Deliverable D1.2

Architectural design - first iteration

Project Acronym	OASIS
Grant Agreement number	297210
Project Title	Towards a cloud of public services

Project co-funded by the European Commission within the ICT Policy Support Programme

Deliverable reference number and title	OASIS_D1.2
Status	Final

Dissemination level[1]	PU
Nature[2]	R

Due delivery date (project month)	M7
Actual delivery date	16/02/2014

Lead beneficiary	ATOL CONSEILS ET DEVELOPPEMENTS SAS
Contributing beneficiaries	Atol, All pilot sites, Open Wide, AtReal, Polito.
Author(s)	Christophe Blanchot, Yannick Louvet, Bruno Thuillier, Sébastien Guardiola, T. Benita, Jérôme Poittevin, Andrea Sanna, Marc Dutoo

Revision History

Revision	Date	Author and Organisation	Description[3]
0.1		Andrea Sanna (Polito)	Creation
vf		Bruno Thuillier (Pôle Numérique)	Final corrections and Approval

[1] Dissemination level: **PU** = Public, **CO** = Confidential, only for members of the consortium and Commission services

[2] Nature of the deliverable: **R** = Report, **P** = Prototype, **D** = Demonstrator, **O** = Other

[3] Creation, modification, revision, final version for evaluation, revised version following evaluation

Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Deliverable abstract

The main objectives of OASIS are the development of a platform able to facilitate the access to information and services by federating services and sharing data, promoting data as commons, and building a social network with a great respect of private life.

The initial step prior to the analysis and design of the architecture is to define and quantify the OASIS system requirements. System requirements are based on the consultation of Service Providers as well as users and stakeholders; moreover, an analysis of privacy requirements will be also used to drive the design step. These inputs are formalized in D1.1 and D1.31 and they have helped to design the platform presented in this document. Details about authentication and privacy will be described in D2.3.

This report presents the deep analysis of the requirements, the apparent contradictions between some constraints, taking into account the needs of security, scalability, performance.

OASIS is at the same time a federation of services and data, an interoperable database, an advanced social network, a user portal, and relies on cloud computing technologies.

So we designed features required to federate applications and data providers (a broker to find resources, centralized authentication to ensure security, notification flow). Federated services are OASIS clients, they can use “à la carte” specific features provided by OASIS kernel.

We designed a “web” architecture (a “REST” architecture) to federate all OASIS components and all services provided during the Pilot, and to include new services.

OASIS must be an “elastic” infrastructure, to support a large volume of data and many competing processings: we use “cloud” architecture and technologies to allow a large scale use.

We worked on the modelling of data and meta-data inspired from the semantic models of RDF linked data. We built a simple modelling for the providers (in compliance with the uses in the REST architectures), guaranteeing satisfactory performance on large volumes and compatible with the superimposing of RDF vocabularies and semantic requests (SPARQL).

Privacy, access rights management, rights delegation, has been investigated and we designed a social graph to describe the relations between people and entities, and to ensure privacy. Detailed functional implementation and technical solutions will be described in deliverable D2.3.

The integration of the portal is defined in this document, but its specifications will be made in the D2.4 document.

We explain why we need to operate OASIS on a cloud platform (OASIS uses a PAAS to provide SAAS), and how we will select cloud architecture and technologies.

OASIS is a complex system: we identified risks requiring our vigilance during the pilot phase.

Project Management Review

	Reviewer 1: WP leader			Reviewer 2: B. Thuillier		
	Answer	**Comments**	**Type***	**Answer**	**Comments**	**Type***
1. Is the deliverable in accordance with						
(i) the Description of Work and the objectives of the project?	☒ Yes ☐ No		☐ M ☐ m ☐ a	☒ Yes ☐ No		☐ M ☐ m ☐ a
(ii) the international State of the Art?	☒ Yes ☐ No		☐ M ☐ m ☐ a	☒ Yes ☐ No		☐ M ☐ m ☐ a
2. Is the quality of the deliverable in a status						
(i) that allows to send it to European Commission?	☒ Yes ☐ No		☐ M ☐ m ☐ a	☒ Yes ☐ No		☐ M ☐ m ☐ a
(ii) that needs improvement of the writing by the originator of the deliverable?	☐ Yes ☒ No		☐ M ☐ m ☐ a	☐ Yes ☒ No		☐ M ☐ m ☐ a
(iii) that needs further work by the partners responsible for the deliverable?	☐ Yes ☒ No		☐ M ☐ m ☐ a	☐ Yes ☒ No		☐ M ☐ m ☐ a

* Type of comments: M = Major comment; m = minor comment; a = advice

Table of Contents

1 Introduction

1.1 Scope of OASIS

OASIS is a federation of services and data

OASIS allows federating applications, provided as services by external suppliers, accessible to the users through a portal and interconnected via shared data.

OASIS is the center of an ecosystem offering a wide and open range of services, giving the users the possibility of objectively choosing the services which suit them best, without being prisoners of format owners.

To reach this goal, OASIS offers advanced features:

- A complete catalogue of the available services and data, described according to common criteria
- An on-line subscription to the services with management of the invoicing modes adapted to every service, and the intermediation between the price paid by the users and the remuneration of the suppliers
- The ability for users to provide review content concerning services, and mediation between the users and the providers
- A complete security management (authentication, rights management, privacy, availability, integrity)
- A workflow management system allowing to dematerialize complete e-government processes using several federated services

OASIS is an interoperable data base

The data management and the data repositories are critical issues in OASIS, and must allow a full understanding of these data by every application.

Indeed, one of the objectives of OASIS is to constitute a public heritage with as open as possible data between all the stakeholders having legally the right or the need to access to these data. It consists in the collaborative construction of a **common good**.

OASIS allows to break the silos of data, and thus to federate data usually segmented by business, or by organization (even by application).

To reach this goal, OASIS offers advanced features:

- The data storage

- A query language on stored data
- A data description model usable by every business and sector of activity
- A data rights management
- A data qualification mechanism
- Mechanisms of collaboration and mediation on shared data

OASIS is an advanced social network

OASIS offers a unique authentication, an advanced management of one's personal data to be used by the applications and services (with a sharp control of access rights), and the ability to link all the users according to their relationships: professional and personal relationships, affiliation to organizations and companies.

This social network thus depicts all types of real life relations between persons, including moral persons (communities, companies, associations, public bodies), and allows to manage the contractual or legal relations, as well as the related rights and delegations.

Accordingly, OASIS offers a base for the description of natural and moral persons (a social graph), their identification, and all the links existing between these entities, with a precise qualification of the links.

OASIS protects private life and confidentiality of everyone, through several specific features:

- A sharp control of data access by the applications
- An interlinking mechanism guaranteeing the privacy of everyone as well as the real identity of the natural person and his relations.
- Mono-directional relations (i.e. relation of an entity A towards an entity B can be defined without imposing a relation of B towards A)
- The management of several possible profiles with a single account allowing people to easily manage their multiple identities and lives (civil, professional, associative...).

These features and mechanisms are established in strong relation with the works on the management of personal data within the framework of the OASIS project, as well as on the data and platform governance.

OASIS is compliant with the laws of the European countries for the protection of privacy.

The authentication, the management of access rights in OASIS and the federation of services totally refer to the social and professional interactions.

Specific technical mechanisms guarantee a total confidentiality, even towards the platform administrators.

OASIS is a smart and user friendly Graphical User Interface

The OASIS portal is an application itself providing a large set of features displayed in a user-friendly manner:

- creation of a user account,
- creation of several profiles,
- an application store,

- a personal environment,
- a data store,
- a dashboard with status of user activities and personal notifications

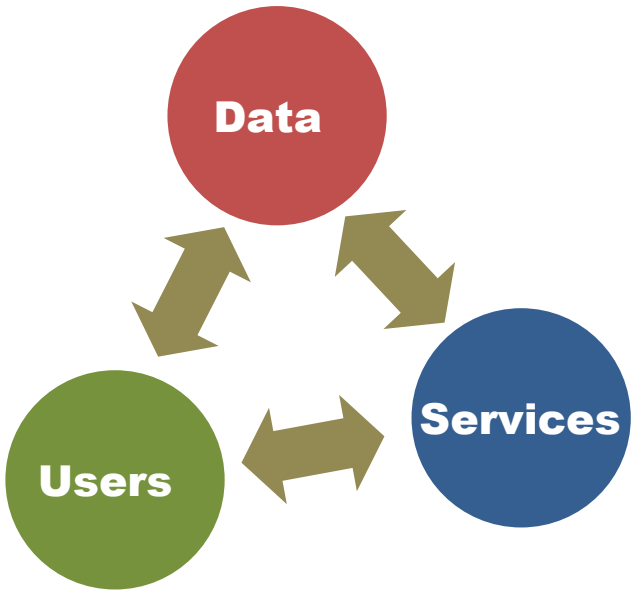
OASIS takes advantage of « cloud computing »

OASIS relies on Cloud Computing technologies which have been tested to allow an efficient access from any Internet access, to monitor and to manage the required resources.

OASIS therefore promotes migration utilities and local government business software in the cloud, facilitating the migration and providing additional benefits that fully justify the migration.

OASIS is built around three pillars of equal importance constituting three centers of gravity according to a perspective :

- **centered on data**: data is at the core of OASIS whose one objective is to make a data patrimony as a common good. This data is available for users and service providers who use it and enrich it. This is the **technical** point of view : data is the backbone of the OASIS architecture.
- **centered on services** : the services being the means to create, update and use data, OASIS is mainly accessible via the services and as such is seen as a system centered on services. This is the **economic** point of view : the OASIS valuation is made via the services.
- **Centered on the users** : The new uses of the IT systems are oriented towards a more social use organized around the users. The user disposes of the maximum possible amount of information (vision centered on data), tools for exploiting them (services) and can organize his work thanks to a portal giving him access to the resources, and thanks to the social graph allowing him to describe his personal information, his interactions and his relations towards the rest of the world and organize the management of information around « his » universe. This is the **functional** point of view : OASIS gives the control to the end user.



1.2 Document objective and content

As specified in the DOW, this D1.2 document is 1.5 task (architectural design) result.

Reviewers asked us to improve the document which was submitted during the suspension period. As planned in the DOW, the provided version was a first iteration.

Given the work done since the resumption of the project, we are capable of drawing up a second iteration of the architecture design.

This document then includes a report of the additional work done during the second half of 2013 to refine the architecture choices and takes into account the implementation progress.

As explained in this document, OASIS is a service federation and a datacore.

As a consequence, the main issue is to provide functionalities and resources to federated services.

These federated services are OASIS clients, and we don't have to design them.

And as the OASIS portal will be developed in a specific task of WP2, this document doesn't give details about the graphical user interface.

The architecture design is based on the work described in other deliverables :

- The analysis of the platform requirements, for the federation services and the end users (deliverable D1.1)
- The analysis of the security constraints, and particularly of the protection of private life (deliverables D1.31 and 1.32)
- The first iteration of the architecture design (previous version of this document)
- The study document of the KPI (D5.1)

This document is then the second iteration of the OASIS architecture design, taking into account the additional studies which have been made, the feedback from the modules implementation and the services adaptation, and the comments and advice from the reviewers at the end of the review of the documents delivered during the suspension period.

This document is based on the new D1.1 version (platform requirements).

It begins with an analysis of these requirements so as to propose ways to address them. A functional architecture is then proposed: OASIS is divided into sub-systems and in functional modules.

From this functional architecture, we define a technical architecture of all the OASIS ecosystem, allowing a large use of the « cloud computing » various technologies.

We then analyse the main risks of the project, and identify areas of vigilance.

A table in chapter 8 shows how the proposed architecture can meet the requirements which are described in the D1.1 document.

Some topics are addressed in a dedicated document, and are then not detailed in this deliverable:

- The management of the authentication, personal data and privacy of data (and then rights management) are detailed in the D2.3 deliverable.
- The portal functionalities are detailed in the D2.4 document: this architecture document however describes the division of roles between the portal and the other modules.

1.3 Methodology

Functional architecture and technical requirements for WP2 have been built according to the following information and studies:

- Analysis of users and providers requirements (D1.1)
- Analysis of OASIS management and administration needs (KPI, governance, business model)
- Prospective study of future needs, evolutions, scalability, big data analysis needs, etc…
- Study of security and performance constraints
- Study of constraints to adapt a service

During the suspension period, we defined a functional architecture which includes the visions of all partners making the OASIS project more concrete, the relationship between the various systems within OASIS and the way it is operated « in the cloud ».
From this first iteration of the architecture and the functionalities of the interfaces between the components, everyone could project himself in the implementation, which allowed to adjust the functionalities which have been retained (in particular data modelling, an important point in OASIS).
The providers 'needs have been refined giving them the possibility of projecting themselves into the OASIS functioning and integrating the use of the services supplied by the datacore and the kernel. A new version of the D1.1 document helps to more precisely target the needs of data sharing, the possibilities of contribution of every federated application and the use of the various functionalities defined in the architecture, so as to refine the precise description.

We are thus designing an adapted architecture for OASIS, a functional architecture and a technical architecture.
We chose a full web architecture, a REST architecture (REST = Representational State Transfer): see chapter 5.
This architecture allows a wide independence between the various modules (loose coupling between modules), and so a wide availability for functional evolutions.
This point is very important for a federation of services: each sub-system has its own responsibility in the complete system, and its own development and management process.

We designed an architecture independent from technologies, each module can be implemented with its own selection of technical solutions.

The « cloud » layer on which OASIS is based also remains very independent from the retained architecture, which gives us a wide choice and evolution for the hosting of the various modules (including the federated services).

Architectural design and cloud layer insure elasticity of each sub system of OASIS.

2 Requirements

2.1 Functional requirements for the users

The deliverable D1.1 analyzes all the functionalities OASIS has to implement.
These functionalities can be summarized as follows according to the stakeholders and issues.

The citizens:

- An access to public services (e-administration), and services proposed by companies (order tracking, banking services, etc.)
- An access from any device provided with a browser and with an Internet access
- Following the progress of one's files and demands
- Subscribe to on-line applications (SAAS mode)
- An access to public data (open data)
- The management of one's personal data, and the control of their use by the various thirds and the applications they use
- Delegate formalities or, on the contrary, to handle the formalities of persons they're responsible for
- Review and rate the services
- Find and get in touch with providers

The users in professional surroundings:

- Benefit from a single account, manage the division or the sharing of data from their various roles (private individual, professional, and possibly multi-professional)
- Access to the services proposed by partners (companies, administrations, etc.) they are in relation with
- Use the applications provided by their employer
- Follow the progress of one's files and demands
- Access to public data (open-data)
- Manage one's personal data and controlling their use by the various thirds and the applications they use

The IT-managers:

- Buy on-line applications on behalf of their organization
- Manage settings and the specific data of their organization
- Manage access rights to data and applications in a unique and centralized way
- Access to public data

- Access to activity statistics
- Review and rate the services
- Find and get in touch with providers

2.2 Functional requirements for the federation

Applications (and web services) and data source providers:

- Registering their services on OASIS
- Finding resources (data, web service) and guidelines
- Getting services purchases

Applications and web-services:

- Having a user authentication
- Using data sources and sharing data to be interoperable

Data sources:

- Having a user authentication

Data governance:

- Managing repositories and ontologies
- Managing correspondence rules between repositories
- Managing access rights to data stored in OASIS
- Allowing a collaborative construction of data
- Qualifying the quality of data

2.3 Technical constraints

Important technical constraints are added to these functional needs (technical requirement expressed in D1.1)**:**

- Security (confidentiality of data and transactions, integrity of data and transactions, availability of data and services)
- Performances and scalability
- Ability of the architecture to manage huge volumes of data (« big data ») and to support an important number of simultaneous users (flexibility of the architecture)
- Measurement of the pilot's technical indicators
- Management, measurement and diagnostic tools

2.4 Federation of data and services

To offer the maximum of flexibility in the integration of services and the sharing of data, and to allow a progressive integration of the services (which can require an important evolution of the latter), the service providers must be able to use the functions they wish:

- Centralized authentication of the users in OASIS: this function must necessarily be integrated by the federated services; the minimum level of integration is thus the unique authentication mechanism of OASIS, and the access to the service from the user portal
- Storage and partial or total mutualization of the data in OASIS
- Use of qualification mechanisms of the OASIS data
- Use of notification and workflow management system in OASIS

To allow an efficient service integration, OASIS will be able to manage default behaviors associated to every service to ensure their functioning and avoid side effects due to the fact that after integration, applications will have to retrieve data from a large set of data instead of dedicated databases.

For example:

- Automatic addition of a filter on research requests
- Setting default access rights
- Default data qualification level

Note:

Additional services can be useful for service providers:

- A forge
- Bugs tracking
- A centralized help desk
- Libraries for the development of services
- Hosting platform
- Etc…

We do not integrate these tools into the architecture. The pilot will allow defining the exact needs, and the work on the governance will allow refining the scope between OASIS and the applications, and the services to be supplied to providers.

It does not exclude that some additional services can be implemented during the experimental phase.

3 Analysis

OASIS is a federation of services and data : OASIS then offers the user a complete ecosystem including specific business services (then the services planned in the pilot phase).

The functionalities seen (and required) by the end users (citizens, employees, IT managers) then include job functions managed by federated services which are integrated as such into the OASIS project and are out of the scope of the architecture design.

However the architecture study includes the specific features of OASIS :

The services which must be supplied to federated applications to improve cooperation and global user experience (cooperation on data, exchange with the portal, single authentication, use of the social graph, notifications between applications, etc…)

- The management functionalities of the ecosystem (management of personal data, subscription to services, setting and personalisation), accessible via the portal, for the end users and the application and data suppliers
- Functionalities which can fulfil safety requirements
- Functionalities which can manage the system, the business model and the governance model (particularly the governance of data and the application qualification)

The design of the technical architecture includes the communication issues between all the sub-systems (then including the federated services), as well as the elasticity (scaling-up), then functionalities studied in the functional architecture, but also solutions allowing the scaling-up of applications and federated data sources.

3.1 Architecture resulting from the first iteration

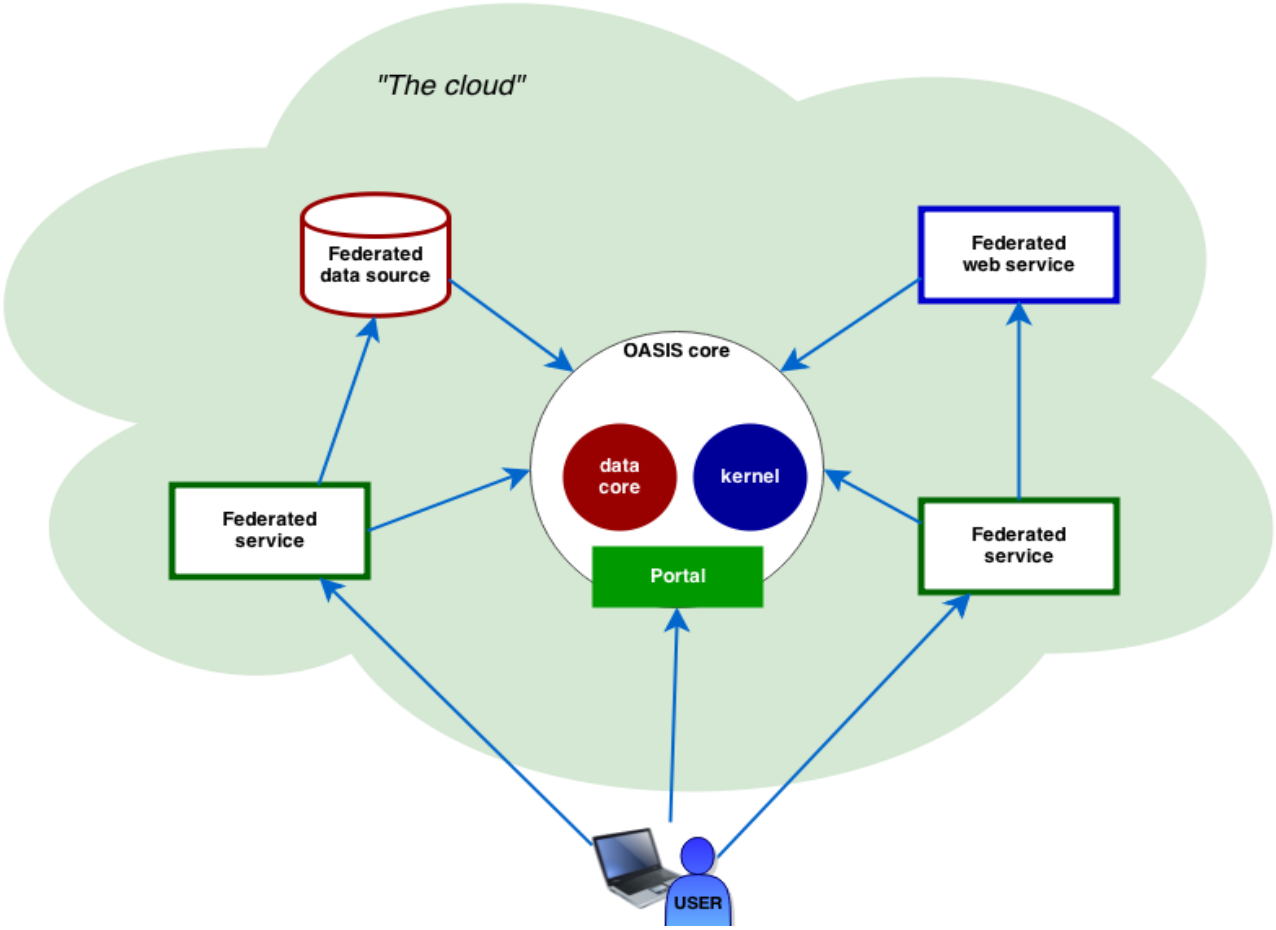
3.1.1 Functional architecture

During the first iteration of the architecture, we studied the functional division of OASIS into sub-systems and nodules.

The OASIS complete ecosystem includes:

- Federated services
- Federated data sources

- Data managed by OASIS (under the responsibility and the governance of OASIS) : the datacore
- Functionalities of service federation (authentication, notifications, catalogues, etc …) : the kernel
- The user portal



We then create several independent sub-systems (but communicating between them according to explicit interfaces).

The functional architecture retained allows to build a modular technicval architecture, guaranteeing scopes of responsibility and different service levels for each sub-system.

OASIS is not an integrated central system but a federation of services and data.

We thus refered to this division into sub-systems for the complementary analysis leading to the second iteration of the functional architecture.
Chapter 4 provides a detailed description of this functional architecture.

3.1.2 Technical architecture

The OASIS architecture is a Web architecture : the various sub-systems from the functional architecture are independently managed and communicate via the Internet network according to the Web protocols.

We then have a REST type architecture (REpresentational State Transfer).

One of the advantages of the REST architectures is to maintain the lowest coupling as possible between the different modules and sub-systems.

These can evolve independently from each other.

The lack of the notion of session makes the services independent and allows a different service level (for instance an availability) according to the uses : for instance, the module in charge of logging the events can à priori be out of order without preventing the system from working.

We further describe (chapter 5) the detailed technical architecture and give more information on the pros and cons of REST architectures.

3.2 Social Graph

3.2.1 Issues

The identity and the private data are more complex than a simple user account: indeed, it is necessary to manage the users, the administrations, the companies, the households, etc…

These data are a directory allowing to define access rights (a company is for example a group of users), but also data themselves which can be used by applications (the link between a user and his/her company indicates his/her employer).

Besides, we wish that the use of OASIS can be centered on the user : the latter must be able to organise the use of OASIS and the access rights to data, around his relations and interactions with the rest of the ecosystem, and in particular around his social relations (including family, private and professional relationships) with other persons and organisations.

Furthermore, delegations of rights are necessary to perform some actions, such as:

- A company or an administration is a person who buys services in OASIS, makes formalities, but these actions are necessarily delegated to a physical person
- Guardians have to make actions for minors or persons under guardianship
- The actions made for a family (a household) can be made by the various adults of the household.
- The minors can be led to use OASIS with their own account (access to child welfare services, events announcement, etc…)
- Some operations can be delegated to friends or relatives.

So, OASIS must then allow the description of the various entities (persons, companies, associations, administrations, working groups, services, families, etc…), and relations between these entities.

That's the reason why we architecture OASIS around a social graph:

The social graph has several roles in OASIS :

- This is the users'directory (in the traditional meaning of the IT directories, and then including the notion of organisation and groups)
- It gives the possibility to each user to describe his social relations and is then the basis of the implementation of functions of « social network » type in federated applications (guaranteeing the user with an absolute control of confidentiality)
- It can store personal data (the relations are personal data and some personal data, such as the parents or the employer, are in fact relations) : we will store all personal data in the social graph to benefit from advanced management mechanisms of confidentiality implemented in the graph
- It is the basis of the rights management on the other data (the rights can be given according to the relations between entities, for instance « all the persons employed by one administration »)
-

This graph contains very confidential private data: indeed, the links between the persons are confidential (for example, a person under guardianship does not necessarily want that it comes out).

Furthermore, these links can be confidential towards one of the extremities: for example, a father who left his family does not necessarily want that his child can find him.

3.2.2 Functionalities

The social graph allows the management of entities:

- Persons
- Organisations (moral persons)
- Various groups

And the relations between these entities:

- Membership to a group
- Employee/employer relationships between one individual and one organization
- Relationships between organizations
- Family ties
- Bonds of friendship
- Delegation (an individual can act for one organization or for another individual)

With the social graph we can store personal data and finely manage the rights for each data. The access to personal data and links is explicitely validated by the user (via secure authorization mechanisms), for one application and a given context of action.

Of course, unlike classic social networks, it must not be possible to make a wide search on the OASIS social graph.

To put in relation persons or entities, we shall then use the explicit procedure of agreement:

- One of the two asks to create a link with an entity or a person he gives the characteristics to.
- A unique code is created that he has to pass on to the other party either by sending an e-mail (if he is sure of knowing the e-mail address), or by any other safe means
- This unique code allows identifying the second part to validate the link, after the verification by OASIS of the indicated characteristics. Of course, the other party has to create an account to validate the link if he/she has not any.

Regarding the critical link between the social graph and the protection of private data (the social graph links all the data of a person, the protection is thus very important there), specific mechanisms of protection of these data are tackled in this document, and the work will be strongly improved in the 2.2 task (WP2), and the deliverable D2.3.

3.3 Data modelling

3.3.1 Background

Data are the key components of OASIS

Indeed, two major objectives of OASIS are:

- Interoperability of applications thanks to the use of the same data. OASIS must allow the removal of business silos: each business manages different databases and have different descriptions model for the same data.
- The creation of a heritage of data constituting a "common good", under open governance.

OASIS must meet the following challenges:

- Providing a data description model serviceable to all business, avoiding data duplication: data defining one thing in the real world must be present only once in OASIS.
- Providing a data description model usable by new applications and allowing them to interact with data, while ensuring the right level of confidentiality and integrity.
- Encouraging users and service providers to share data that are not personal or confidential.
- Providing a strong data securing, and in particular ensuring the confidentiality of private data with a sharp control of shared private data.

3.3.2 RDF

During the first iteration of the architecture study, we identified RDF (Resource Description Framework) as being the solution to model data.

RDF gives the possibility to represent all knowledge in the form of a triplet « subject predicate object ».

Subjects, predicates and objects are represented by a single identifier (the object can also be a value), and can then be used in numerous triplets, as subject, as predicate or as object (see in annex for more details and examples of data representation according to the RDF logic).

Identifier are most of the time URL (uniform resource locator): thus RDF allows to link data between the different sources over the Internet. We are dealing with Linked Data.

RDF is completed by the description of taxonomies (RDFS), ontologies (OWL), rules (RIF and SWRL), and by a query language (SPARQL).

All these languages, as the basis of the semantic and linked data web, are referred to under the generic term « RDF » in the next section of the document.

Indeed, RDF, as we are going to see below, offers a unified model to represent:

- Repositories (called vocabularies or ontologies) by business
- The description of the links between all the elements, all the data, to be represented
- Mechanisms of metadata (description of vocabularies), being "piled" without limit
- The possibility of describing the same data according to several vocabularies
- Inference rules to create knowledge

During the first iteration and after further thinking, we also identified some difficulties linked to RDF:

- potential access performance problems to data through the RDF model
- difficulties to guarantee the integrity of data, RDF being unable to guarantee that data is complete and allowing the description of conflicting information
- the choice of the level of granularity of a « document » : how can we group triplets (thus basic information) to create objects (documents) useable by applications
- the complexity of using the RDF representation by the providers of the services federated by OASIS
- the choice of the ontologies to describe the data shared in OASIS

3.3.3 Complementary studies

Further studies were realized, investagating other groups and other projects, and in particular the W3C« Linked Data Platform » working group.

We met specialists who had worked on the RDF implementation to represent and analyse data, and particularly:

- Professor Djamel Abdelkader Zighed, from the Institute of Human Sciences (CNRS)
- Professor Jérome Darmont, from ERIC lab (Warehouses, Representation and Knowledge ingeneering, Lyon 2 University)

- Professor Dimitris Kotzinos, from the Cergy Pontoise University, who is involved in the European InGeoCloudS CIP project.

We worked with Mr Bernard Vatant, from Mondeca, and with Mr Nicolas Chauvat, from Logilab, on the representation of data in pilot services according to RDF ontologies, so as to concretely assess the difficulties.

We also worked with many books and studies available online: see the « references » chapter.

Concrete RDF implementation cases that we studied did not allow us to find usecases for « management » applications, with real-time updates, and data completeness and integrity.

The various projects we could study, in meeting persons who worked on them, present at least one of the following differences with OASIS:

- Either data is much more targeted
- Either updates are not frequent
- Either the RDF datacore is made of business components, algorithms adapted for every type of data.

The W3C « LDP » project is particularly interesting for the issues it tries to solve are mainly the same than those in OASIS.

But the work of this group is still under progress and thus not so far advanced to use it for the OASIS development.

Moreover, the proposed solutions are based on the extension of existing protocols, not yet fully validated by IETF.

The work done on data modelling from concrete pilot cases showed us that the existing ontologies are today inadequate. One of the RDF interests, that is the use of « normalised » ontologies, is then not applicable.

The problem of the granularity level of the documents corresponding to a URL (basis of the principle of data linked in RDF) is also complex: the choice of the triplets mentioned in a document becomes a business choice (we choose what describes an « object » according to a business view

of that object). And the choice to remain at the triplets level (and thus to build the documents with SPARQL or similar queries) is very complex for the applications.

The various projects based on RDF to represent the applications business data (then data needing integrity, completeness, frequently updated and requiring fast response times in reading and writing) made one of the two following choices :

- either the implementation of specific business mechanisms using data (what we want to avoid for the OASIS datacore)
- either a less universal representation than RDF, with solutions to create RDF views on data (RDF is then used only for data reading and for the creation of meta-data).

3.3.4 Final choice of data representation

These various studies led us to the following conclusion: the use of a pure RDF representation as the OASIS datacore architecture is too complex and is beyond the scope of a CIP type project.

It would be necessary to carry out consistent research and development activities based on the ongoing work done by the WC3 LDP group.

We then took the decision to represent data according to models which are commonly used in REST type architectures, by describing the various « objects » used and shared by the federated applications in the form of a tree diagram of attributes and properties, in the JSON format.

For instance, this is the data model for a city:

```
"sample.city.city" : {
    "name" : "sample.city.city",
    "fieldMap" : {
        "founded" : {
            "name" : "founded",
            "type" : "date",
            "required" : false,
        },
        "pointsOfInterest" : {
            "name" : "pointsOfInterest",
            "type" : "list",
            "required" : false,
            "listElementField" : {
                "name" : "zzz",
                "type" : "resource",
                "required" : false,
            }
        },
        "inCountry" : {
            "name" : "inCountry",
            "type" : "resource",
            "required" : true,
```

```
            "resourceType" : "sample.city.country"
        },
        "name" : {
            "name" : "name",
            "type" : "string",
            "required" : true,
        },
        "populationCount" : {
            "name" : "populationCount",
            "type" : "int",
            "required" : false,
        },
    },
}
```

A description according to RDFS taxonomies will be modelled on these native descriptions, to allow third applications to discover and exploit these data. A SPARQL engine will also be implemented for the query of data.

But the updates will be made within the context of the native representation of data (the RDF views will be in reading only).

3.3.5 Data containers

Several constraints on data coexist:

- The analysis of privacy requirements (D1.31) shows that some experimental sites must store data of public bodies in their country.
- We have to validate during the pilot the performances of the retained solutions, and the growth of the volumetry of a database has a direct impact on its performances
- Some data will remain linked to a particular entity (there will be an interoperability of the applications for the various businesses, but no direct sharing with the outside)
- Some data are widely open and collectively built (and do not thus need fine management of access rights), while others are restricted on the contrary to some bodies
- There are data which will remain stored in service providers (for example because they want to keep a of specific access mechanism, or because they are very frequently updated and because it generates strong constraints of performances)

For these different reasons, we add two notions :

- The introduction of several datacore to use, if necessary, different storage technologies and to manage in which datacenters the data is stored (this replaces the notion of container defined in the first iteration : it appears easier and more efficient to manage distinct datacores rather than managing severla storage types for one same datacore)
- The notion of « scope » can include a set of data under same governance rules

We however integrate in OASIS a mechanism which can punctually manage inter-datacore requests: the data homogenizer (see below).

3.3.6 Governance scope

We must define scopes which correspond to a data governance scope. The scope is then defined by types of data and by a scope which can be geographical or of another type.

In order not to include any complex business rule in the datacore, the data scope will be indicated by the application which creates data (or modifies it).

However, in reading, it must be possible to read all data of one type without knowing all the scopes concerned. The scope is a searchable attribute (the use of one or several scopes in reading request limits the research to this scope).

The scope defines governance rules and includes the following information :

- Default owner and authorized owners
- Mediation rules : types of mediation authorized, default mediator, authorized mediators (see below the paragraph on mediation)
- Histories conservation rules
- For each access type : default access rights, maximal access rights (that is limitation of the access), minimal access rights (to prevent an application from limiting access beyond a certain limit)
- Data licence (this is an accessible information which does not activate an automatic treatment)

The access authorisation management at the level of the applications themselves will allow to define the rights of each application for each scope: the person in charge of a scope explicitely gives the use rights of the scope to the applications he accepts to be consulted or modifies the scope data.

In order to better understand this notion, here's an example :

Imagine OASIS federates applications which can manage the street furniture of public bodies.

We can define 3 types of data:

- The reporting of their state (damage, etc…) : « reporting » (this being also used for other things than street furniture)

- Maintenance interventions : « maintenance » (this being also used for other things than street furniture)

A citizen appplication was created by one third to consult the available equipments (on a map) and report damage.

Public authorities in Drôme agreed with accepting a collaborative management of furniture (under the responsibility of each community)

But Ardeche does not accept this collaborative management (public authorities do not want to open their information to the public).

Maintenance companies of furniture have an application on which they indicate the interventions made and on which public authorities ask for interventions and follow them.

The scope : « management of the street furniture in Drôme ». The scope includes the 3 types of data defined above and is applied (by the applications during the creation of data) to data related to street furniture in one community in Drôme.

The owner :

- For the « street furniture » type: the community (thus the node of the social graph representing the community) owner of the equipment.
- For the « reporting » type: the community owning of the equipment, even if the reporting is made by a citizen from another community
- For the « maintenance » type: the company in charge of the maintenance (even if the request is made by the community)

The mediator:

- For the « street furniture » type : the job of the person in charge of « street furniture » job, moderation a priori
- For the « reporting » type : the job of the person in charge of maintenance, moderation afterwards
- For the « maintenance » type : no moderation.

3.3.7 Rights management

The rights management mechanism of OASIS is a compromise between a universal management at the level of OASIS itself, and the performances.

We thus define 3 types of data:

- The **public data**: these data are unreservedly accessible in reading to everybody, and can be restricted in writing.
- The **private data**: these data are modifiable only by their owner or the users he delegated rights to, and are by default not visible except for explicit mention of the owner

- The **data with managed rights**: these data of a definition of access rights in reading and in writing

The management of sharp rights on the data is largely delegated to the applications (because these rights are generally defined by business processes).

For every container, a type of default data is defined, with possible exceptions by service (a service being authenticated by an access code).

During the access to a container by a service, the OASIS datacore thus knows immediately which of the 3 methods it must use to verify the rights, which allows to restrict the cases of advanced management (which will necessarily lead to less good response time for the users).

For private data, the rights to other users than owner are explicitly indicated on the data.

For reading and writing rights on managed data, or writing on public data, it is possible to define them both ways:

- A clear description of the rights at the level of the data itself, when created or modified by a user and an application later authorized.
- A default definition of the rights at the level of:
  - A type of data
  - A scope of data: the data depending on an entity (via the social graph)

The rights are then defined for:

- Non authenticated users (some data being accessible without any authentication)
- A user or a users group (users belonging to a group or an entity)
- The data owner (if an owner is associated to the data)
- A group defined in the social graph
- Users linked by a specific relation with a give node in the social graph

The rights which are defined for each kind of access (reading, addition, modification, etc…).

Default values for owner and rights management are defined at the scope level.

Rights management is further investigate in the deliverable D2.3

3.3.8 Data qualification

The data managed by OASIS must be qualified, that is their quality (their probability of being accurate, the possible validation by trustworthy third parties, etc.) must be indicated and improvement mechanisms of data must be implemented.

This functionality allows applying « fuzzy logic » rules to data for the applications which request it.

This qualification will be used by the applications either by including qualification criteria in data requests or with default criteria (associated with the application/entity couple which uses it as for

the rights and the default proprietary) automatically added to any request from this application in (reading and writing).

This mechanism is necessary so that the applications requests are not polluted by data from other applications meeting the criteria but which would be less qualified.

We propose the following qualification criteria:

- The data quality is summarized by a coefficient of 0 to 9, 9 indicating a validated and reliable data, and 0 an information entered by a not authenticated Internet user.
- Application having generated (or updated) the data
- Last update

Precise rules should be defined within the framework of data governance.

To each application, a maximum coefficient is applied: this means the application will not give the data a coefficient higher than this maximum.

3.3.9 History management

OASIS allows the management of the update history of update (who updated, when, which was the value before update)

History management rules (number of entries stored, maximal ancientness) are managed by scope and by type of data.

3.3.10 Data collaborative management

OASIS offers data collaborative management mechanisms.

History management and qualification mechanisms are part of them.

A mechanism allows managing a collaboration around open data update (and only on the open data: private data are managed by their proprietary and the data with advanced rights are managed by the applications).

This mechanism allows introducing the moderator notion on a dataset.

A dataset is defined by:

- A data type
- A scope of data: the data depending on an entity (via the social graph)

This moderator is an appointed user, an entity or the data owner.

This principle of moderation can start (according to what is defined for the dataset) in the following cases:

- Attempt of modification (either deletion or creation) of data for which the user has no rights.
- Modification (or deletion or creation) of data for which the user has the rights.

In the first case data is not modified (or created or deleted), but a modification proposal is submitted to the moderator: he then has tools to validate or not validate the suggestion (these tools will be defined in the specifications of the portal, D2.4 document).

The requester is then informed of the decision which has been taken. Every application can define whether it wants start or not this mechanism during updates (with a default behavior and a behavior by application or type of process).

For the second case, the update is made and the moderator is informed of the update (former and new value, who made the update and which application was used). The moderator can cancel the update if he has correct access rights (the requester is informed in this case).

3.3.11 Data homogenizer

This OASIS module is integrated in the architecture to allow:

- The conversion of data from one format to another
- Access to external data via OASIS
- Multi-datacore requests

This module is planned to allow a wider and easier integration of external data (data provider) and services (service provider).

Its use has to remain marginal.

It won't be used in the Pilot. So it won't be implemented in the Pilot.

It refers to rules of data translation.

It is seen by the applications as virtual containers (and as such described in the catalog of containers), and implements only some requests, developed in coordination with the services publishers (requests described in catalogs).

3.4 Federtaion of the services

OASIS is a federation of services.

For that purpose, some functionalities are required to improve the user experience, to allow services to communicate between them and with users.

3.4.1 The authentication

OASIS supplies a centralized authentication mechanism, allowing the users to authenticate only once.

This feature is widely requested in D1.1, by pilot sites and services providers.

This feature and the technical architecture related to are studied with more details in 2.2 task (D2.3 document).

But we have already worked on this subject, and the DOW requests a first iteration of this subject in D1.2.

Various mechanisms exist and have been tested, there's no need defining a specific solution: the implementation study (WP2) will allow to choose a method and the associated protocols, to ensure the robustness of the authentication, and the availability of numerous libraries to implement it by the federated services.

Thus, we don't have to define a specific solution with its specific APIs

It should be noticed that the European Community develops a project of authentication and management of identities: the **Stork project** (see appendices and references).

This project is not yet sufficiently developed and used to be a basis for OASIS.

At this experimental stage, we prefer to implement only one method of authentication to simplify the developments and maintenance.

It is likely that Stork must be secondly implemented and live with the solution first retained.

For the same reasons and to control the level of strength of passwords, we also decide not to implement the possibility to authenticate with a third account (Google, Facebook, etc). It's also better to insure the privacy settings of personal data.

Authentication must be done on a web page provided directly by OASIS, because the user must be able to verify all the certificates of the web page.

Passwords must be stored via a hash system, that is we don't store the passwords themselves, but the result of a cryptographic algorithm, which can be reversed (we can't retrieve the passwords with the hash codes).

Identities management

The authentication allows with a login code and a password to check that the person who well connects corresponds to the account registered in OASIS.

OASIS is not capable of validating the effective identity of a person. Eventually, Stork will allow verifying the identity.

In the first version of OASIS, we do not implement a specific mechanism for the identity verification (which would imply that main private data are not modifiable by the user).

The scenarios of use of public services described in D1.1 document show that the verification is made for each procedure with specific documents joined to the request.

User management is made in the social graph (see chapter 3.3).

3.4.2 Context notion and role

For each user, it's necessary to distinguish several contexts: "my personal context", "my civil servant context", "my employee context".

The context also corresponds to a role (the function of the user within the context).

In case of break of the link (transfer of an agent for example), the context is kept (after having been purged of the private information): it allows to keep the links which were established between some data and a user (for example, who made a defined action).

When a user connects to OASIS, he selects the context he connects to. For security reasons, the change of context requires to enter the password (It avoids that a colleague can get back a personal context on an unlocked browser).

The relation links with other entities are visible in the concerned context.

Each user can select the private data he makes visible in each context.

A specific password can be associated with a context (entering this password automatically when connecting under this context).

This functionality is particularly interesting to create contexts dedicated to access from mobile applications, to reduce the risks of intrusion in case of theft or loss of the device.

All these subjects are detailed in D2.3 deliverable.

3.4.3 Catalogs

The OASIS kernel must manage the resources catalogues available in OASIS, which can be used by the various actors:

- Federated services (see next paragraph)
- Service pricing
- Available data (data sources, scope of data)
- Available web services

These catalogues are public data (except for a few elements such as the exact information of the contacts: a « restricted distribution » indicator is thus planned on the catalogues data), essentially

used by the user portal for the search of resources and for the collaboration between the various OASIS suppliers.

The catalogue is also a complete documentation to look for resources, make services communicate, build new services on the basis of what already exists.

A mechanism of community "rating" (mark and comment) of the catalogue resources is included (and implemented in the portal).

Catalogs describe the information in several languages: for every element, at least in English, default value, and languages in which the services are valid and where the providers operate.

The portal can possibly implement mechanisms of automatic translation for the information not existing in the user's language, or show by default the text in English and propose the other available languages.

Federated services

The catalogue of federated services is exploited by the portal (also by other mechanisms of the OASIS kernel, notification mechanism included).

It contains the following information:

- Name of the service
- Description
- Coordinates of the company proposing it (or link to a provider)
- Registration date on OASIS
- Categories (for search and display on the store)
- Targets (type of entities to which it is addressed)
- Type of license
- Versions follow-up (list of the modifications)
- Link to access to the service
- Managed functionalities/processes (according to a common OASIS repository which will be defined in catalogs itself), and for each:
    - The general business functional chain
    - The step in the functional chain
    - Direct link if available (this link can contain parameters, like the current user or the identification of data to be treated)
    - Types of required upstream data (in connection with data repositories)
    - Types of produced data (in connection with repositories)
    - Type of upstream processing (functional chain and previous stage in the business process)
    - Type of downstream processing (functional chain and next stage in the business process)
- Languages managed by this service
- Countries or regions it is compatible with (regarding regulations)
- Data sources used
- Web services used

We must note that the entry of information in this catalogue during the integration of a service can be boring, but this stage seems necessary to ensure an efficient interoperability between services.

Service pricing

The definition of the suggested pricing options, of the price distribution between the various actors (services, web services, data, OASIS infrastructures), of invoicing modes, services subscription modes, will be made at the portal level.

Data sources

A data source can be external (federated data provider) or internal, directly stored in the datacore.

The data source catalogue contains the following information:

- Source name
- Description (repositories whom data, scope, etc. are associated)
- Coordinates of the structure which manages it
- Openness (open data or private data)
- Access authorizations: which services registered on OASIS have access to these data
- Access modalities (explanatory text)
- Access Protocols (explanatory text, in restricted distribution)
- Languages managed by these services
- Countries or regions it is compatible with (regarding regulations)
- Data sources used
- Web services used

Web services

OASIS allows cataloguing web services. It contains the following information:

- Name of the web service
- Description
- Coordinates of the company which proposes it (or link to a provider)
- Registration date under OASIS
- Categories (classification for the data presentation tools)
- Type of license
- Versions follow-up (list of modifications)
- Detailed specifications of the service and access syntax
- Subscription modalities (how to register, how much it costs)
- Languages managed by this web service
- Countries or regions it is compatible with (regarding regulations)
- Data sources used
- Web services used

In this first version, service providers who want to refer to a web service must directly negotiate with the corresponding supplier.

OASIS only manages one catalogue and does not at present manage the service subscription. On the other hand, the remuneration of the web service operator is possible directly through OASIS, if this cost is directly connected to the global cost of the service which uses it (the management system of the OASIS suppliers' remunerations allows distributing the price paid by the user between several contributors).

Rating and comments

The service community rating is implemented in OASIS with the usual mechanisms:

- A rating of satisfaction (0 to 5 stars)
- The possibility to make comments

Comments are kept with the catalogues.

A moderation mechanism which can be exploited by the portal is implemented (comments and notes are then in « check » mode during an adaptable deadline, and are not visible during this deadline).

This feature is consistent with open and participative governance.

Other catalogs

The lists of the values used for the classifications in the catalogues are catalogues themselves, for example:

- Business processes
- Languages
- All messages and texts for the different languages (can be used by all applications)
- Target types (types of companies, communities, citizens, associations, etc.)
- License types

These catalogs will be defined while implementation.

3.4.4 Status and notifications

According to requirements, a mechanism of notification and management of status must be implemented.

This mechanism allows passing on information between the services or between a service and the portal.

It offers advanced integration services to the users and to the applications:

- Display on the portal of the status of the actions or the current formalities (for example, status of a formality with an administration)
- Notification of events to the users (among which changes of status) on the portal or by a means of communication (SMS, e-mail, etc.)

- Notification to the applications themselves of an operation concerning them.

The services do not know one another (the services are not interfaced but share data), it is OASIS which must determine the service which can receive a notification (or which user must be informed that an action must be done).

This is the reason why the catalogue of services explicitly describes the processing that each service carries out, the input data (that is the data which activate the need to make the processing) and the data downstream (that is the data which are produced and which will be transmitted via another service in the functional chain): this is from these data that OASIS can define which service it must inform (and the OASIS internal technical data enable you to know who uses the service, to inform the users).

Regarding the status, the services have the ability to manage status associated with objects (data).

When a service indicates to OASIS the status of an operation (follow-up of an order, of a formality, etc.), it provides the exact corresponding data.

This allows OASIS to follow the status modifications automatically even if some services treating the data have not implemented the functionality.

These mechanisms allow adapting to the integration level of each service, in offering extremely powerful functionalities to simplify the succession of operations within the various services.

3.4.5 Communication with users

The OASIS portal, security features and federated services may need to send information to users.

A communication module will be thus implemented. It provides the following advantages :

- Ease the communication to users thanks to different channels (e-mail, text message, OASIS portal, etc…)
- Centralized management of the means and the reception periodicity of the different type of information (OASIS is user-centric).
- Confidentiality of personal coordinates (e.g. e-mail address or text message): applications can communicate with users without having access to their personal data.

3.4.6 Service Identification

External applications (service providers, web services providers and data providers) using the features supplied by OASIS must be identified: all the requests (made via a secured protocol) must include an identifier and the signature of a third service (the signature management information are managed in the technical data in a strictly confidential manner).

It is not essential to use certificates delivered by an external authority, nor even to pass by a system of third-party certification with an authority of certificate integrated into OASIS: systems purely « peer to peer », as a pre-shared key, can be enough.

But this mechanism must be secured to guarantee that a service is not usurped. The implementation of limitations on the domain name and the address would be relevant.

It is also important that OASIS cannot be usurped towards third services: an OASIS signature will also have to be registered in the responses make by OASIS to services requests.

3.5 Data sources in OASIS

Various types of data sources are described in this analysis, this paragraph brings extra light on their roles and their differences.

Data in OASIS are the fundamental point, the objective of OASIS being to make a patrimony of data.

Several data sources coexist in OASIS :

- The social graph (paragraph 3.2)
- Service data stored in containers of collaborative data : datacore (paragraph 3.3)
- Data stored and managed by external data providers, federtaed to OASIS
- Catalogues

Thanks to OASIS we can achieve fundamental (and founding) objectives regarding data :

- Sharing of data controlled with the management of access rights, guaranteeing for the users that these rights are respected.
- Incentive to open data more widely for a large contribution on the enrichment and updating of these data
- Complete social network (with a fine management of the relationships), but with strong confidentiality guarantees total control for the user of what will be disseminated.
- Respect without limit of personal data.

The datacore

The datacore is a secured storage service of data, directly guaranteed by the OASIS governance and under the technical control of OASIS.

It allows the management of all types of data, with an access rights management. It guarantees the respect of implemented collaboration rules, availability of data (availability rate and restoring capacities), and a neutral governance of these data (governance rules being defined by scope).

But this is the applications themselves which manage the data access rights in the datacore.

Data providers

Data providers are external data sources, federated to OASIS.

You can find their description in the catalogue. They are in charge of the access protocols, models, rights management, etc…

They can rely on the services of the OASIS kernel (such as the authentication and the social graph).

The legal structure which controls OASIS cannot guarantee the safety of data from data providers :

The interest of the datacore is then to guarantee the respect of the access rights, the availability and a neutral governance of data.

The social graph

The social graph is particularly significant for OASIS.

Indeed, a social dimension is more and more integrated by many services. Besides, the companies'applications are based on a directory, describing the roles, relationships between entities and people, etc…

But these information are particularly confidential for they are personal data.

There can be information redundancies between the social graph and the datacore, and there will be some. For instance the mayors' list implicitly exists in the social graph but cannot be extracted in a list : services will then need to maintain this list in the datacore (or another data source).

This graph is social because it structures all the « social » functions of OASIS guaranteeing the respect of personal data.

Contrary to the datacore, the social graph data is completely managed by OASIS which manages the job meaning and defines the access rights (whereas the datacore data is governed by federated services).

The catalogs

The catalogues with data linked to the working of OASIS, in which the elements of the services and data federation are described.

3.6 Logging and supervision

OASIS must integrate a logging and audit module (activity, requests and errors).

These logs are used for the system administration, for the incidents resolution, and for the history of activities proposed to the users and suppliers via the portal.

The aim of this feature is to provide a unique API to log all kind of information, to allow their exploitation for many uses (search and error analysis, logging, audit, debugging, preventive analysis, statistics ...)

This module is designed for core modules (kernel, datacore, portal), but nothing prevents from extending it to federated services.

Advanced functionalities may be drafted: alerts, load forecasting, real-time security analysis, etc ...

This module will also allow to meet the supervision need and the KPI follow-up (see works in WP5).

It then integrates counters for various types of requests, bumped every time a request corresponding to the counter is logged.

The setting of these counters is made in the technical data. We will count in real time the access to datacore, authentications, incorrect requests, etc…

These counters will be questionable according to the SNMP protocol, which allows to manage them automatically on a supervision platform (which will manage alerts, historical curves, etc.)

3.7 Cloud computing

OASIS is “on the cloud”, because it provides Software as A Service, with an Internet browser and thru Internet to final users, and because it federates data and services thru Internet.

OASIS also uses specific “cloud computing” technologies.

OASIS must adapt itself to the variation of load, allow the increase of the volumes of data and network load, and allow a distribution of data, processings and redundancies, the system will then be implemented on a Cloud Computing solution.

It was decided to only set one OASIS system (the architecture is distributed at the cloud level but is seen as a single system).

However, the data management must impose the storage localization of some data (the next section describes the solutions implemented).

The technical choices for the cloud platform will be made within the WP2 (document D2.1). The choices turn towards an IAAS type solution (Infrastructure as a service, see terminology, chapter 1.3), provide by a cloud hoster.

In an IAAS cloud, the middleware contributing to optimize the elasticity (that is the capacity to take up in load and to absorb the load variations).

The arbitration between a "public" cloud, that is a wide sharing of the resources with other solutions, and a "private" cloud will be finalized within the framework of the governance: but for the pilot, we will choose a private cloud for the kernel, the datacore and the portal.

The security is assured in both cases with a similar level (even if a private cloud is retained, it must be opened on the outside since this is the vocation of OASIS).

The retained solution must be controlled by European companies (to avoid being dependent on laws as the American Patriot Act), and will have to guarantee the localization of data and processings (only in Europe, and possibility of restricting some data to one country).

4 Functional architecture

This section describes the functional architecture of OASIS: we work on requirements (implicit and explicit), to design functional sub-systems, divided in functional modules.

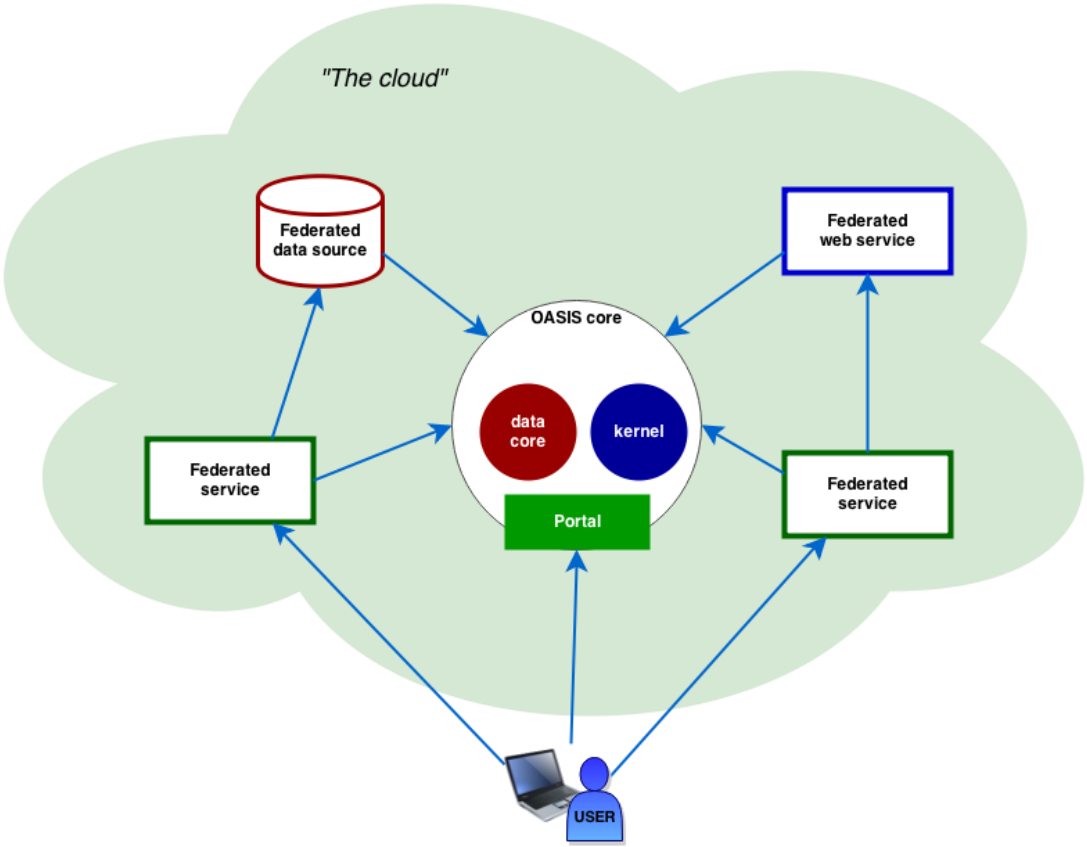
These functional modules are designed in order to be strongly consistent (functional proximity of data and functionalities) and weakly coupled (which allows them to evolve without any impact on each other).

This functional division is then the technical architecture support described in the following chapter.

As seen in chapter 3, the scope of this functional architecture is:

- The services which must be supplied to the federated applications to improve cooperation and global user experience (cooperation on data, exchanges with the portal, unique authentication, use of the social graph, notifications between applications, etc…)
- The functionalities of the ecosystem management (management of personal data, services subscription, setting and personalization), accessible via the portal, for the end users and for the applications and data providers
- Functionalities meeting the safety requirements

4.1 OASIS and its ecosystem



In the following chapter, the focus is on the functional architecture of the « OASIS core », the central circle of the above diagram (kernel, datacore and portal).

The OASIS core includes all the modules allowing the implementation of all the services supplied by OASIS to the users, federated services and federated data suppliers.

The user accesses OASIS via a portal and accesses directly or via the portal to the federated services.

The services refer to data stored directly in OASIS or by external suppliers.

The OASIS core gives services for authentication and management of authorizations, to federated applications and data suppliers

Please note that the applications federated by OASIS include services directly accessible to the user by a Web interface, as well as web services which can be used by a desktop application, an application for smartphone or tablet, or by another service (for example, services of route calculation, payment, etc.).

Remarks:

1) We only represented one portal on this diagram : the portal can in fact be regarded as a service and different portals will be implemented (by country, topics, etc…). It will be necessary to define the safety requirements and the validation process in compliance with the governance rules, the portal giving every one the possibility of managing one's personal data and authorizations (the portal can then intercept all personal data).
2) We also represented one single datacore: but several datacores can coexist. In the context of the pilot, we will implement two datacores (see details in document D2.1 which describes the implementation).

4.2 Portal

The portal is the entry point to OASIS for the user: the end users (citizens, agents and employees, IT managers) use the OASIS services via the portal and via the federated services. They do not see the datacore and kernel directly : these two sub-systems are used by the portal and by the federated services.

The portal is then the OASIS user interface and offers the following functionalities :

- Management of ones'account and personal data
- Management of the access to personal data
- Management of one's relations with the social graph's entities
- Management of various display preferences
- Access to services federated by OASIS

The portal functionalities are described in document D2.4 : we then do not describe them in this document.

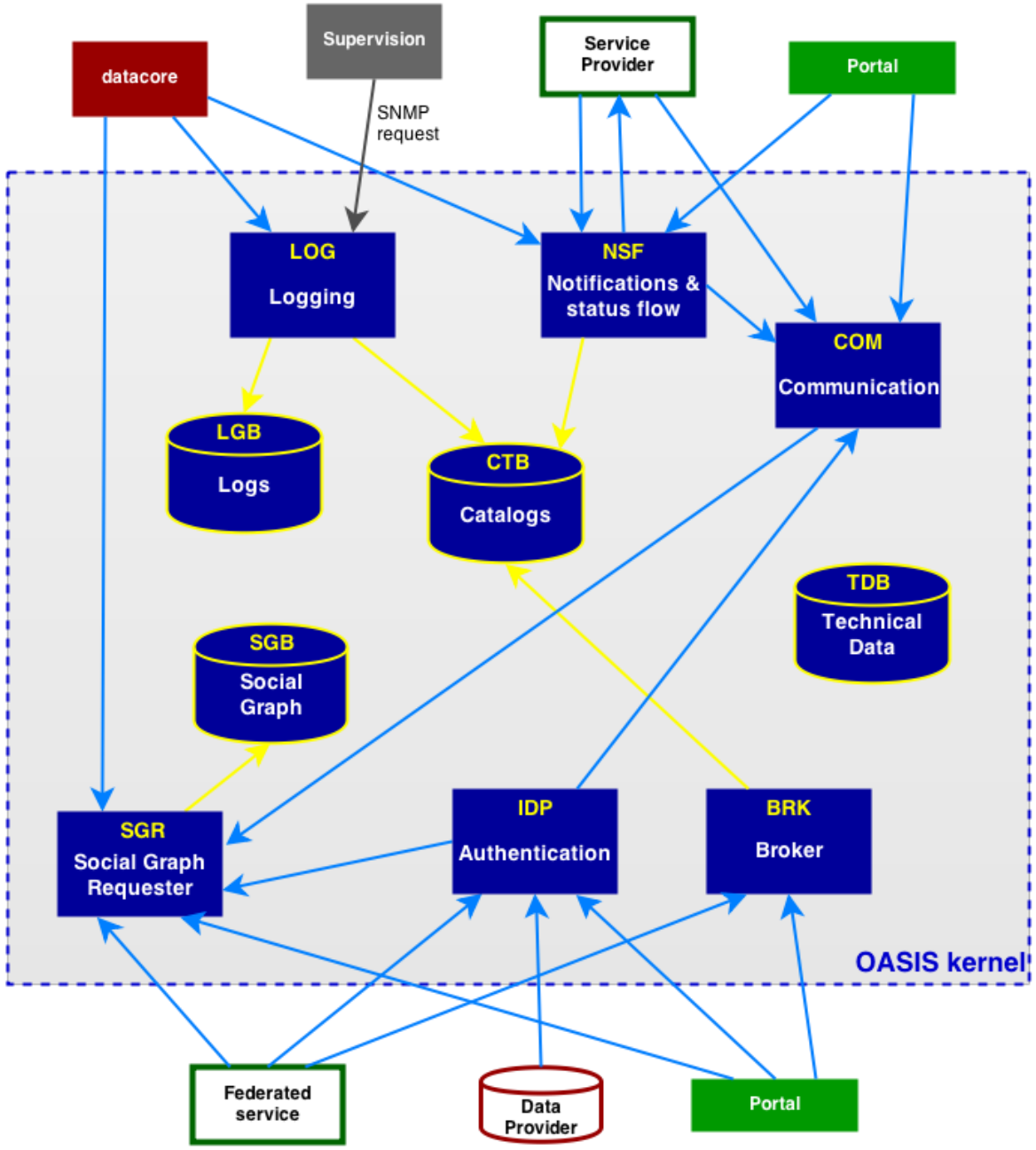
4.3 Kernel

Additional work made since the end of the suspension period led us to develop the functional division since the first architecture iteration.

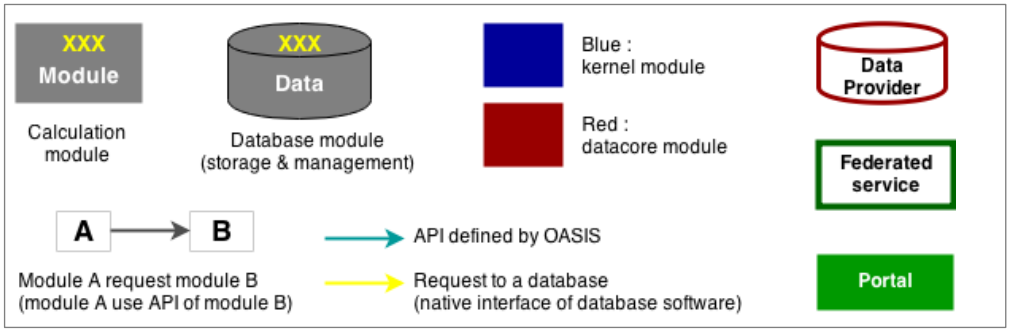
The main evolution concerns the transfer of the rights management module in the datacore (see the following chapter on the datacore functional architecture)

The social graph does not refer to this rights management module.

The division of the kernel into functional modules is then as follows :



Légende



Remarks :

- The « technical data » database is used by all the kernel modules : to simply the diagram, the links are not represented.
- The « log » module is also used by all the kernel and datacore modules : to simplify the diagram, the links are not represented.

OASIS kernel includes 4 databases :

- **Technical data** : all internal data of datacore modules including temporary data is stored in a dedicated base. This allows modules to simply benefit from the elastic property of the database solution retained, and thus to easily share data between various module types.

- **Catalogs** : all the OASIS catalogs (see paragraph 3.4.3) are stored in a dedicated database.
- **Logs :** The « LOG » module (see below) uses a dedicated database to store and centralize logging and audit info.
- **Social Graph :** the social graph is based on a specific database, adapted to graph representation and guaranteeing a very high security level of personal data (see deliverable D2.3)

This functional division into 4 databases does not require to use 4 different technologies.

OASIS kernel is made of several modules.

- **Social graph requester**: this module allows you to ask and update the social graph while respecting the access rights and confidentiality. It is used by the federated services, external providers who want to use it to manage the access rights to their data via the datacore (the access rights to data rely on the links and nodes of the social graph : see deliverable D2.3). The authentication module also relies on the social graph to identify the users.
- **Authentication**: this module allows to manage the mechanisms of authentication and access authorization to data. It manages a tickets mechanism. This module is studied in detail in the deliverable dedicated to this topic : D2.3.

- **Broker**: the broker is the access module to the catalogue data (see paragraph 3.4.3) including the questioning of the OASIS federated resources.
- **Logging**: a single module managing the events logging, the audit information and calculating the KPI. The LOG module is then made of API SNMP to allow control tools to retrieve the KPI (see paragraph 3.6)
- **Notification and status flow**: this module manages the notifications (particularly to determine which application must receive it), and the updating of the statutes. The evolution of the statutes and the notifications (see paragraph 3.4.4).
- **Communication**: this module allows the other kernel modules and federated services to send info to the users without knowing their details nor their preferences regarding the reception of notifications and various info (see paragraphe 3.4.5).

4.4 Datacore

Compared to the first iteration, we decided to allow the implementation of several datacores : one datacore is in fact a particular data provider.

As seen above in this document, the main differences between a datacore and a dataprovider are :

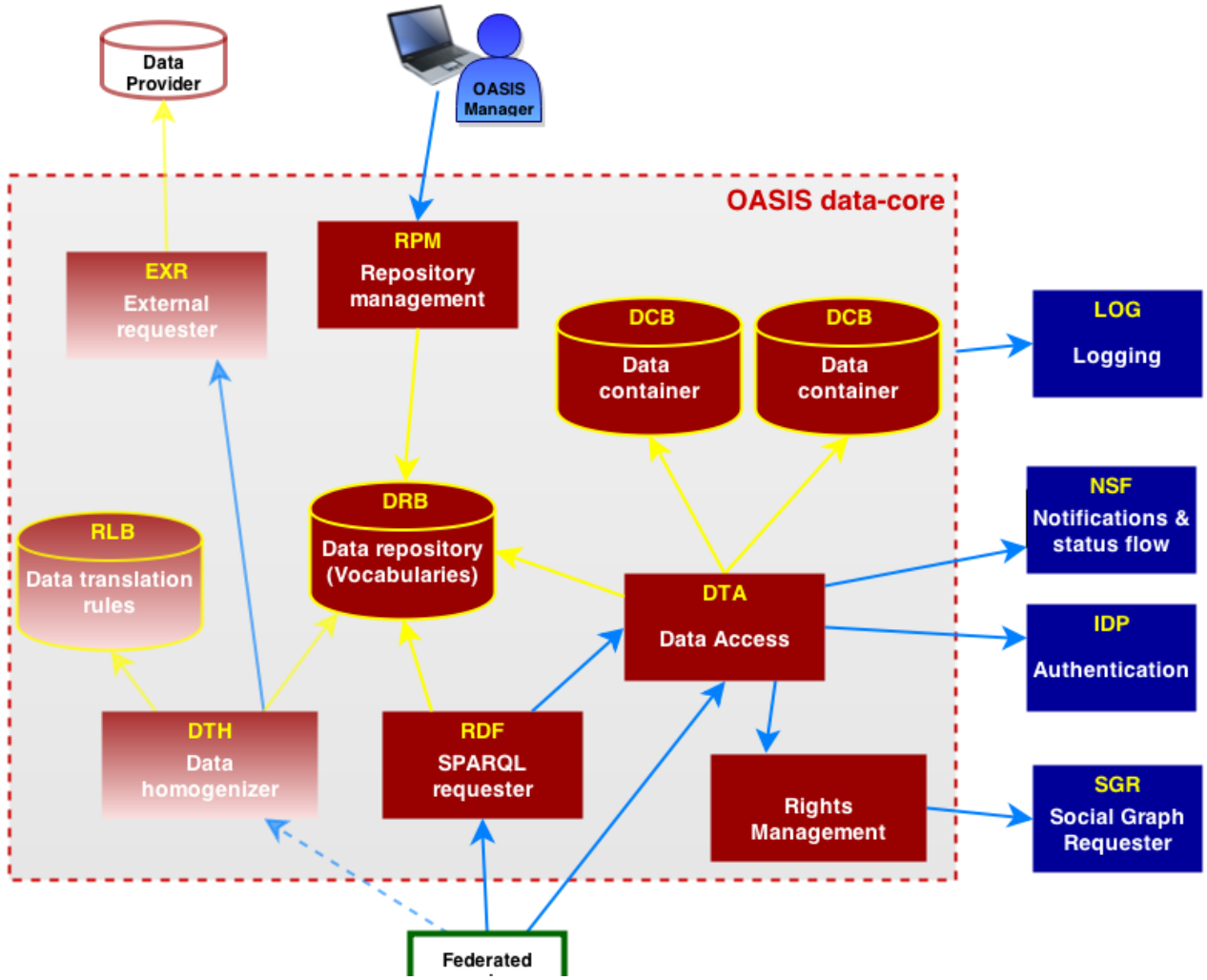
- Datacore is directly under the OASIS governance and allows OASIS to guarantee the security of data for the users, and the equality of treatment of service suppliers for the use of data.
- Datacore offers standardized access API for all data sets.
- Datacore offers a description of data useable by all applications
- Datacore offers advanced functionalities of rights management, data qualification, mediation.

The complementary work done since the suspension period also led us to significantly develop the functional division since the first architecture iteration.

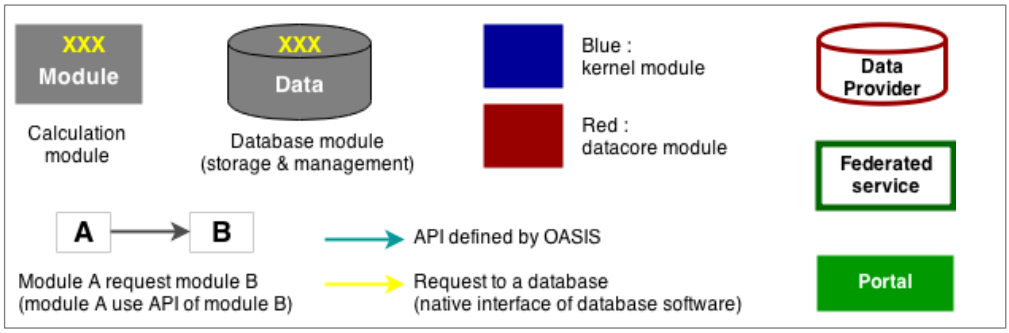
This evolution meets the following points :

- Taking into account of the RDF use and data modelling (RDF is only implemented in the form of views to consult data)
- Reducing data access layers to improve performance and not creating distinct modules when the coupling must be important.
- Transfering the control module of the access rights to data in the datacore (we originally placed it in the kernel), because after a finer analysis of rights management, it is more consistent that the datacore manages the access rights to data independently (with the social graph as support).

The division of datacore (more exactly of one datacore) into functional modules is then as follows :



Légende :



The “data homogenizer” module we defined in the first iteration will not be used by the pilot services : it is mentioned on the diagram below but will not be implemented. This is an optional functionality we will develop if it is necessary. But the principle we adopt is that the homogenization processes of data are, where possible, made by the services themselves.

The kernel LOG module is used by all the datacore modules.

OASIS datacore includes 3 databases:

- **Data containers**: internal databases of OASIS, to store data of applications. Each container can have its own administrator, its own database software, and can be localized in some specific datacenter in the OASIS cloud
- **Data repository:** this specific container contains all the repositories (vocabularies) to describe data (including RDF onthologies)
- **Data translations rules** *(will be implemented later)***:** this specific container contains rules to convert data from one repository to another, or from one format to another.

OASIS datacore is made of several modules:

- **Data access** : this module provides the federated applications and the portal with the access services to the datacore data (reading, addition, modification, deletion), with the management of the access rights, qualification, history, mediation and requests priorities.
- **Rights management** : This module analyses the access rights to data for the « data access » module with the social graph as a support. This module cannot at present be used by other sub-systems or external applications, but it can be put at dataproviders'service who would like to implement a management of rights similar to that of datacore's.
- **SPARQL Requester** : this module makes requests (in reading only) according to the SPARQL language (specific to RDF).
- **Repository Management** : this module manages diagrams of datacore data, associated RDF ontologies as well as governance rules of various data sets. This module is used at present by the OASIS data managers only.)
- **Data Homogenizer** : this module is planned for a future development and allows to access to hybrid data set (that is stored in several containers, several datacore or external data providers. It also gives access to federated data providers via datacore and converts the requests or data format.
- **External Requester** : future model, linked to « data homogenizer » and allowing to make requests to data providers.

5 Technical architecture

This section describes the OASIS technical architecture.

This technical architecture is based on the functional architecture described in the previous chapter (the sub-systems and the functional modules are technical sub-systems and modules).

Several criteria guide the definition of this architecture :

- We manage a federation of services which for some of them have their own architecture constraints and different levels of availability, security and different response times.
- We must guarantee a high availability and a good elasticity of the datacore, kernel and portal.

We then retained a REST architecture which allows a loose coupling between the sub-sytems and which offers a big independency between the sub-systems.

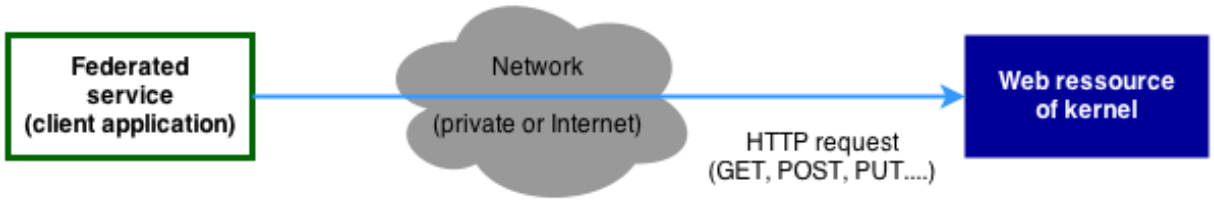
5.1 Features and benefits of REST

The principles of a REST architecture are :

- Responsibilities are divided between a client (for instance a federated service) and a server (for instance datacore) : the server provides the clients with resources.
- There's no status : a client's request must give all information to answer, the server shall not take the previous clients'requests into account. This principle allows a good elasticity so that the successive requests of a client can be treated by separate instances of the server which does not have to synchronise status or contexts between its instances.
- Caching : the result from a request can be cached and mechanisms allow to manage data freshness and the cached time limits for the storage : this reduces the number of requests to be made by the client as well as the load of servers.
- The resources provided by the server are identified in a single way, messages are self-describing.

REST is not a protocol. But REST uses the web protocols : http and its secure https implementation.

This gives the possibility of using all the Internet architecture tools (and particularly proxy and caching mechanisms) as well as http requests types (GET, PUT, POST, DELETE) to act on the resources.



The OASIS benefits are:

- Mode without status : this allows the routing of successive requests to differents modules instances without having any information session to store
- Discoverable API and upward compatible easy to guarantee
- Strong independency between the modules : the failure of a module does not block the system
- Load increase is made easier (there's no status replication between the instances of a same module, which will generate an exponential load).

The REST drawbacks do exist but are made up for by the benefits :

- Lack of status : the client must keep and send all the elements
- Loose coupling : the customer must be able to manage a lack of answer and errors.
- Heavy requests (the contents of the request being generally rather large and requests must be often implemented on several resources to make an action) : but within OASIS requests are between services (and do not come directly from the user's browser),the bit rates and the latency periods between the modules are fine.

Regarding hosting, the retained architecture allows then to manage in a total independent way the various sub-systems (the only limit being the bandwidth and the latency of transport of the messages exchanged between the sub-systems).

5.2 Final architecture

With the choice of REST architecture, every functional module can be instantiated several times on several servers and in several datacenters to manage the load and the availability of the solution.

The data storage requires databases with two major properties :

- Elasticity (the increase of the processing volumes)
- Availability, that is the possibility of losing a server or even a complete datacenter without losing data and even without losing the access to data.

The solutions of data storage retained shall then allow :

- A storage divided into several servers
- A redundant server on several servers and even several datacenters

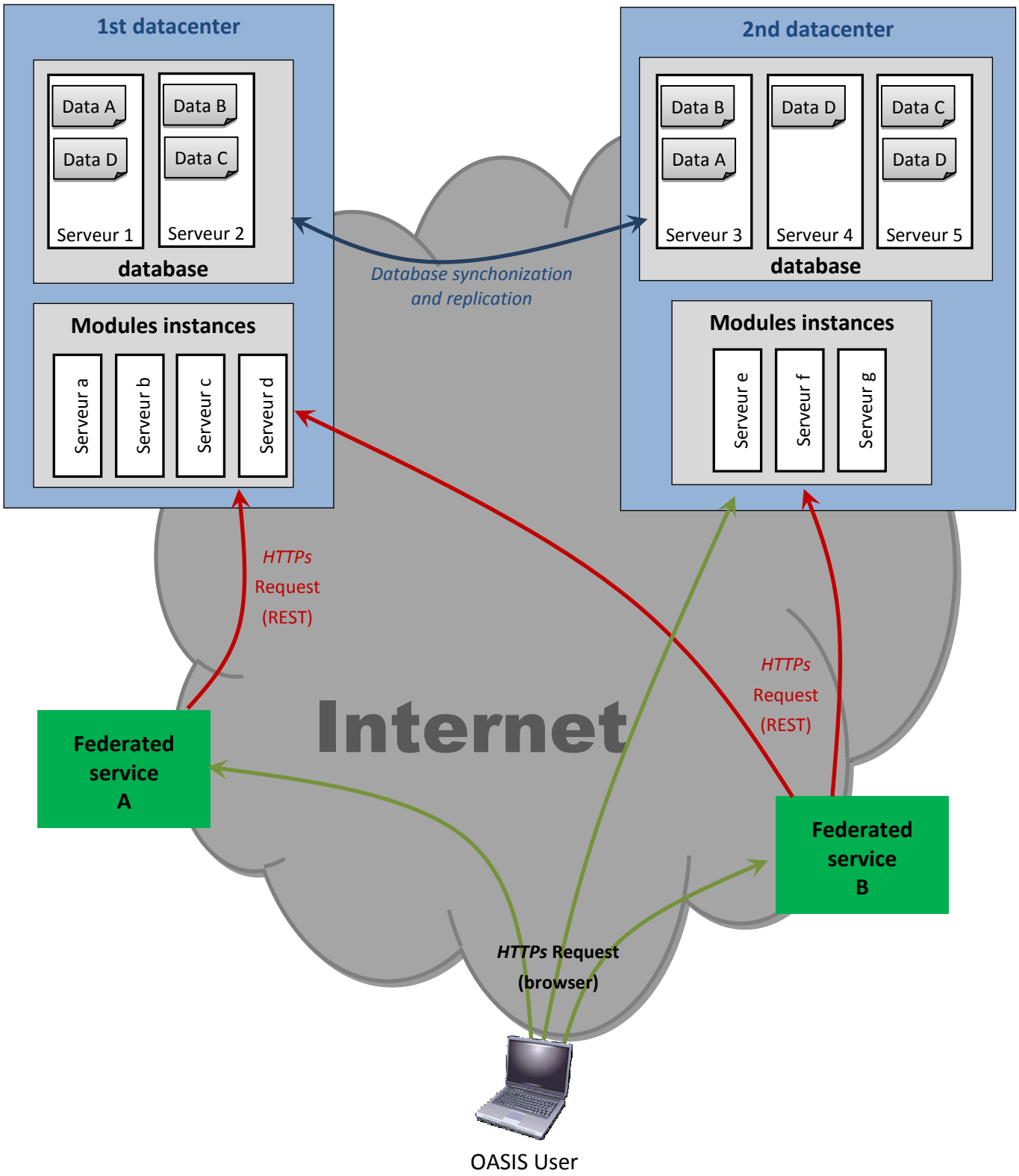
Several solutions of database meet these criteria : this point is then not managed at the level of OASIS architecture but in the implementation choices (document D2.1).

Plesae note that the access to the various databases is systematically made by a datacore or kernel dedicated module. The database and the access module are strongly dependent, their coupling must not respect the REST principles : this coupling will be made according to the principles and protocols of the databases solutions retained (as mentioned in the functional architecture diagrams).

The following diagram summarizes the OASIS technical architecture :

For simplification (and then legibility) measures :

- Only one datacore and two datacenters are mentioned on this diagram
- Only one database (linked to datacore) is represented : the principle is the same for the other databases
- We do not detail the kernel and datacore modules
- We only mention two federated services regarded as being hosted on two separate « clouds » and without giving details of their architecture.
- During the implementation, the services who want to can be hosted on the clouds used for the kernel, the portal and datacore. The cloud and security management equipments are not represented (they will be represented on the implementation document : D2.1).


1st datacenter
Data A
Data D
Data B
Data C
Serveur 1
Serveur 2
database
Modules instances
Serveur a
Serveur b
Serveur c
Serveur d
2nd datacenter
Data B
Data A
Data D
Data C
Data D
Serveur 3
Serveur 4
Serveur 5
database
Modules instances
Serveur e
Serveur f
Serveur g
Database synchonization and replication
HTTPs Request (REST)
HTTPs Request (REST)
Internet
Federated service A
Federated service B
HTTPs Request (browser)
OASIS User

Security:

Security is managed by several architecture principles :
Use of SSL protocols (Secure Socket Layer) to encrypt and authenticate the exchanges.

- Use of a strong mechanism of authentication and authorization tickets (see D2.1 et D2.3)
- Identification of services exchanging between them

Elasticity:

The architecture defined allows to simultaneously integrate several mechanisms to ensure security, at a lesser cost for some of them (that is without unnecessary burden on the infrastructure hosting OASIS) :

- Shards and/or implementation of Master/Slave replications on database.
  The distribution of data on datacore and containers, directly accessed by the user services (then with no complex routing mechanisms)
  Direct use of DNS for the localisation of services with use of GeoDNS and Round Robin functions to route the requests intelligently.
- Management of the high availability at the level of the DNS (allowing to support the complete loss of a datacenter)
- Use of the BGP dynamic routing to manage multiple Internet connections to datacenters and reroute one datacenter on another.
- Load distribution at the HTTPS level at the entry of each datacenter (knowing that the loose coupling allows to process the successive requests on different servers)
- Virtualization mechanisms to quickly develop the available power.

Services use their own mechanisms to guarantee elasticity and security.

Three main hypotheses can be used (and will be implemented within the context of the pilot):

- Services hosted on their own cloud architecture (private, public or both)
- Services hosted on a more classical architecture (physical or virtualized servers, redundant or not according to the level of service required), with a new instance for every organisation
- Services hosted on the OASIS cloud

Remark on the cloud management:

The architecture retained allows to use virtualized or physical servers.

The cloud computing technologies furthermore include mechanisms allowing to automatically provision new servers (or to stop some). Modules must then integrate the possibility of being reconfigured in a dynamic way to modify the number of instances.

Within the context of the pilot, we will use virtualized servers (to benefit from the flexibility of this technology). The modules will be developed to integrate the automatic reconfiguration (and then the addition or deletion of instances without an interruption of the service).

We will however not use the automatic provisioning : it seems more interesting to finely manage the resources forecast. Besides, an automatic provisioning is technically more flexible but can generate non anticipated costs.

6 Vigilance points, risks

At this stage of the project, technical risks have been identified, and must be particularly followed.

This section presents these risks, their measures during the experimental phase, and the first thoughts for the solutions which can be brought according to the cases met.

The risks studied in this section are the specific risks with OASIS: we do not refer to the general problems of security, bugs, load system, etc. …

6.1 Data access performances

6.1.1 Description of the risk

OASIS complicates data access, by the fact that the interfaces have to pass through web-services which add a layer.

Besides, the applications refer at present to a data model specifically designed for their processings, while the OASIS model is wider and is not optimized for specific businesses.

The access procedure thus contains several stages, which may lengthen processing times. It will be compensated with the efficiency of the retained technologies, and the computing power of the cloud computing platform.

It is not possible to determine a priori the influence of both factors because it totally depends on the types of requests and on applications.

But there is a risk that some requests have insufficient performances to offer acceptable response time.

6.1.2 Monitoring

This risk will be watched in different ways:

- Regular tests of access to data by the teams in charge of the OASIS development Description of problems of performances by service providers
- Measure of average response time to the requests at the level of the interfaces between OASIS and the applications: these measures will be directly implemented in modules, and put at the disposal of the monitoring platform via the module of journaling
- Satisfaction surveys made within the WP 5 framework.

6.1.3 Solutions which can be implemented

In case of problem of performances, we can work on the following levers:

- Simplifying the detail of the rights management made by the datacore for the reliable applications
- Implementing specific requests which do not need perfectly updated data, and which can then work on caching
- Work on indexes (strong indexation of the relations and the links), which implies bigger volumes of information to be stored
- Separation of data in smaller (and more numerous) containers
- Preprocessing of common requests
- Sharp regulations of database technical parameters
- Work on the requests prioritization (a mechanism of prioritization will be integrated for the requests on data)
- Work with the services publishers to optimize the requests and the logic of access to data

6.2 Emergence of doubles

6.2.1 Description of the risk

Considering the multiple vocabularies and the presence of several containers, the data which represent in fact the same entity can be registered as different entities in OASIS. The problem is that it has an impact on the quality of the data, and the analysis which can be made: countings, accumulations or statistics can be falsified, or even attributes can have different values according to which copy an application works on.

6.2.2 Monitoring

This risk will be watched in different ways:

- Descriptions of problems linked to this subject by service providers
- Evaluation of the creation requests rate having indicated the verification of doubles
- Regular search requests for potential doubles

6.2.3 Solutions which can be implemented

When faced with the apparition of doubles, we can work on the following levers:

- Implementing a doubles search before any creation of data at the level of the datacore, and managing an alert (even a rejection of the request)
- Setting up search tools for doubles, mediation (to validate the right data), and for merger of two entities
- Work with the services publishers to help them having a data processing logic which avoids doubles

6.3 Problem of Data quality

6.3.1 Description of the risk

The data being managed in a collaborative way, through applications having various levels of verification, there will be problems of accuracy of the information (as for Wikipedia, for example).

Qualification integrated mechanisms are intended to manage it.

But the cohabitation of purely informative and business practices on the same data is new: we thus have to validate that the mechanisms implemented to take the quality of the data into account are sufficient.

6.3.2 Monitoring

This risk will be watched in different ways:

- Description of problems linked to this subject by service providers
- Control of the average qualification rate of data (average value of the global quality index, taking all objects together)
- Control of the problems occurring on qualified data
- Control of mediation actions on data

6.3.3 Solutions which can be implemented

In case of problem on the quality of data, we can work on the following levers:

- Strengthening the mechanisms of the qualification management, and refining the management rules of these information
- Working with the services publishers to help them improving their management of the data quality.

6.4 Security breach on private data

6.4.1 Description of the risk

The management of private data, and the security on these data is a critical subject for OASIS. We have to guarantee that the data are used only with the agreement of their owner, and only for authorized processings (no illegal crossing of data for example).

It is thus imperative to make sure that the implemented solutions are correct (that they do not introduce breaches due to their design), and that they are correctly implemented (that there is no bug opening breaches).

6.4.2 Monitoring

This risk will be watched in different ways:

- Implementation of a form so that the users can indicate their suspicions of data leaks
- Description of problems linked to this subject by service providers

- Statistical analysis on the accessed data to see if the accesses are not abnormally numerous or large.

6.4.3 Solutions which can be implemented

In case of breach on the confidentiality of private data, we can work on the following levers:

- Strengthening access restrictions (knowing that on this subject, it will be necessary to implement strong limitations and rather open accesses if we notice that they are too much restricted)
- Strengthening of the design of the security mechanisms·
- Strengthening of the security checks implemented in the programs
- Increase of the separation of data (necessity to pass by several functional chains, accessing to different bases, to reconstitute data)

6.5 Integrity problems

6.5.1 Description of the risk

As in any database, there can be errors of integrity.

With traditional databases, numerous rules can be integrated at the level of the database server itself to control the integrity

Within the framework of distributed, redundant architecture managing «big data ", it is necessary to make compromises between the performances and the controls, and the design of the database management systems increases the risk.

6.5.2 Monitoring

This risk will be watched in different ways:

- Control of the errors which can be linked to problems of integrity
- Descriptions of problems linked to this subject by service providers

6.5.3 Solutions which can be implemented

In case of too frequent integrity problems, we can activate the following levers:

- Reworking the compromises made between the performance constraints and the database ACID properties.
- Working with the SGBD developers to improve the solidity of the solutions.
- Strengthening of the integrity control mechanisms
- Increase of the data separation (necessity to pass through several functional chains, accessing to different database to reconstitute data)
- Working with the service publishers to improve the verifications they made upstream

6.6 Inaccessible data

6.6.1 Description of the risk

Some data can become inaccessible, in reading and/or writing because the access restrictions deprive everyone of one's rights.

6.6.2 Monitoring

This risk will be monitored in different ways:

- Description of the problems linked to this subject by service providers
- Sampling audit of data and access rights

6.6.3 Solutions which can be implemented

In case of frequent recurrence of this risk, we will be able to act on the following levers:

- Controls when affecting the rights
- Implementation of mediation mechanisms on the accesses to data to restore rights

6.7 Problems of adaptation of the applications

6.7.1 Description of the risk

The strong integration of the applications (services) requires to strongly develop the logic of access to the data, and even the processes of modification or creation of data. Furthermore, research requests working on a very wide basis, it is necessary to put more complete selection criteria than the values which existed when the application was isolated.

The choices of architecture and functionalities integrate solutions to limit the impacts on the applications (in particular by planning default values and constraints and attributes to be automatically added during the various requests).

The pilot will have to validate that these solutions are necessary and sufficient.

6.7.2 Monitoring

The frequency of this risk will be reported by service providers.

6.7.3 Solutions which can be implemented

When such problems occur, we can act on the following levers:

- Addition of mechanisms allowing the automatic adjustment of the service requests with rules per service, per functional chain, etc.
- Reduction of the data scope managed in OASIS (the service keeps a database for part of the data and registers it in OASIS as data provider)
- Work with the service publishers to help them improve their integration in OASIS

This risk decreased in this second iteration: some functionalities have been modified following the work done on the service adaptation (and especially data modelling in the datacore).

7 Conclusion

To design the architecture described in this document, we have carefully analyzed the consequences of the requirements, the contradictions between some of them, the need to supply users and providers on-demand services with a guarantee of security, integrity and performances.

The designed architecture resets data at the center of the ecosystem favoring their reuse, their pooling, their opening and the constitution in a collaborative way of a common legacy.

The fluidity of the processes has not been omitted with a notification module which allows to build an advanced workflow.

We defined an architecture where the OASIS security and upgradability can be managed.

With the specialization of databases and the notion of container we can improve the security management, optimize the performances (by splitting data in several databases, and by choosing the technologies according to the types of main requests), and better adapt to the applications which are not ready for a share of data.

The architecture also allows a progressive integration of the services.

We particularly made sure that OASIS is endowed with an intelligence and with advanced collaboration functionalities, remaining centered on data so as to treat all businesses correctly without including a business logic in OASIS, and by leaving the system neutral, capable of welcoming numerous services, favoring the competition, and helping the creation of new innovative services based on data and interoperability.

This architecture is also intended to promote dialogue between the OASIS teams and services publishers, to move progressively towards a deeper integration, promote consistency in the ecosystem, and to encourage collaboration and collective intelligence : enrollment in catalogs and data modeling are also grids of reflection to build an effective ecosystem.

The requirements, and therefore the architecture, are compliant with the OASIS vision.

Requirement (D1.1)	How we comply them
Data managed in the social graph : • Person management • Organization management • Group management • Relationships management between organizations / persons • Family home management • Management of underage users	These requirements are the description of the social graph entities and relations. Underage users are manged with delegation feature. See paragraph 3.2, and deliverable D2.3
Data managed by the OASIS application store (catalog) : • Service providers • On-line applications • Data sets • Applications price • Categories • Notes and user review	These data are stored in the catalogs (database of the kernel) and manage by the module “broker”. Users access to these data through the portal See Paragraph 3.4.3
Collaborative and shared data : • Data model • Metadata model • Data storage Data quality • Dedicated onthology • Data moderator Data historization	Datacore complies these requirements (see chapte 3.3 and 3.5) Data model are managed by the module “Repositoru Management”, in the database “Data repository” Metadata are the RDF model to extend natives datac schemes. Data quality and moderator are features of datacore (see 3.3.8 and 3.3.10). Historization is described in paragraph 3.3.9
Authentication system : • Unique authentication system • Possible double authentication procedure • Secured password recovery procedure	Authentication is managed by a specific module of kernel. We describe theses features in paragraphes 3.4.1, and more precisely in delivable D2.3. Authentication system is based on social graph.
Access rights managed by the social graph : • Delegation • Unidirectional links • Nodes and links privacy • Link creation • Role and context management	Social graph features, especially privacy features, are investigated in delivrable D2.3. This document addresses theses topics in paragraph 3.2. Context management is addressed in paragraph 3.4.2
Data access rights : • General access rights on a container • Access rights on a data type • Access rights on a data scope • Access rights for a user/a group • Access rights according the context	Access rights in datacore investigated in delivrable D2.3. These features are based on social graph, and use « social graph requester » and « authentication » modules. This document addresses theses topics in paragraphes 3.3.6 and 3.3.7.

Requirement (D1.1)	How we comply them
Notification system : • Event notification by an application • Service notification by the OASIS system • Management of data status • User notification	Notification system is addressed in paragraphes 3.4.4 and 3.4.5 These features are implements by two modules : “Notification and status flow”, and “communication” (in the kernel)
Log management : • Event registration (audit) • Event classification • Statistics of use	Log and audit management is a dedicated module of the kernel, with a specific database. This topic is addressed in paragraph 3.6
Development environment	Development environment is not directly addressed in the architecture : it’s in fact a second implementation of the architecture, for developping and testing
Cloud hosting : • Under European law • Distributed by geolocalized	These topics are addressed in paragraph 5.2, and mainly in D2.1 deliverable.
Governance : • General governance (data model) • Local governance (local law, data collaboration) • Terms of Use	OASIS can be hosted in several country, and we can deploy specific datacore for a country. Data models are managed in the datacore. A specific feature, “scope of governance” has been introduced in the datacore to insure data governance (see paragraph 3.3.6) Term of use will be stored in catalogs and display by the portal.
User friendly interfaces to get informed, access and use all the available resources and features, available on every kind of devices, with language management	These features are addressed by the OASIS portal. This portal use mainly authentication, social grah requester, and broker (catalogs)
An application store with search engine, ability to filter results, in the appropriate language	This feature is addressed by the portal and by the broker (in the kernel)
System of evaluation of the applications by the users, and mediation between the users and the suppliers	
A personal work environment displaying : • targeted services (local public services and shared services), according to the user location or role • services selected from the store, • a notification centre, • service consumption, billing history • user profiles and personal data • a social graph creator allowing users to link his account with another one, • contexts to manage several role in one account	These features are addressed by the OASIS portal. For these features, it mainly uses authentication, social graph, notifications and status flow. Context management is addressed in the paragraph 3.4.2, and fully described in the deliverable D2.3,
A digital safe allowing complete dematerialization of procedures	These features are addressed by specific federated services, and by the portal

Requirement (D1.1)	How we comply them
A data store	This feature is addressed by the datacore (through federated services) and by specific “open data” federated service.
Detailed information sheet describing every resources (services and data)	This information is displayed by the portal, and managed by the broker (kernel’s module) in the catalogs.

8 References

Books :

- « Le Web Sémantique », written by Fabien Gandon, Catherine Faron-Zucker and Olivier Corby, published by Dunod [ISBN 978-2-10-057294-6]
- « Practical RDF », written by Shelley Powers, published by O'Reilly [ISBN 978-0-596-00263-3]
- « Semantic Web for the Working Ontologist », written by Dean Allemang and Jim Hendler, published by Morgan Kaufmann [ISBN 978-0-12-385965-5]

Interviews with experts outside the consortium :

- Professor Djamel Abdelkader Zighed, from the Institute of Human Sciences (CNRS), expert in Representation and Knowledge ingeneering
- Professor Jérome Darmont, from ERIC lab (Warehouses, Representation and Knowledge ingeneering, Lyon 2 University)
- Professor Dimitris Kotzinos, from the Cergy Pontoise University, who is involved in the European InGeoCloudS CIP project.
- Mr Bernard Vatant, from Mondeca, expert in RDF modelization
- Mr Nicolas Chauvat, from Logilab, expert in semantic web technologies

W3C Workgroup :

- Linked Data Platform Workin Group, whose mission is to: produce a W3C Recommendation for HTTP-based (RESTful) application integration patterns using read/write Linked Data (http://www.w3.org/2012/ldp/wiki/Main_Page)

Other Projects :

- STORK 2.0 EU project. http://www.eid-stork2.eu/

W3C Standards :

- [RDF97] RDF First Public Draft, 1997-10-02 http://www.w3.org/TR/WD-rdf-syntax-971002/
- [RDF97] RDF First Recommendation, 1999-02-22 http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/
- [RDF04] RDF Current Recommendation, 2004-02-10 http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

- [RDFS] RDF Schema Recommendation, 2004-02-10
  http://www.w3.org/TR/rdf-schema/
- [OWL] Web Ontology Language overview, 2004-02-10
  http://www.w3.org/TR/owl-features/
- [SKOS] Simple Knowledge Organization System, Recommendation 2009-08-18
  http://www.w3.org/TR/skos-reference/
- [SPARQL] A Query Language for RDF, Version 1.1, Recommendation 2013-03-21
  http://www.w3.org/TR/sparql11-overview/
- [RDFa] RDFa 1.1 Primer, 2012-06-07
  http://www.w3.org/TR/xhtml-rdfa-primer/
- [TURTLE] Terse RDF Triple Language, Candidate Recommendation, 2013-02-19
  http://www.w3.org/TR/turtle/

Ontologies used for scial graph study :

- WAI (Roles and Profiles Ontology) : http://vocab.ctic.es/wai/wai.html
- ORG (Core Organization Ontology) : http://www.w3.org/ns/org
- REL (Relationship Ontology) : http://vocab.org/relationship/.html
- GEN (Linked Genealogical Data Vocabulary) : http://purl.org/gen/0.1
- FOAF (Friend of a Friend Vocabulary) : http://www.foaf-project.org/

And we have extensively used the LOV project site

- LOV (Linked Open Vocabularies) : http://lov.okfn.org/dataset/lov/index.html

9 Annex 1. Glossaire

Term	Description	Comment
ACID	ACID is the acronym for Atomicity, Consistency, Isolation, Durability, the 4 properties of a database to insure integrity of data.	Distributed database are in general not ACID, we have to evaluate these properties to select the DBMS
API	API is the acronym for Application Programming Interface : we use the term API to describe protocol and language to use a service provided by a program	
Availability	Availability is one of the three security criteria: availability means that data and services can be used when need. We can measure it with a maximum delay of recovery, and with a percentage of time (a day, a month, a year) when the service or the data is unavailable.	
Broker	OASIS module design to search resources in catalogs	
Catalog	List and description of resources registered in OASIS	
Cloud computing	Cloud computing is a general term for everything that involves delivering large scale and distributed hosted services over the Internet	
Confidentiality	Integrity is one of the three security criteria : integrity means data (or processing, or transmission) can be alter or compromise	This properties is very important for personal data
Data homogenizer	Specific service of OASIS designed to access external or aggregated data, in the same way as internal data	
Data provider	This term refers to the external data source, the technical resources to store and manage it, the API to access to these data, and the supplier itself.	
Data repository	The vocabularies (or ontologies) describing data.	The word repository in English can mean “store” : we don’t use this meaning.
Database	A database is a structured set of data stored on computers or a dedicated storage system	

Term	Description	Comment
Database Management system	A database management system (DBRM) is a software to manage database, to interpret an run requests, to secure data and to optimize access time	We have to choose several DBMS for OASIS
DBMS	DBMS is the acronym for DataBase Management System (see definition)	
Dependency	Dependencies of a module (or of a component) are the other module or the external resources it use	
Elasticity	Elasticity is a property of a system : it means the ability to sustain large variation of load or of data volume, with the same performances. Elastic systems must be able to be operate of large scale infrastructure, with a lot of servers in several points. Specific technologies exists to insure elasticity	Elasticity is a very important requirement of OASIS
External interface	An interface is called "external" when it can be used by external applications	
GUI	GUI is the acronym for Graphical User Interface. This means all the graphical screens, and their sequences, to display information to users and to allow him entering data.	GUI of OASIS is the portal. Each external applications of the federation has its own GUI
HTTPS	HTTPS is the secure (authenticated and crypted) protocol to access to web resources (web pages, web services)	
IAAS	IAAS is an acronym for Infrastructure As A Service. It's a cloud architecture where the cloud middleware and hardware provide only virtual servers and its management tools.	OASIS uses a IAAS cloud to provide Softwares as Services (SAAS) to end users.
Integrity	Integrity is one of the three security criteria : integrity means to avid data (or processing, or transmission) being altered or compromised	
Interface	An interface is the description of the interaction between two modules. One module provide some API, these API are used by the second module	
Internal interface	An interface is called "internal" when it can be used only by OASIS modules	
Log	A log is an event added in a file or a database, to have an historical of actions, events, or errors	
Metadata	Specific information add to application data to provide a high level of description of these data, and additional information (eg: semantic information, access rights, etc…)	OASIS use RD model for metadata

Term	Description	Comment
Middleware	A Middleware is a software providing specific services for applications (it's a middleware because it's between the operating system and the applications programs)	
Module	We can divide a complete system in subsystems, and a subsystem is divided into modules. Modules can be divided into components. A module provides a coherent set of capabilities.	
OASIS datacore	The OASIS data core is the subsystem to store, manage, and secure centralized data, and the set of services to access to these data.	
OASIS ecosystem	We call "OASIS ecosystem" the OASIS system itself, plus service providers and data provider	
OASIS kernel	The kernel is the subsystem providing all technical capabilities of OASIS	
OASIS portal	The portal is the GUI of OASIS	
OASIS system	OASIS system is the portal, the kernel and the data core	
PAAS	PAAS is an acronym for Platform As A Service. It's a cloud architecture : the cloud middleware and hardware provide high level services to implement applications (it provides database management systems, java virtual machine, inter components communication module, etc...)	
RDF	RDF is the acronym for Resource Description Framework. It's a model describing linked data, semantic web and metadata. See appendix and section 4	
REST	Representational state transfer (REST) is an architectural style	OASIS is based on a REST architecture
Request language	A request language is a specific language, with a vocabulary and a grammar, to access to a database (search and filter, update, etc…)	
SAAS	SAAS is an acronym for Software As A service. It's a cloud architecture where the cloud supplier provide directly services to end user.	OASIS uses a IAAS cloud to provide Softwares as Services (SAAS) to end users
Service	A service is a set of capabilities used through explicit defined interfaces	
Service provider	This term means an external application, the technical resources to operate it, the GUI to use it, and the supplier itself.	

Term	Description	Comment
SPARQL	SPARQL is a recursive acronym for SPARQL Protocol and RDF Query Language. It's a specific request language for data modeled in RDF	
SSL	SSL (Secure Socket Layer) is the protocol used by HTTPS to secure the transmissions	
Subsystem	A subsystem is a slice level, used to describe a complete system in a more comprehensive way. A subsystem is divided into modules.	
Translation rule	Rule to convert data from a format to another, or from a model to another	
Vocabulary (= ontology)	Set of information used to describe the semantic meaning of data in a particular scope	
Web service	A web service is a specific service (an application) with an interface (API) using web protocols. It can be used easily through Internet by other applications.	

10 Annex 2. Modèle RDF

RDF data model

The RDF data model is based on triples (**s**, **p**, **o**) where:

The ***subject*** **s** represents a « thing » which is described. The triple can be seen as an elementary « molecule » of information about its subject.

- The ***predicate*** **p** represents a property applying to the subject. It is generally defined in a vocabulary, using one of the standards RDFS or OWL.
- The ***object*** **o** is often called the *value* of **p** (provided by this triple)

s and **p** are ***resources*** and are generally identified by a URI.

o can be a ***resource*** (identified by URI, or blank node), or a ***literal*** (plain data, recommended data types are defined by XML Schema such as string, numeric, date...).

A ***blank node*** is a resource which has no global identication (no URI).

The built-in predicate ***rdf:type*** links a resource to a declared type or class. The value (object) of a triple of which predicate is rdf:type is typically a class in a vocabulary (or ontology) using one of the standard ontology languages RDFS or OWL.

RDFS and OWL

A ***RDF vocabulary*** (or ontology) defines a set of predicates and classes, and the formal axioms which link them together such as hierarchy of classes and properties, constrained classes for the subject or object of a given predicate, cardinality constraints (mandatory or unique property).

The axioms of a vocabulary can be expressed using a variety of logical models. The W3C recommendations RDFS and OWL provide the logical foundation and the syntax for such models. RDFS provides basic expressivity for simple vocabularies, and OWL provides more advanced constructions for applications needing advanced reasoning capacities.

SPARQL

SPARQL is a specific language to make request on RDF database.

Example of RDF graph and query

The following graph represents an organization, its site and and one of its members, and a project in which he is involved.

Blue rounded rectangles are *individual entities* displayed by their name, each each of them will be represented in RDF by a URI (excepted the « blank » address)

Pink ellipses are the *types* of the entities, they will be represented by *classes* defined in shared vocabularies.

Yellow rectangles are *typed links* between entities, they will be represented by *object properties*, also defined in shared vocabularies.

Purple rectangle is a *datatype property*, also defined in a shared vocabulary

The same graph is now presented in RDF, using Turtle syntax.

Prefixes are defined for external vocabularies, used to describe the data. All those vocabularies are fully available using HTTP GET on their URI (or namespace). The last prefix is for the linked data (this is a placeholder, the data are not actually published)

@prefix dcterms:<http://purl.org/dc/terms/>.
@prefix foaf:<http://xmlns.com/foaf/0.1/>.
@prefix org:<http://www.w3.org/ns/org#>.
@prefix igeo:<http://rdf.insee.fr/def/geo#>.
@prefix swpo:<http://sw-portal.deri.org/ontologies/swportal#>.
@prefix oasis:<http://data.oasis-eu.org/>.

The commune of Valence. A full description is provided by INSEE at this URI.
<http://id.insee.fr/geo/commune/26362>
a igeo:Commune;
igeo:nom "Valence";
igeo:codeINSEE "26362".

The organization "Pôle Numérique"
oasis:org-PoleNumerique
a org:Organization;
foaf:name "Pôle Numérique";
org:hasSite oasis:site-Rhovalparc.
The physical site of this organisation, with its address. The address is described as a « blank node » (no URI)

```
oasis:site-Rhovalparc
        a org:Site;
        foaf:name "Rhovalparc";
        swpo:hasAddress
         [ a swpo:PostalAddress;
         swpo:hasStreetAddress "1 avenue de la Gare - Allée B - B.P. 15155 - 26958 VALENCE
CEDEX 9";
         swpo:inCity <http://id.insee.fr/geo/commune/26362> ].
### A member of this organization
oasis:person-BrunoThuillier
        a foaf:Person;
        org:memberOf oasis:org-PoleNumerique;
        foaf:name "Bruno Thuillier";
        foaf:currentProject oasis:project-OASIS.
### The OASIS project
oasis:project-OASIS
        a foaf:Project;
        foaf:name "OASIS";
        foaf:homepage <http://www.oasis-eu.org/>.
```

A SPARQL query can now be ran on this graph, such as:

"Select ***people*** *member of an* ***organisation*** *based in Valence (INSEE code 26362) and* ***projects*** *they are involved in*"

```
SELECT ?person ?organization ?project
WHERE { ?p a foaf:Person.
        ?p foaf:name ?person.
        ?p foaf:currentProject ?pr.
        ?pr a foaf:Project.
        ?pr foaf:name ?project.
        ?p org:memberOf ?org.
        ?org a org:Organization.
        ?org foaf:name ?organization.
        ?org swpo:hasSite ?site.
        ?site swpo:hasAddress _:b.
        _:b swpo:inCity ?city.
        ?city igeo:codeINSEE "26362". }
```

The results will be as following.

?person	?organization	?project
Bruno Thuillier	Pôle Numérique	OASIS