Intelligence
Report
Statistic
Accounting
Solor
Admin
Manage
Pre-SIP
Pre-Ingest
QA
Generate SIP
SPAR
Data Management
Admin
desc
Administration
Rights Management
Generate Licence
Ingest
SIP QA
Generate AIP
SIP
Archival Storage
Manage
Replace media
desc
IR
PDI
DC
AIP
AIP
Access
Generate DIP
Manage
Search
DIP + lic
Dissemination system
Search
Viewer
DIP
Preservation
Strategie
PDI
IR
Transformation
Transformation/Emulation
Validation
Emulation
Format Registry

5.2 THE STORAGE ABSTRACTION SERVICE

The storage management must be carefully designed especially in relation with perpetuity.

Thus, the goal of this service is to ensure that the SPAR modules (especially the Storage module) see the infrastructure through abstract storage elements : the **storage units** ("capsules de stockage »). Those storage units lean on **storage managers** ("logiciels de pilotage") of the elements of the storage infrastructure.

This layer break up is shown is the following diagram :



The « SPAR module » layer represents one of the module of the SPAR system which must use the storage infrastructure:

- the ingest module for the temporary storage of the SIP,
- the preservation module for the temporary storage of SIP/DIP,
- the access module for the cache of DIP,
- the storage module for the storage in perpetuity of the AIP.

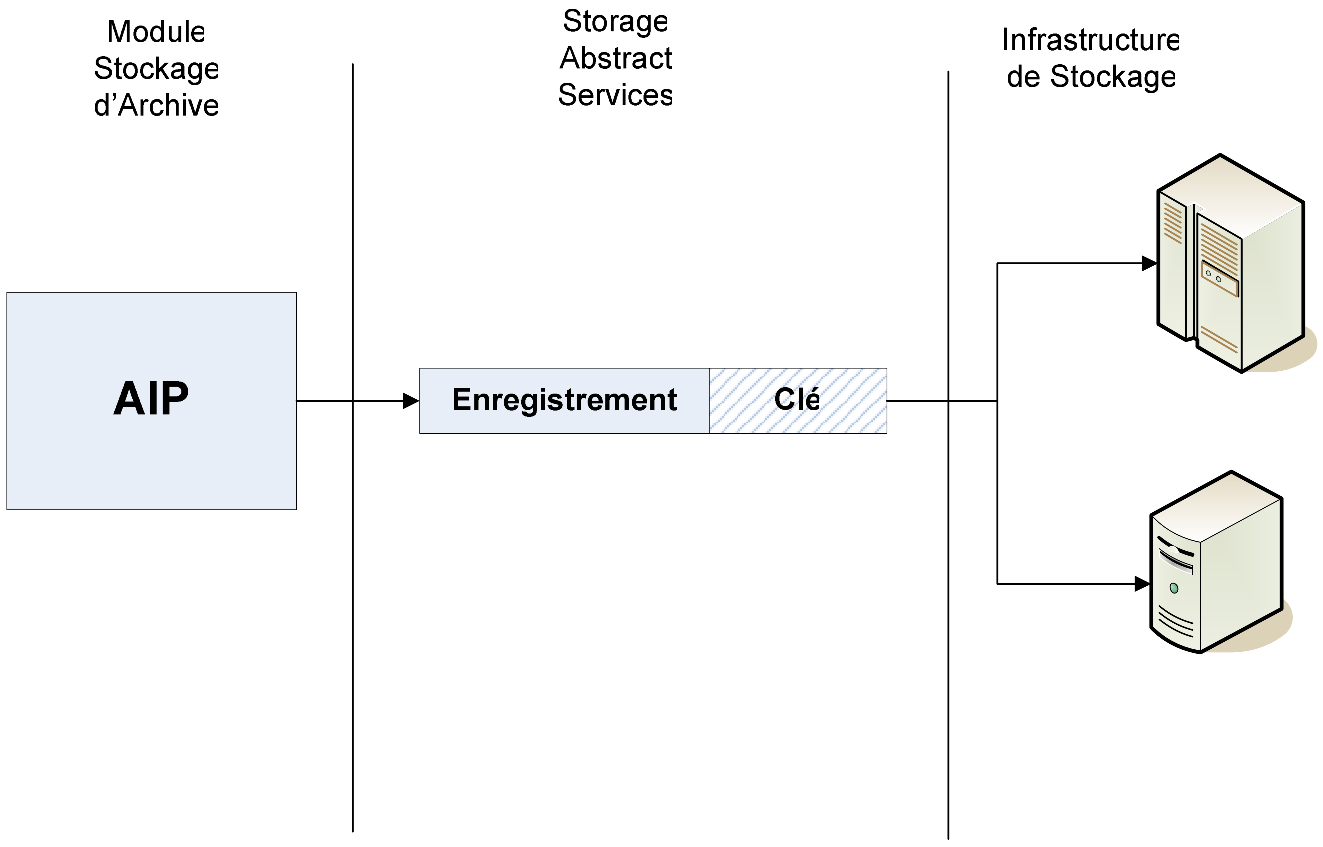
These modules have the responsibility of satisfying certain level of service. With this intention, it uses the adequate storage unit among those available. It can then restrict itself to manage packages at the SIP, AIP or DIP level.

The abstraction layer "Storage Abstract Service (SAS)" offers services to manipulate storage unit. It leans on the specific « storage manager » to each element of storage that it abstracts in a *storage unit*. It's a complete module, seen through the internal services of the system. The other modules MUST use the SAS module to access the data.

The contractor will use the definition given in this section to propose a complete solution for the SAS layer.

The « storage manager » layer includes the drivers allowing the real handling of the storage elements. It manages files or bit-stream called **records** ("enregistrements").

To illustrate, here is how an AIP crosses the layers during its storage:


Module Stockage d'Archive
Storage Abstract Services
Infrastructure de Stockage
AIP
Enregistrement
Clé

The storage abstraction service must allow the use of all the elements making the Storage Infrastructure as described in chapter 7. TO allow its use during the development phase, a mock mode must be available on the "Development platform". In the same way, to make the tests easier, a version compatible with the "Verification platform" (in mock mode for unit testing and in complete mode for integrated testing) must be available.

The contractor can choose:

- either to use the drivers shown in the chapter 7 describing the actual storage infrastructure (in which case, it will describe how it answer the requirements of this chapter),
- or to propose another solution (in which case, the derived costs must be included in the firm phase according to the CCAP).

5.2.1 STORAGE UNIT DEFINITION

A storage unit is a conceptual entity representing a storage element, a set of storage elements or a part of a storage element. A storage element is a storage system associated with the needed drivers in order to provide storage services in a specialized network[10] through the interface of the storage unit. A storage unit is defined by a set of characteristics and services.

For example, a storage unit can be made of a pool of tapes of the same type in a library, associated with a logical partition of a disk array.

Each storage unit is seen by the SPAR modules in the identical way and presents the same interface.

An interface allows the command and the management of the storage unit. The storage units manipulate and manage **records**. Hence, the modules only see records characterized by their internal identifier and not the elements of the Storage Infrastructure (disks, tape library …).

5.2.1.1 CHARACTERISTICS

A storage unit has the following characteristics:

- minimum, mean and maximum time for the creation of a record
- minimum, mean and maximum time for the retrieval of a record
- minimum, mean and maximum time for the update of a record
- minimum, mean and maximum time for the deletion of a record
- minimum, mean and maximum time for the seek of a record
- flow rate (in bits/second)
- kind of storage (permanent or cache)
- kind of offered copy (synchronous or asynchronous)
- number of simultaneous copies
- minimum, mean and maximum time for a copy
- availability in percentage over a year (ex : 96%) and in maximum temporal unit over a fixed period (ex : no more than 10 hours per month)
- assurance level of the associated media (probability of failure of the media during a year of functioning)
- total capacity (in bytes)
- maximum size of a record (in bytes)
- maximum number of records
- hardware compression (active or not)
- encryption (active or not) [11]
- kind of encryption (used algorithm, length of encoding)
- kind of writing (RW or WORM)
- kind of media (hard disk, optical disk, MO, etc)
- kind of integrity (md5, crc32…)

These characteristics are the characteristics really measured. They are defined by administrators through the Administration module.

[10] For security reasons, it's a specific and independant (either physically or virtually) network.

[11] This characteristic is related to the third party archiing. It will then be implemented during the realization of the given channel.

5.2.1.2 States

A storage unit has a current state defined by:

- available capacity (in bytes)
- number of records
- state (active, inactive, busy, unavailable, error…)
- running operations (requestor identification, number, kind, state…)

5.2.1.3 Exposed services

A storage unit must expose the following services:

- Create a record (*)
- Retrieve a record (*)
- Audit a record (*)
- Update a record (*)
- Delete a record (*)
- Seek a record (sort, filter) (*)
- Lock a record
- Unlock a record
- Modify the characteristics of a record
- Give the characteristics of a record
- Notify an event
- Monitor a service (*)
- Cancel a running service

(*)

1. Each service must allow the definition of a priority level (high, medium, low).
2. Each service is transactional (ACID service).
3. Each service can be synchronous or asynchronous. In the latter case, the use of "Monitor a service" service allows the monitoring of the execution.

The « Seek a record » service allows the search of records using the characteristics of a record.

The « Retrieve a record » service is asynchronous and should answer all the queries. The integrity control key is always verified. In case of a huge demand, a storage unit can be set to the « unavailable » state.

The « Audit a record » service verifies the integrity control key of all the copies of a record. In case of an error, the record changes its state and a trap is sent to the Administration module.

The « Notify an event » service sends a trap to the Administration module. It can be the case of a breakdown, an error, an unsatisfied level of service, etc.

Moreover, services to describe or discover storage units exist:

- List the storage units
- Give the characteristics of a storage unit
- Current state of a storage unit
- Declare/Suppress a storage unit (*)

- Activate/Inactivate a storage unit[12] (*)
- Modify the characteristics of a storage unit (*)

(*) Service reserved to administrators

The “List the storage units” service gives the list of all the available storage units with their state.

The “inactivate a storage unit » doesn’t stop immediately all the operations going on but stops any new operation. This way, all initiated operations is completely realized.

5.2.1.4 LOCK MANAGEMENT

Given the distributed nature of the elements making an AIP (see informational viewpoint), some functions (store an AIP or audit an AIP) need to guarantee the availability and the coherence of the data in the records. This guarantee is brought by the lock mechanism. Two kinds of lock exist:

- Read lock: ensure the availability of a record
- Update lock : disallow any other update operation on the record

To avoid dead locks, a lock is released after a period, defined by the administrator, has expired. The expiration delays depend on the kind of locks.

5.2.1.5 SATISFACTION OF LEVEL OF SERVICE

A storage unit allows the implementation of the levels of service. If, due to an unrecoverable failure of a particular element of storage or during the audit of the records, the storage system is not able any more to ensure the level of service, an operation made by the administrators is necessary. The storage unit MUST be able to send alarms to the monitoring systems as well as log the failure. The satisfaction status of the level of service is downgraded. The necessary actions should then be accomplished in order to restore the adequate level of service.

5.2.1.6 DETAILS ON THE MIGRATIONS

The OAIS standard distinguishes four kinds of migration. The migrations « refreshment » and « replication » influence only the media with no impact on the AIP. Those migrations MUST be taken into account by the SAS in a transparent manner for the read/write operation of the module layer.

The « refreshment » migrations are transfers of information from a media to an identical media. It occurs in a planned manner in the context of a migration plan or in an automatic manner when a critical threshold is detected : number of retentions, number of error blocs, etc.

The « replication » migrations are transfers from a media to a possibly different media. The major difference with the « refreshment » migration is that, in this case, the infrastructure of the storage unit has evolved.

Those migrations are internal operations inside a storage unit. However, each of those actions are logged and notified to the Administration module.

5.2.1.7 INTEGRITY CONTROL FUNCTION

The function of integrity control provides the insurance that no one record has been corrupted by the storage unit. This guarantee is implemented through the use of a integrity control key (i.e. CRC32).

Various kinds of keys can be used. The key is given by the storage unit. The validation of conformity of the key is part of the creation process of a record. The storage unit MUST execute the verification of this key during any of the operations of migration or supply of the records.

[12] This allows or disallows any new connection

5.2.1.8 MANAGEMENT OF THE ATTRIBUTES OF THE RECORDS

A storage unit must be able to manage specific attributes. These attributes are of simple type : name, category, size, … These attributes must be customizable (add/update/delete).

5.2.1.9 MEDIA MONITORING AND STATISTICS

The monitoring of the media results in the analysis and the periodical control of the media to verify their degradation status.

The prevention consists in starting automatic operations of refreshment or replication of the media when predefined thresholds are reached.

Various monitoring means are offered by the storage manager layer :

1. Use the functions of error detection made available by the hardware, i.e. to be able to analyze the use of the algorithms of error correction[13], to follow the tape retentions,
2. Implement control processes using the integrity control key of each record.

When predefined thresholds are exceeded, this function:

- Sends alarms to the Administration module,
- Starts automatic operations of refreshment or replication supplied by the storage manager layer (storage infrastructure).

5.2.2 DEFINITION OF A RECORD

A record is an elementary unit (one piece) manipulated and managed by a storage unit. The size of a record can vary from a few bytes to hundreds of gigabytes.

A record is characterized by:

- identifier
- Keep period (possibly infinite)
- Size (in bytes)
- Habilitation[14]
- Integrity control key
- Attributes

Moreover, a record has the following states:

- OK
- Lock (read or update)
- Downgrade, if at least one copy of the record is in error
- Error, if all the copies of a record are in error

5.2.3 THE LEVELS OF SERVICE

The requirements in terms of performance and storage of the AIP are characterized through specific attributes called “level of service”. The attributes of a level of service turn into reality the requirements of the QSA (Quality of Service Agreement) and allow SPAR to attribute the appropriate storage unit.

[13] As Reed-Salomon
[14] In the sense of rights on a filesystem

A level of service is defined as the combination of a performance guarantee, a storage guarantee and the availability of the data in time. A maximum size file is defined for each level of service. The available levels of service are dependent on the intrinsic qualities if the storage units of the storage system.

5.2.3.1 PERFORMANCE GUARANTEE

"Performance guarantee" means the maximum time needed to make available a given file on the storage system when a request of retrieval is sent.

The maximum time of making available corresponds only to the time of identification of the request and to the time needed to execute the processes needed to retrieve an AIP from its current place in the hierarchy of the media until the beginning of the file transfer on the network to the system that has emitted the request, ignoring the time to transfer the data on the network external to the storage system.

The choice of the "Performance guarantee" has a direct impact on the specific allocation of the information package.

5.2.3.2 STORAGE GUARANTEE

"Storage guarantee" means the level of security of the data that the user through SPAR wishes to give to the information package.

To be noticed: the media used to store a file comes from the "Performance guarantee" asked by the user.

The "Storage guarantee" depends on the number of copies and on the location of the media available on the storage system.

The level of service points to the needed number of backup copies to store.

The backup copy can be a copy of a mirror of an AIP on a media not directly accessible, so that the latter can be rebuild in case of a failure on the media that stores it.

The guarantee means that at any given time the user is ensured, the associated level of service been chosen, of the existence of the right number of copies asked for.

5.2.3.3 PERFORMANCES OF THE STORAGE HARDWARE

The list of the levels of service is elaborated to cover a extensive spectrum taking into account :

- The needs expressed in terms of archiving./retrieval (criticality of the files, possible simultaneity of the requests, …),
- The storage hardware that are currently available.

The levels of service are defined by the needs and their evolutions, the increase of performance of the hardware and the media (addition and evolution of the storage elements). This classification will evolve based on the characteristics of he system made by the contractor. This evolution will only improve the quality of service.

The performances of the storage hardware has defined elsewhere in the CCTP should allow to guarantee the levels of service that will be defined for the whole storage system (hardware and software).

5.2.3.4 MONITORING AND MODIFICATION OF THE LEVELS OF SERVICE

This section describes the principles of operation for the execution and the modification of the levels of services. The inclusion of any change of any level of service is done asynchronously to the request.

The operational process of requalification is started periodically at most 48 hours after the request. When a requalification implies the transfer for one storage unit to another the level of service can not be guaranteed anymore. This fact must be logged.

5.3 INGEST

This module receives the SIP, verifies them, makes the transformation to AIP, extracts the metadata from the AIP for the « Data management » module and sends the created AIP to «the Storage module.

The functions of this module are clearly distinct from the pre-ingest functions, which are used to elaborate a conforming and complete SIP based on the elements given by a producer.

The possible scenarios are described here from an computational point of view.

Scenarios

Here is a description of the operations made where a new or an update SIP is ingested.

As a preamble, a producer establishes a connection to the ingest module through an interface. The transaction been either synchronous (directly) or asynchronous (through a token), it receives the signal of the end of the treatment or can consult the state of advance step by step by means of a transaction token.

Submit new SIP (ING_1.1)

1. A producer uses the interface of the Ingest module to connect. This connection needs a authentication.
2. Once connected, the producer is able to import one or more SIP. The period of time, the quantity, the volume, etc.. MUST respect the I-QSA.
3. The format and the content of the imported SIP MUST match the description made in the I-QSA.
4. Generation of the identifier
5. Conforming to the I-QSA, the SIP are transformed in one or more AIP. This operation can imply operations of transformation of formats of the data-objects.
6. The necessary metadata are extracted from the created AIP and send to the Data Management module.
7. The AIP is sent to the Storage module. The AIP contains its access and preservation policy.
8. When the preservation policy is fulfill (record of the data, number of copies, …), SPAR notifies the producer.

Submit modified SIP (ING_1.2)

1. A producer uses the interface of the Ingest module to connect. This connection needs a authentication.
2. Once connected, the producer is able to import one or more SIP. The period of time, the quantity, the volume, etc.. MUST respect the I-QSA.
3. The format and the content of the imported SIP MUST match the description made in the I-QSA.
4. Retrieval of the previous edition/version of the AIP updated through the Access module.
5. Conforming to the I-QSA, the SIP are transformed in one or more AIP. This operation can imply operations of transformation of formats of the data-objects.
6. The necessary metadata are extracted from the created AIP and send to the Data Management module.
7. The AIP (new edition/version) is sent to the Storage module. The AIP contains its access and preservation policy.
8. When the preservation policy is fulfill (record of the data, number of copies, …), SPAR notifies the producer.

5.3.1 Submit SIP (ING_1)

This function (cf. ING_1) allow the import of SIP in the system. Various kinds of SIP exist:

- SIP for creation
- SIP for update

The ingests apply on the data-objects as well as the metadata. This function should:

- Verify that the ingest conforms to the I-QSA (Quality of service Agreement for the Ingest)
- Guarantee the integrity of the transaction from the beginning to the end. The import transaction is finished when all the digital objects and the metadata are stored in the modules, i.e. the level of service required is reached. This transaction is ACID. A transaction may be synchronous or asynchronous.

5.3.2 Quality assurance of the SIP (QA)

This function is responsible for checking the structure of the submitted SIP. It uses:

- A « METS profile » (cf. informational viewpoint) for a narrative description of the structure of a SIP,
- A mechanism proposed by the contractor to validate the structure of the manifest and the metadata as described in the "Detailed Technical Book". The proposed solution will preferably be based on a XML formalism, as Schematron for example.
- Information representation metadata (cf. informational viewpoint) for the validation of the data-objects.

If the SIP is not conforming, the transaction is aborted. No data is created or updated.

The errors are logged and notified to the Administration module as well as the producer.

The monitoring process is logged and notified to the Administration module as well as the producer.

5.3.3 AIP generation

This function transforms a SIP into an AIP. The process of creation of the AIP may generate one or more AIP from a single SIP. The actions contributing to this result are as follows:

1. Transformation of the SIP manifest conforming to the "SIP METS profile" to the AIP manifest conforming to the "AIP METS profile", as defined in the "Detailed Technical Book".
2. Transformation of the formats of the data objects into the AIP formats in accordance with the I-QSA. This operation can last for a significant period of time up to several hours. Depending on the case, the operation will be synchronous or asynchronous. An asynchronous operation imply the use of a token allowing the monitoring of the operation.
3. Supply of the AIP to the "storage" module. Depending on the level of service, the AIP are stored in the appropriate storage unit.

The generated AIP contain their ingest and preservation policies.

The ability to resist the load is a decisive criterion. The contractor will supply any evidence to evaluate this criterion.

The operations of generation of the AIP are logged.

The performance of this function can be prioritized and configurable.

5.3.4 Extract metadata

This function extracts the metadata of the created AIP and sends them to the "Data management" module.

5.3.5 Submit an AIP to the storage

This function gives to the "Storage" module an AIP completed with its ingest and preservation policy.

5.4 STORAGE

5.4.1 STORE AN AIP (STO_1)

This function stores the elements forming an AIP in the appropriate storage unit.

The operations to be made are:

- From the list of the available storage units, search the storage unit containing the level of service according to the policies of the AIP.
- Use the recording services of the unit to store the elements of the AIP in a single transaction.

5.4.2 RETRIEVE AN AIP (STO_2)

This function finds and retrieves an AIP from the appropriate storage unit.

An AIP is retrieve as a whole, even if, in the storage infrastructure, it's the combination of various elements.

This function should allow for the definition of priority levels. It's then dynamically configurable.

5.4.3 AUDIT AN AIP (STO_3)

This function asks for the retrieval of an AIP and uses the control keys to verify the integrity of the AIP.

This function should allow for the definition of priority levels. It's then dynamically configurable.

5.5 DATA MANAGEMENT

5.5.1 STORE THE METADATA (DM_2)

This function records, in the Database, the metadata given by the "Ingest" module.

This record can result in an insertion or an update in the Database. This function is ACID. It can imply an update of the indexes, if the new metadata are used for simple search.

5.5.2 SEARCH THE METADATA (DM_1)

Two kinds of searches can be performed:

1. the simple searches (DM_1.1): these searches use predefined indexed metadata. The contractor will supply evidence to evaluate this criterion.
2. the deep searches (DM_1.2): these searches can be applied to all the stored metadata. The search times are not predictable and can take a large amount of time which implies the use of a token to monitor the operation and allow the retrieval of the final result. The token has an expiration date. The contractor will propose solutions to implement such kind of search. The proposed solution will allow:
    a. comparison operators: equal, inferior, superior, etc.
    b. Boolean operators : EXIST, AND, OR
    c. Tree navigation functions
    d. Restriction functions
    e. Sort functions

The ability to support complex queries (various levels of imbrications) on several millions of objects is a decisive criterion. The contractor will provide evidences to evaluate this criterion.

The searches are logged.

This function should allow for the definition of priority levels. It's then dynamically configurable.

5.5.3 RETRIEVE THE METADATA (DM_3)

This function allows the retrieval, from the Database, of part of the metadata, as provided during the storing.

This retrieval is ACID.

5.6 RIGHT MANAGEMENT

This module is detailed in the annex 4.

This module uses the following information on rights:

- The rights metadata associated with a digital object;
- The use condition associated with a particular access context;
- The decision tree defined by category of digital object;
- The agreements that apply to a set of digital objects defined by a query or as a list of access identifiers.

This information is collected from an external system, the SOLON system in charge of its management and its administration.

This module is in charge of generating the licenses of the digital objects, according to the context of access, to provide them to the "Access" module in a synchronous way at will. As way of consequence, this module is subject to the same performance requirements . Thus a pre-generation mechanism must be envisioned. This mechanism MUST take into account the evolving nature of the information on rights.

5.6.1 GENERATE THE LICENSE (RM_1)

This function provides the license associated to a digital object in a particular context of access. Depending on the case, an agreement or the result of evaluation of a decision tree will be use.

5.6.2 HARVEST RIGHTS INFORMATION (RM_2)

This function harvest periodically – according to a configurable frequency – the rights information in behalf of the SOLON system.

This information is a mirror of the rights information hold by SOLON and are stored locally for a latter use.

5.7 ACCESS

The "access" module exposes the following functions:

1. the search for digital objects from the metadata
2. the exposition of the metadata
3. the supply of DIP

It must be noticed that all these functions are always filtered through the "Rights management" module, except for an internal usage (Administration, Preservation).

5.7.1 SEARCH DIGITAL OBJECTS (ACC_2)

This function can be viewed as an interface to the "Search the metadata" function in the "Data management" module. The search is initiated by a query which result is either a list of identifiers or a mean (like a cursor) to browse into such list. The choice between the methods depends on an technical parameter (number of results) as well as on the query itself.

The queries are logged.

This function should allow for the definition of priority levels. It's then dynamically configurable.

5.7.2 EXPOSE THE METADATA (ACC_3)

This function exposes the metadata through an OAI-PMH interface.

The harvesting operations are logged.

This function should allow for the definition of priority levels. It's then dynamically configurable.

5.7.3 EXPORT DIP (ACC_1)

This function exports the DIP. This function uses the access policy. To guarantee the performances, it may use a cache for the DIP.

5.7.3.1 MANAGE THE PROCESS OF CREATION OF THE DIP

The creation of the DIP is the process of generating at will one or several DIP from one or several AIP. The operations needed are the following:

1. Ask the "Storage" module the appropriate AIP. If the AIP is AIC, the result can be a collection of AIP, if the query asks for it or a single AIP
2. Transformation of the AIP manifest conforming to the "AIP METS profile" into the DIP manifest conforming to the "DIP METS profile" according to the "Detailed Technical Book".
3. Transformation of the data-objects of the AIP to the data-objects of the DIP restricted to the information representation asked for. This action can take a long time. If necessary, the operation will be synchronous or asynchronous. An asynchronous operation implies the use of a token allowing the monitoring of the operation.
4. Depending on the access policy, the DIP is stored in a cache.
5. Filter the result with the license provided by the "Rights management" module according to the context of access. In case of failure of the "Rights management" module, a technical parameter gives the default license.
6. Supply of both (DIP, license) to the requestor.

The contractor must suggest a solution to structure and organize the DIP in order to optimize the performances, in particular in terms of execution time, bandwidth and storage space. For example, it may generate as many pre-DIP as possible uses, i.e. for example a pre-DIP for the A format, one for the C format, etc. A DIP is then generate by assembling the pre-DIP that are or not in this cache.

The ability of resistance to the load is a decisive criterion. The contractor will provide evidences to evaluate this criterion.

The supplies of DIP are logged.

This function should allow for the definition of priority levels. It's then dynamically configurable.

5.7.3.2 MANAGE THE CACHE OF DIP

In order to guarantee the performance of the level of service of a DIP, this module must manage a cache of DIP.

The contractor will describe its solution for managing the cache of DIP. The suggested mechanism should:

- Lean on the frequency and the freshness of the generated DIP
- Allow the definition of the retention time in order to impose the caching of selected DIP
- Make a regular verification of the freshness of the DIP

The management of the cache of DIP must include interfaces (GUI and API) for consulting and managing the cache in order to:

- List the state of the DIP by access frequency, freshness, retention time, etc.
- Order the rebuilt of all or part (depending on a level of service) of the cache

This list is not complete. The contractor can suggest other functionalities needed to manage the cache.

5.8 ADMINISTRATION

This module is central. It manages the monitoring of the system by the administrators as well as offers coordinating tools.

The functions of external administration will be specified latter.

5.8.1 COORDINATE THE ALARMS

This function receives the alarms from all the other modules – including the administration module – and from the SAS. Depending on parameters defined by the administrators, it can pass these alarms to the supervising system.

The alarms have at least three (3) different levels : critical, serious, warning.

5.8.2 IDENTITY FEDERATION (ADM_2)

This function distributes to all the modules the identity and authentication information to ensure a overall coherence in terms of security of access.

5.8.3 MONITORING (ADM_8)

This function allows the retrieval of the current state of a particular instance of :

- A module
- A process
- A storage unit
- A record

5.8.4 PLANNING (ADM_6)

This function instantiates a plan. The plan applies on:

- An access policy
- A storage audit
- The rebuild of the metadata stored in "data management" module
- A migration program

5.8.5 INTERFACE FOR THE ADMINISTRATORS

5.8.5.1 MANAGING THE STORAGE UNITS

This function allows the management of the storage units. It gives access to the following operations:

- Configure the characteristics of a storage unit
- Configure the interfaces with the drivers of the storage elements
- Monitoring
- States of the storage units, the records and the locks.

5.8.5.2 MANAGING OF THE POLICIES (ADM_3)

This function allows :

- The definition of a policy

- The editing of a policy
- The import of the description of a policy
- The monitoring of a policy (ensure the right use of it)

5.8.5.3 MANAGING THE PROCESSES (ADM_4)

This function allows :

- The definition of a process
- The editing of a process
- The import of the description of a process
- The monitoring of a process

5.8.5.4 MANAGING THE IDENTITIES (ADM_1)

SPAR provisions the identities.

The latter are provisioned in a regular manner, i.e. they are updated by a external identity manager.

The transaction is initiated by the external system. SPAR then has to supply a LDAP server interface to allow the accomplishment of this operation.

5.8.5.5 MONITORING OF THE PLANS (ADM_7)

This function allows the monitoring of the execution of a plan. It provides in particular:

- Its status
- The number of processed AIP
- The number of AIP to be processed
- The time of execution
- An estimation of the remaining time
- The mean time of processing for an AIP

5.9 PRESERVATION

5.9.1 MANAGING THE REPRESENTATION INFORMATION (PRES_1)

This function supplies representation information (cf. informational viewpoint) to allow the validations and the necessary migrations on the formats of the data. The validations are made by the "Ingest" module during the QA of the SIP. The transformations are made by the "Ingest" module for the generation of the AIP and by the "Access" module for the DIP generation.

5.9.1.1 VALIDATION (PRES_2)

A format profile for the validation presupposes the existence of processes allowing the comparison of data against those profiles. The implementation needs the existence of a couple: format profile, process. This couple is called "validation package".

Several solutions for the implementation can be used: supply of the validation package with the process (in the form of a plug-in) and the format profile, supply of the profile alone with check that a process exists, etc.

5.9.1.2 TRANSFORMATION (PRES_3)

The information representation for a transformation presupposes also the existence of processes allowing the transformation from one format to another. The implementation needs the existence of a three elements: input information representation, output information representation, process. This set is called "transformation package".

5.9.2 MANAGING THE TRANSFORMATIONS (PRES_5)

This function covers the planning, the monitoring and the execution of transformations. These operations lean on the migration plan as defined by the OAIS standard. In our case, this implies the execution of the following operations[15]:

1. Supply of the transformation packages needed by the "Ingest" module to generate the new AIP. For example, transformation package from the "TIFF uncompress color image" format profile to the "JPEG2000 lossless compressed color image" format profile.
2. Query of the SIP according to the plan (prioritized) of migration for supply of these DIP/SIP to the "Ingest" module. This operation can take a very long time even months….
3. Supply of the transformation packages needed by the "Access" module to continue to provide the DIP generation. For example, transformation package from the "JPEG2000 lossless compressed color image" format profile to the "JPEG lossy compression in A size" format profile, to the "JPEG lossy compression in C size" format profile, etc.

The operation number 2 needs a monitoring of the progress of the operations : AIP transformed, AIP to be transformed, priority and sending of progress report to the administrators and the preservation experts.

The migration processes have in general less priority than the processes of DIP generation. This should be defined by the priority management.

All the operations of transformation should be logged.

5.9.3 HARVEST OF INFORMATION REPRESENTATION (PRES_4)

This function allows the harvesting of information representation given by the format registry. It's a planned operation. It's parametered by the administrators.

The operations of harvesting are logged.

[15] They presuppose the previous elaboration of a migration plan : study, design, test, validation, etc (out of scope).

5.10 COMMON SERVICES

5.10.1 ADMINISTRATION

This function allow the administration of a module. The main operations are:

- Configuration of the module (authentication, parameters, etc.)
- Consultation of the state of the module; the current operations, the events, the logs, etc.

5.10.2 COORDINATION

This function allows a module to supply its logs to the "Administration" module. It allows also the immediate propagation of alarms to the "Administration" module.

5.10.3 STORAGE

Each module has its own mean of storing data, supplied by a management or treatment unit.

The information packages SIP, AIP and DIP are stored by the SAS.

5.10.4 SECURITY

Each module has its own means for security. In addition, the identities they managed are provisioned to them y the "Administration" module.

5.10.5 LOGGING

This function allows a module to log all the events and actions occurring.

5.11 INTERFACES

The interface MUST be described with the WSDL (Web Service Definition Language) formalism, except when they use a existing recommendation or standard, as for example OAI-PMH.

These interfaces MUST be secured: the contractor should explain in its response all the security measures he will implement, especially concerning the Web Services.

5.11.1 EXTERNAL INTERFACES OF SPAR

Here is the description of the external interfaces of SPAR. The contractor is free to add other interfaces or protocols, in addition to those asked for in this section.

5.11.1.1 API INTERFACES

Actor	***Module***	***Exchange***	***Type***	***Mandatory protocol***	***Optional protocol***
Producer Pre-ingest	Ingest	Control messages	Web Services	http (rest)	http (soap)
Producer Pre-ingest	Ingest	Import SIP (data)	Web Services	http (rest)	http (soap), ftp, sftp, rcp
Disseminator	Access	Harvest metadata	OAI-PMH	http	
Disseminator	Access	Search metadata	Web Services	http (rest)	http (soap), open-URL
Disseminator	Access	Control messages	Web Services	http (rest)	http (soap)
Disseminator	Access	Export DIP (data)	Web Services	http (rest)	http (soap), ftp, sftp, rcp
SOLON	Rights management	Harvest metadata	OAI-PMH	http	
Format registry	Preservation	Harvest metadata	OAI-PMH	http	
Administrator	Administration	MIB	Web, command line	snmp	
Business intelligence	Administration	Export administrative data	To be defined		
Accounting system	Administration	Export accounting data	To be defined		
Identity manager	Administration	Manage identity		LDAP	

5.11.1.2 GUI INTERFACES

The graphical user interfaces (GUI) are Web interfaces. The contractor is free to add other interfaces, in addition to those asked for in this section.

Actor	***Module***	***Function***
Administrator	All	Administration
Producer Pre-ingest	Ingest	Import SIP (ING_1)
Disseminator	Access	Export DIP (ACC_1)
Disseminator	Access	Search in metadata (ACC_2)
Administrator	Access	Manage DIP cache
Administrator	Administration	Manage storage units
Administrator	Administration	Monitoring (ADM_8)
Administrator	Administration	Planning (ADM_6)
Administrator	Administration	Interface for the administrators
Administrator	Administration	Manage policy (ADM_3)
IT staff Administrator	Administration	Manage process (ADM_4)
Administrator Preservation expert	Administration	Monitor plan (ADM_7)
Preservation expert	Preservation	Manage information representation (PRES_1)

5.11.2 INTERNAL INTERFACES OF SPAR

The internal interfaces MUST be described with the WSDL (Web Service Definition Language) formalism. The SAS MUST also offer a command line interface. Moreover, to ensure the performance requirement, the contractor CAN propose additional interfaces.