Streamer
European research on energy-efficient healthcare districts

D5.3

PLM architecture for EeB within the context of healthcare districts

Deliverable Report: D5.3 Final version

STREAMER - Optimised design methodologies for energy-efficient buildings integrated in the neighbourhood energy systems.
The STREAMER project is co-financed by the European Commission under the seventh research framework programme FP7.EeB.NMP.2013-5 ; GA No. 608739)

D5.3

PLM architecture for EeB within the context of healthcare districts

Main author Audrey Vial (CST)
Co-authors Nick Nisbet (AEC3), Joachim Benner (KIT), Pim Vandenhelm (TNO), Rizal Sebastian (DMO)
Dissemination Public

Document history

Version	Date	Status	Produced/Reviewed by	Comments
1	2015/04/xx	Table of contents	Audrey Vial: reviewed by Matthias Weise (AEC3), Pim Vandenhelm (TNO), Karl-Heinz Haefele (KIT)	
2	2015/08/07	Final draft	Audrey Vial	
3	2015/08/10	AEC3 contribution to the final draft	Nicholas Nisbet (AEC3)	
4	2015/08/15	Co-review	Rizal Sebastian (DMO)	
5	2015/08/17	Co-review	Jan-peter (DWA)	
6	2015/09/2	Last review	Freek (TNO)	

Colophon

The STREAMER project is co-financed by the European Commission under the seventh research framework programme with contract No.: 608739 - FP7-2013-NMP-ENV-EeB. The information in this publication does not necessarily represent the view of the European Commission. The European Commission shall not in any way be liable or responsible for the use of any such knowledge, information or data, or of the consequences thereof.

Deliverable summary

Define PLM architecture based on Semantic Web technologies, in order to represent the semantic assertions and constraints, as well as to couple the fine access right management that represents the core of any PLM solution with the BIM. As a result, this deliverable aims at presenting an advanced PLM solution that manages both access rights and views of users when accessing the information models according to their profile/ expertise/ project context together with parametric information about the objects they are accessing.

List of acronyms and abbreviations

- AEC: Architecture, Engineering and Construction
- API: Application Programming Interface
- BCF: BIM Collaboration Format
- BIM: Building Information Modeling
- BIMSie: BIM Service Interface Exchange
- CAD: Computer-Aided Design
- CMIS: Content Management Interoperability
- EDM: Electronic Document Management
- gbXML: The Green Building XML
- GIS: Geographic Information System
- GML: Geography Markup Language
- GUID: Globally Unique Identifier
- IFC: Industry Foundation Classes
- MOA: Maîtrise d'Ouvrage / Project Owner
- MOE: Maîtrise d'Oeuvre / Project Supervisor
- PLM: Product Lifecycle Management
- SVN: Version Control System
- WFS: Web Feature Service

Definitions

CityGML: GML Profile for representing the built environment in 3D.

gbXML: Open schema defined to facilitate the transfer of building properties stored in 3D building information models (BIM) to engineering analysis tools.

mvdXML: Model View Definition XML file that is used to define a subset of IFC schema needed to satisfy one or many Exchange Requirements of the AEC industry. Together with the IFC schema subset, a set of implementation instructions and validation rules, called **MVD Concepts**, are published. The method to publish the concepts and associated rules is **mvdXML**.

Contents

Table of illustrations

1. Introduction

1.1 Objectives of the WP5.2

The goal of WP5.2 is to develop an advanced Product Lifecycle Management (PLM) solution in order to capture and support the organizational and workflow processes based on knowledge and optimization of products (like equipment, installations, utilities, and buildings) throughout their lifecycle.

1.2 Problem statement

1.2.1 Explanation of the STREAMER problem

The global objective of STREAMER is to reduce by 50% in the next 10 years the energy use and carbon emission of new and retrofitted buildings in healthcare districts in the EU. To carry out this project, analyses have to focus on two levels: building and neighborhood scales which imply basing analysis on Building and Geo Information Modelling. Another important aspect of the project is the number of actors. It involves clients, architects, technical designers, contractors, building operators and occupants all in charge of designing energy-efficient buildings using optimizes semantic-driven design methods and interoperable tools for Building and Geo Information Modelling. According to this huge number of actors, it appears fundamental to define and use organizational tools and workflows well known by PLM software's. The STREAMER project is so a huge ICT integrator that combines BIM, GIS and PLM technologies and whose objective is to make interact as many software's as actors.

1.2.2 Building Information Modeling status

Building Information Modeling (BIM) is a complex technology to enforce into old practices. Nevertheless the Building Industry feels the necessity to better organize around collaborative processes to face multiple issues:

- The control of information becomes unclear because all building information is put together into one model
- Energy profitability and efficiency seem unavoidable (better competitiveness and quality is required)

The IFC ("Industry Foundation Classes") are information which allows describing the objects we need to design during the life cycle construction of a building according to various points of view (architecture, structure, thermal…). For each element of the building the IFC gives indications onto the shape, the characteristics and the relations with the other objects.

The IFC format guarantees the interoperability between the CAD software and the engineering ones. This exchange format enables a better coordination between design offices and structural engineers. Moreover it allows to easily isolating the differences between various versions of files.

Nevertheless to be optimal and lead to a maximum of results, BIM requires from its users an integral and collaborative work style. The main advantages of integrating BIM into collaboration AEC practice are the following:

- Increase of facility documentation value (due to handover from one phase to the next, or from one actor to the other)
- Reduce redundancy in the facility documentation production (reduce man/hour effort to produce and maintain the information during all the project phases)

1.2.3 Geographical information systems

In design and construction of large scale industrial installations, PLM systems are also commonly applied alongside GIS tools. This need arises from strong requirements for managing machine engineering processes as well as geographic distribution of extended facilities. As both of these aspects are subject to the same planning process, efforts in PLM and GIS domains need to be coordinated. The process can thus benefit from integration of supporting systems. The architecture proposed in this document will also integrate the conceptual model based on reference PLM and GIS data model, workflow and user interface level.

1.3 Challenges

1.3.1 BIM approach

BIM focuses on the functions of visualization, properties computation and interaction checking between its various building elements. It also allows the collaboration between the various actors of a building construction.

Nevertheless, it does not handle the management of documents nor check their states or versions. That is the reason why the BIM needs a complementary solution to insure a good management of the life cycle of a building project.

1.3.2 GIS approach

From a software perspective, a GIS consists of a special type of computer program capable of storing, editing, processing, and presenting geographic data and information as maps.

Spatial data refer to the real-world geographic objects of interest, such as streets, buildings, lakes, and countries, and their respective locations. In addition to location, each of these objects also possesses certain traits of interest, or attributes, such as a name, number of stories, depth, or population. GIS software keeps track of both the spatial and attributes data and permits us to link the two types of data together to create information and facilitate analysis. One popular way to describe and to visualize a GIS is picturing it as a cake with many layers. Each layer of the cake represents a different geographic theme, such as water features, buildings, and roads, and each layer is stacked one on top of another.

1.3.3 PLM approach

The PLM formalizes the processes of validation, exchange and modification. It allows the management of the life cycle of a project construction project and strengthens an overall view of the data and documents of the project.

Nevertheless, it does not allow representing graphically the building. Moreover to face the increase of actors who ask to reach the building model during all the phases of the project, the PLM feels the necessity to connect to the digital model BIM.

1.3.4 Coupling BIM, GIS and PLM

An application coupling BIM, GIS and PLM concepts allows to formalize and to streamline the exchanges between several companies around the digital model BIM, the GIS spatial data and around all the data of the project, by protecting the management of the various actors' rights during the whole life of the project.

Such an approach brings undeniable assets:

- Insurance of a global management during all the life cycle of the project
- Collaborative processes and traceability of the exchanges
- Guaranteed that there is no incoherence of size and management
- Professional sector interface by user profile (day tasks, targeted part of the IFC file)
- Evaluation of the impacts during changes and evolutions
- Alerts subscription to people concerned by modifications

No exhaustive solution coupling BIM, GIS and PLM already exists. More and more efforts are done from PLM side on one hand and BIM/GIS software on the other hand to do the connection between the three technologies. Indeed PLM solutions start integrating BIM/GIS web viewer to enable a 3D visualization of the building and of the constructive elements. BIM/GIS viewers also start implementing connector with PLM solution that store the files, and manages rights and deal with workflows and processes.

1.4 PLM Scope

The proposed architecture developed in this document is still in a conceptual research stage. Research on the different parts of the architecture is still ongoing (i.e. development of the Design Configurator, Dashboard…). So it might be possible that the architecture will be brought to change to be optimized further and according to the solution developments between WP3, WP5 and WP6.

1.4.1 Final users

The PLM solution is a collaboration tool around the BIM and GIS models intended for all the actors and participants to the project: architects, engineers, designers, technical subcontractors…

We can classify the users into two categories:

- The contributors (architect, engineering consultants, general contractors…)
- The consultants (simple analysis/consultation use)

The first ones can deposit BIM and GIS models on the server, whereas the other ones can only consult the documents and eventually annotate the BIM and GIS models in order to check the good progress of the project.

1.4.2 Project type (construction or reconstruction)

This type of solution can either be used for construction or for renovation.

The only constraint about using this solution for renovation is the existence exigency of a digital model. If it does not exist yet, it has to be created first using tools of software development as the Rastertech solution that scan 2D plans to IFC format or the Viz'All one that generate the BIM model using laser meter on the existing building.

1.4.3 Proponent of the BIM-GIS-PLM solution

The people that can be the instigator of such a solution could be the following:

- The consortium representing
- The project owner that wants to organize its team work
- Regional association/federation platforms to allow the companies they represent to answer calls for tenders with their grouping
- Control office/insurers
- Any kind of organization which has the mission to federate several actors around the same project

1.4.4 Role of the BIM Manager

The emergence of this Model-based solution implies the appearance of new jobs as for example the BIM Manager function. This BIM specialist masters perfectly the processes of virtual construction and documentation. His role is to assist the various actors of the construction in the implementation, the management and the use of the BIM.

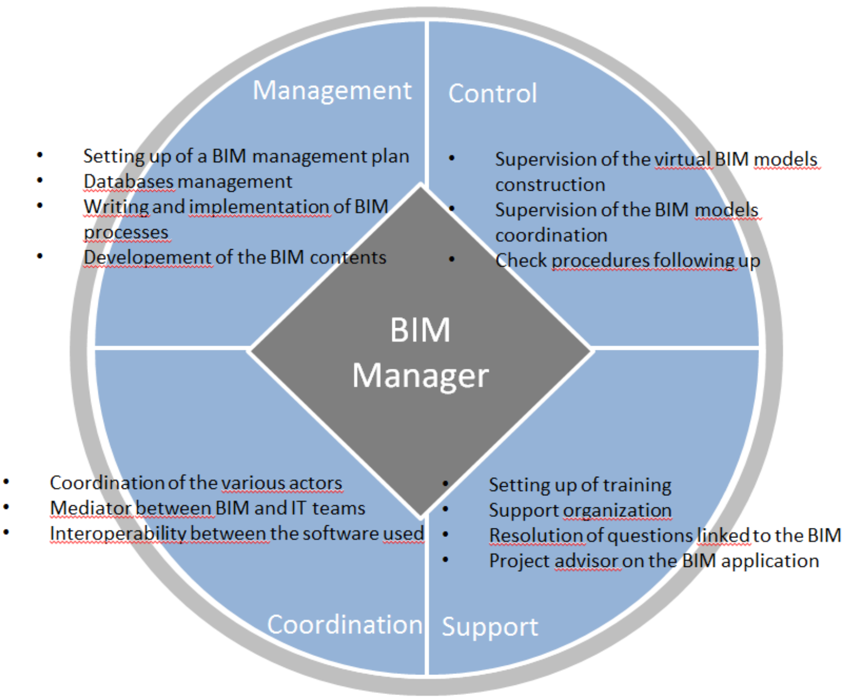

Figure 1: BIM Manager main roles

1.5 Organization of the document

The deliverable is organized as following:

- Firstly, it summarizes the objectives and challenges of the task.
- Secondly, it gives a definition of what a PLM is by providing a general overview of what it should be in charge of: storing document, managing rights, providing workflows and processes, supporting collaborative work. PLM is already well established in multiple industries like automotive, aerospace and defense (that have to manage large volume consistent data), energy, pharmacy (that have obligation to bear conformity), fashion and cosmetics (which have to follow collections and brands). All this industries are facing with different issues and challenges. The goal would be to expand PLM in building sector. In particular the points raised in this section are:
  - Panel description of PLM solutions
  - Main services deployed by PLM solution to manage complex business processes and information associated with document, product, project and complex systems?
  - Why does the Building industry feels the urge to use the PLM concepts and technology and what will be the impact according to the actual needs of customer and society?
  - What kind of problems a collaborative solution could solve?
- Thirdly, it presents the architecture of an ideal PLM architecture and the one proposed within STREAMER according to the context of healthcare districts. BIM allows thanks to IFC to represent a structured set of information on an existing building or project, GIS enables thanks to CityGML to have the same at a city scale. BIM and GIS are key bricks that allow collaboration between various actors but they need an additional solution to ensure proper management of the life cycle of the building project, a PLM tool enabling to formalize validation processes, exchanges and modification. This document will present a collaborative design tool taking into account the multiple dimensions of information models (semantic BIM, GIS).
- Finally it concludes with the further research direction track and particularly with the link to the other work packages in STREAMER project.

The prototype that will instantiate this specification (deliverable D5.4) and stemmed from these specifications should be available for all STREAMER partners to test the concepts presenting into this current document according to their own tools, processes and application field.

2. Product Lifecycle Management state-of-the-art

2.1 PLM definition

2.1.1 Goal of a PLM solution

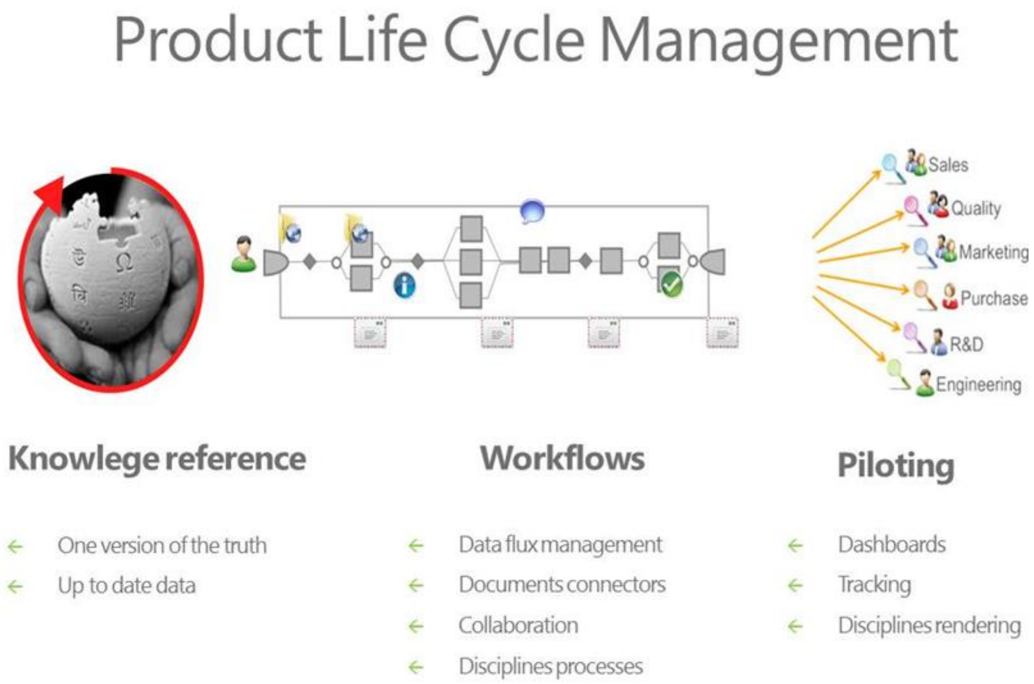

Figure 2: Product Lifecycle Management definition

PLM (Product Lifecycle Management) consists in capitalizing on all information regarding an industrial product. The PLM is an approach revolving around the product. Its goal is to allow the different actors of a project to share the knowledge of the different stages of the life cycle of a project/product. It provides a set of elements (computer tools, organization system, and working methods) of information system. It manages information from various life stages of industrial products that run the processes. And it provides all this information to all building industry stakeholders.

A PLM solution represents:

- A version of the truth / data and documents checking
- Effective collaboration / always up to date
- A way of sharing the IFC digital models

2.1.2 Positioning compared with OMG standards

About the Object Management Group

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.
OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets. More information on the OMG is available at http://www.omg.org/.

OMG already propose PLM services whose specification available on the web (http://www.omg.org/spec/PLM/) defines a Platform Independent Model for Product Lifecycle Management Services. Some companies and working groups (like ProSTEP: http://www.prostep.org/fileadmin/user_upload/ProSTEPiViP/Standard-Infos/OMG_PLM_Services/PLM-Services_1.0._Ergebnisse.pdf) have then proposed PLM Services specifications based on the OMG standard.
The objective of the PLM API the CSTB will propose is also inspired of this standards.

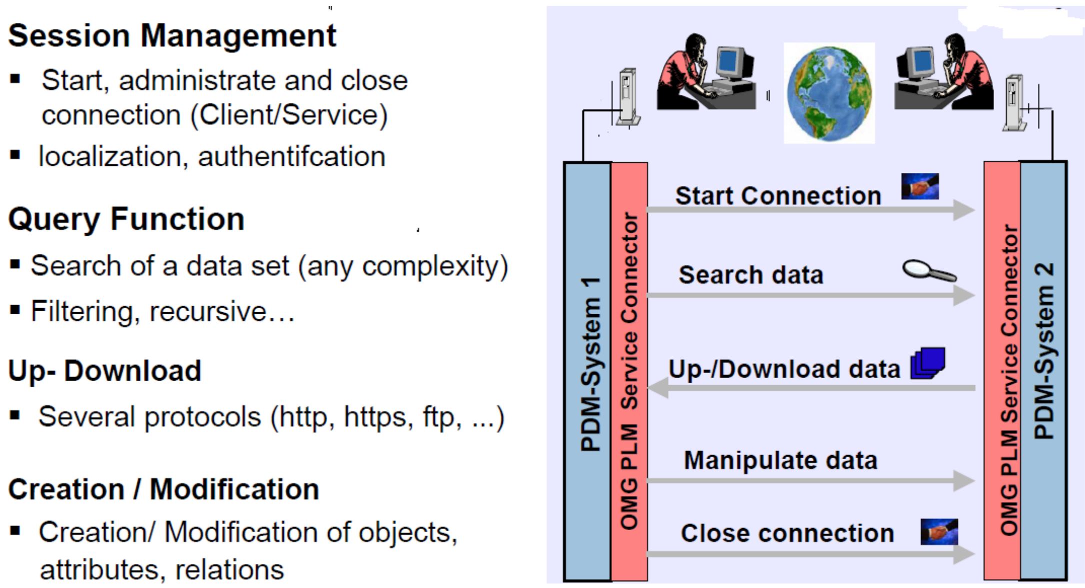

Figure 3: Use case example based on OMG PLM Services

2.2 PLM state of the art

2.2.1 Main advantages of a PLM solution

In this section the objective is to describe the main advantages of a Project Lifecycle Management that could be useful to exploit for our BIM and GIS problematic. Indeed we may be attentive to store data in a reliable way, to easily reach and visualize the data wherever and whenever we are. Moreover we are searching for an easy configurable solution that may be capable of defining rights and processes to enable collaboration processes.

Advantage	Comment
Reliable and safe	The PLM enables to store the data and to control them ➢ Centralized and secured databases on the PLM ➢ Backup and archiving functions on the PLM ➢ The server allows each actor to follow in real time the progress of the project and to adjust it if needed
Easy access	The PLM is most of the time accessible through a web browser The PLM solution enables easy access from AEC business tools ➢ Often full-web solution ➢ Personalized workspaces (dedicated according to the users) ➢ Multiple research tools
Fast	➢ All the exchanges between the different actors of the project are centralized on the server and managed by the PLM ➢ Quick and productive collaborative work
Wherever and whenever	➢ In window office (in network or on a construction site) ➢ Whatever are the complexity of the project and the number of actors ➢ Any modification is immediately taken into account and real time (to allow possibility of reacting, of rectifying, to adjust, to specify, to exchange)
Flexible	➢ In its management of partners access rights and contributions to the project
Configuration management	The PLM enables « Product » structure management
Change process management	➢ Access rights control ➢ Workflow system
Collaboration management	➢ Shared workspaces ➢ Synchronous ans asynchronous collaboration tools
Integration with business tool	➢ CAD, CAM, Office, project Management, ERP
Visualisation 2D/3D	➢ System configuration ➢ PLM construction in relation to business needs

2.2.2 Panel of some PLM applications

The objective of this section is to present a short panel of some existing PLM applications.

Company	Solutions	Software family	Functionalities
Dassault system	ENOVIA VPLM-MatrixOne-SmarTeam	CAO	➢ 3D live ➢ 3D immersive environment ➢ 3D measurements ➢ 3D cross sections for collaborative studies and presentations ➢ Live similarity ➢ Access real-time information on part substitutions ➢ Navigate in the Graphical User Interface and identify similar ➢ Retrieve instantly parts information and view appropriate available components in 3D thumbnails with names and ranking information ➢ Live validation ➢ It enable to share a complete 3D report of how development decisions are made ➢ Optimize visualization, navigation, analysis, communication ➢ Users can analyze product consistency, display review presentations, execute design validation, conduct 3D meetings, replys to reviews ➢ Process collaboration ➢ Collaboration for Microsoft ➢ Collaboration 2D and 3D viewing with Autovue ➢ Live Collaboration ➢ Live connect ➢ Report generator
PTC	Windchill	CAO	➢ Enable collaborative work multisite ➢ Multi-CAD environment (manage all digital data regardless of their nature; documents, images, videos, CAD…) ➢ Structured and secure information ➢ Tree structure of files depending on their nature (reports, CAD documents, simulations, videos, photos…)
SAP	SAP-PLM	ERP	➢ Development project control: gateway, assignment, resource, deadline and project portfolio management ➢ CAD model and drawing administration for the most popular CAD systems, such as SolidWorks, SolidEdge, Inventor, Pro/Engineer and CATIA ➢ Document administration and project archive ➢ Parts master data and bills of material ➢ Classification

Company	Solutions	Software family	Functionalities
			➢ Integrated change management (development statuses, series switches) ➢ Variant configuration ➢ Development-accompanying costing ➢ Data sharing platform cFolders
Cegid	Isoflex	ERP	➢ Traceability of your documents in accordance with quality standards ➢ Find information quickly with secure search engine ➢ Keep the history of changes in documents ➢ Secure your documents with the Document safe ➢ Graphically design your workflows directly in Visio ➢ Workflows can drive directly into Outlook ➢ Archive and simply classify in the ERP incoming and outgoing mails. ➢ Simply import documents with "drag / drop"
Assetium	Audros	Not connected to any CAO	➢ Detailed technical EDM ➢ Collaborative work ➢ Projects managements ➢ Automation of process ➢ Project management
Siemens	Teamcenter	Large industrial group	➢ Management of product data ➢ Ubiquitous 3D visualization of multi-CAD data to review, analyze and collaborate on complete product assemblies ➢ Digital validation integrating a digital model, a product configuration and aggregation designs. It aggregates design changes continuously into digital models "always displayed." ➢ Modification management allows a workflow orchestration for advanced engineering processes to view impact, initiate, administer, review / approve and execute product changes. ➢ Room Classification ➢ Connectivity with the enterprise resource planning
Oracle	Agile	Large industrial group	➢ Document Management ➢ Product Data Management ➢ Configuration Management ➢ Product Catalog ➢ Custom Engineering ➢ Extensive Change Management ➢ Product Catalog ➢ Real Multi-CAD Support and comprehensive integrations ➢ Support of a distributed working model with workflows and optional workspaces

Company	Solutions	Software family	Functionalities
			➢ High configuration ability and flexibility
Lascom	Advitium	Linked with CAO	➢ Collaborative platform ➢ Plans and documents management ➢ IFC integrated viewer ➢ Link between the documents and IFC models ➢ Validation process ➢ Forward lists ➢ Efficient searches ➢ Rights Management ➢ Sharing digital models ➢ Management of records and deliverables ➢ Annotations and comments ➢ Automatic information sharing ➢ Monitoring delays ➢ Connections management ➢ Project directory
Gespro	Centrinfo, Bimpro, DifDoc, profi, Oreli	Linked with CAO	➢ Ifc model synchronization ➢ 3D visualization ➢ Analysis and model sharing and enrichment ➢ Historical and tracing events ➢ Integrated videoconferencing for simplified work meetings
Visiobat	Visiobat	CAO	Key features: ➢ Real-time management ➢ Written documents, maps, messages… ➢ Collaboration architects and BET ➢ Push messaging ➢ Quality procedures ➢ Implementations plans ➢ Summary documents ➢ Visas and vouchers for execution ➢ Version management and conversion ➢ … ➢ Security ➢ Compatible with all formats: ➢ Office, CAD, PDF, scanned documents… ➢ Automatically convert DWG to DWF ➢ Recognition ZIP formats ➢ PDF printers ➢ Versioning and converting files in AutoCAD format ➢ Full-integrated Web Viewer ➢ Version comparison

Company	Solutions	Software family	Functionalities
			➢ Distances, areas and perimeters computations ➢ Management and paper space AutoCAD object ➢ Powerful search functions ➢ Documents functions ➢ Access permissions ➢ Shares history ➢ Configuration visa circuits

The choice of a PLM rather than another one will depend on the project constraints and objectives. No solution is in general better than another one.

- Most of them are web solution.
- Some of them are already integrated with BIM viewers (that can be web solution or not) and propose 2D and 3D additional computation tools
- Their application sector can be different according to the family they come from (CAO, ERP…) and to their experience.
- All the workflow proposed can be adapted to BIM needs as far as the processes proposed are modular and adjustable.
- Some of them propose additional tools like search functions, dashboards, messaging…

The PLM solution that will be used in STREAMER is still not defined. It will depend on the functionalities which are going to emerge from the global use cases of the project. It will also depend on the development required and in progress.

2.2.3 Details of generic PLM solution functionalities

The goal of this section is to present the advantages that a PLM could provide in the case we decide to couple BIM/GIS with any kind of PLM solution. Indeed independently of the PLM solution chosen, all the functionalities presented in this section which are common PLM functionalities will be to the benefit of the STREAMER project.

Function	Comment
Structure product information	➢ A common reference for all stakeholders -> central access point to pool information: ○ Standardize information management (cross-sites) ○ Have an overview of the product ○ Identify and justify the product configuration states ○ Facilitate data management (life-cycle, maturity) ○ Control evolutions ○ Reproduce the product and reuse information **PLM contribution** ➢ A common reference for all stakeholders : central access point ○ Nomenclature management

	○ Tree structure product tree according to various viewpoints trades ○ Item management (classification, interchangeability, effectiveness) ○ Searching functions (associated documents, use case) ○ Nomenclature comparison ➢ Link with business applications (CAD, CAM, CAE…) viewers or modifiers -> CAD file management ➢ Link with standard component libraries
Discipline data management	➢ Find up to date information (not to manufacture with the wrong data) ➢ Do not "create" existing components ➢ More "recopies" ➢ Keep track of the operations performed on data ➢ Secure information ➢ Provide quality information ➢ Facilitate collaborative work **PLM contribution** ➢ Secure information from its creator and based on the user profile ○ Electronic safe ○ Access rights ○ Locking a given ➢ Instruct the data validation process ➢ Centralize changes (any change in an application through PLM)
Manage processes around product	➢ Automate reporting and disseminaion ○ Limit unnessary foreclosures ○ Limit the physical transfer of paper ➢ Keep track of transactions and decisions ➢ Orchestrate processes (automation of tasks, transmission time) ➢ Involving the data to process activities ➢ Automatically generate documents ➢ Have a better management of product development project ➢ Have a better change management ➢ Increase standardization of processes ➢ Define and apply the approval circuits, change, evolution ... ➢ Increasing the rigor in the processes ➢ Facilitate project transfers between location **PLM contribution** ➢ Workflow engine

  - Definition of the process: steps, actors, associated data sequences
  - Orchestration: notification, passage of a activity to another, timing, action on the life cycle of data, generating reports
- User management (groups, profiles, roles, substitutions)

2.3 Use of PLM in STREAMER

2.3.1 Benefits of PLM for the STREAMER project

The use of Project Lifecycle Management in STREAMER will enable to simulate the course of a collaborative project and to illustrate the advantages of such a solution. This will imply resolving “historical” malfunctions by centralizing document, enabling data security, optimizing product development, automated processes implementation, performance encouraging and also productivity at any stage increasing.

The PLM will be the glue between all the different tools and data. All the documentation, models, computation results will be stored on the server, will be accessible at any time by any actor and will be visible through any solution that will implement the PLM API.

2.3.2 Life-cycle validation through PLM

2.3.2.1 General definition

One of the main advantage of combining PLM and BIM is that PLM solution supplies validation processes. Each kind of document can be validated according to a completely customizable circuit. Only PLM administrators can have access to such processes definition.

Each validation process involves defining the following criteria:

- Document type (plans, technical document…)
- Process type
- Project concerned
- Document transmitters
- Document validators

Here is an example of process flow:

- Defining the validation model:

According to the model selected and the kind of document to validate, the process will be different.

- Sending the document to approval:

Once a document is finished and has to be validated, only users allowed (good rights on the PLM) can send it to approval: the applicant can ask for a review for the document and the validation process starts.

  - Users selection

During this phase the approver actors list is selected. A distinction is made between the approvers whose opinion is only an advice (no impact in the validation process) and those that can stop the process if giving a negative appreciation.

  - Review/visas step

Each approver has access to the document(s) to validate. He can add a comment, request for further review, affix his signature or not, refuse the document or eventually join more documents to add information.

- Last validation step

At the end the state of documents to validate will change according to the predefined configuration process. The possible states are: approved, delayed of approval or refused and sent back in approval.

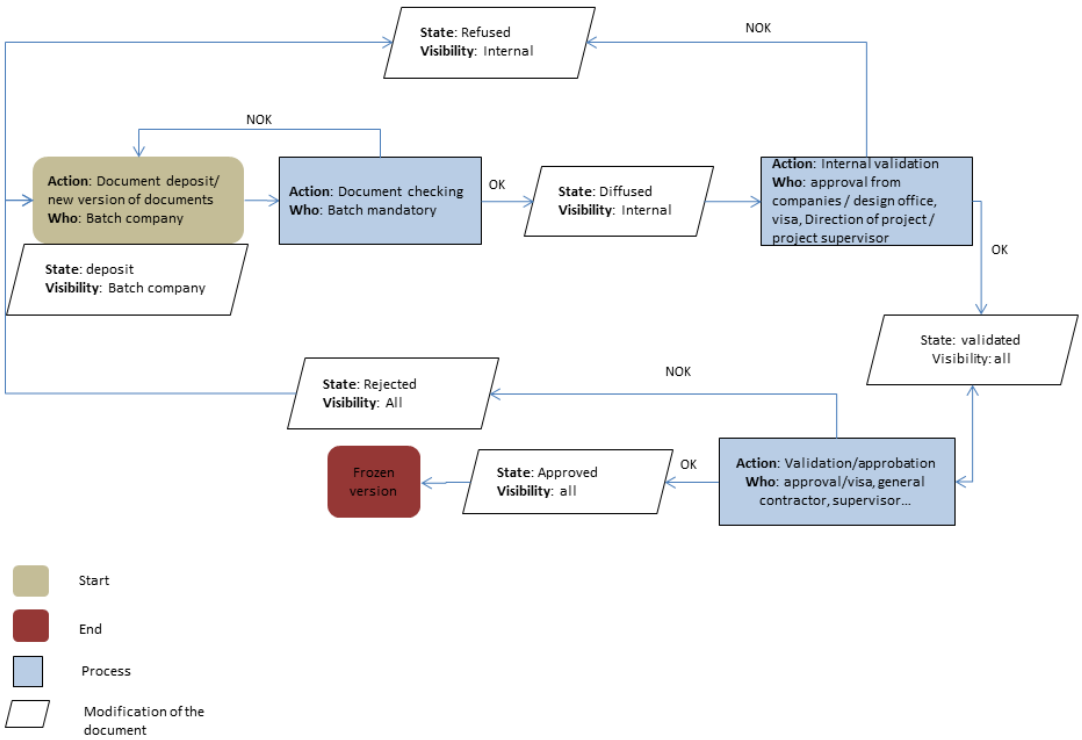

Figure 3: Example of document lifecycle in building industry

The previous diagram is an example of workflow that can be instantiated within a PLM.
At each stage of the process a document that is deposit on the server has to be validated by a person or a group of persons and the validation process implies a modification of its state and visibility.
The example starts with the deposit of the document by a company. The document has to be checked by the batch mandatory. If this step succeeds the state of the document gets from “Deposit” to “Diffused” otherwise a new version of the document must be uploaded and the process starts again from beginning. Then the document has to be validated internally by various actors (companies, design officer, direction of project…) to get from state “Diffused” to state “Validated”. If an actor refuses the document the process has to start again from beginning. Otherwise it goes through approbation step to get the final “Approved” state.

In Figure 4: Example of STREAMER workflow, a document is deposit and must successively be validated by different actors to be first checked, distributed, approved, validated and finally broadcast. At each validation stage of the workflow, the validation process has to start again if someone has rejected the document.

2.3.2.2.2 STREAMER workflow example

The following diagram is a suggestion of workflow that could be set up within the STREAMER project.

The process starts by the deposit of a first document: the Program of requirements represented in the green box. Once validated, design propositions are suggested by the Early Design Configurator. If rejected a proposition can return to the program of requirements or only be aborted. Once validated, the reserved solution is then rounded out by the GIS data of the neighborhood through BIM-GIS Integration tool. Once checked the BIM-GIS data can be used by energy simulation tools that propose a panel of results according to design choices. On the other hand the design support tool also computes KPIs and store them in analysis server. Simulation results and KPI value will then allow choosing the design that will propose the most efficient solution in terms of energy consumption.

This workflow is a proposition that has to be confronted to deliverable D4.1.

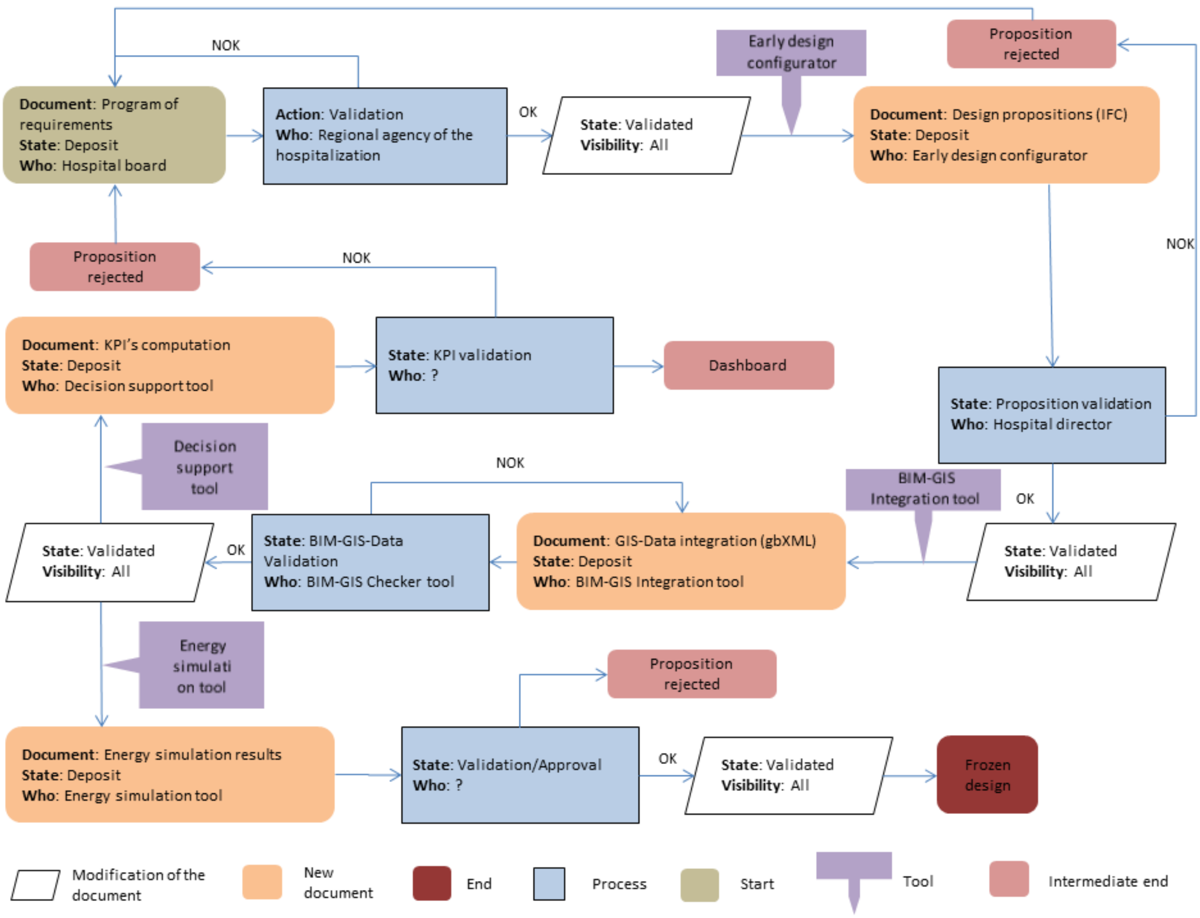

Figure 4: Example of STREAMER workflow

2.3.3 PLM and semantic web

The semantic web is a set of techniques allowing connected computers to give meaning to the accessible data on the Web in order to share or aggregate them more efficiently and to facilitate data exchange between computers.

By easily sharing the data of conception and production distributed in the various systems of information, the semantic web enables the convergence of the various software packages and solves the interoperability problems between data bases.

A Semantic web for a PLM suitable for each user

Semantic web is an old initiative implemented to better use available resources on the Web. This is a Web extension which operates its initial architecture. It focuses on the notion of resources and on their matchmaking.

Initially the Web was documentary by nature, focusing on the presentation of documents through a browser. Markup standards allowed a much more refined and generic approach of the data. The modeling of internal structures of a document independently of a particular usage led to a related data Web.

In practice the semantic web is a set of standards used to describe and link resources according to public models which are objects resulting from consensus by field.

These initiatives open new opportunities to extract data from applications where they have been published and to enhance them through the interoperability they offer like for instance:

- web-services: they refers to "web sites that do not merely provide static information but allow one to effect some action or change in the world". The Semantic Web will enable users to locate, select, employ, compose, and monitor web-services automatically.
- agent-based distributed computing: The Semantic Web will use ontologies to describe various web resources, so, knowledge on the Web will be represented in a structured, logical, and semantic way. This will change the way that agents navigate, harvest and utilize information.

3. Model-based PLM

3.1 Global architecture of the PLM solution

3.1.1 Generic PLM architecture

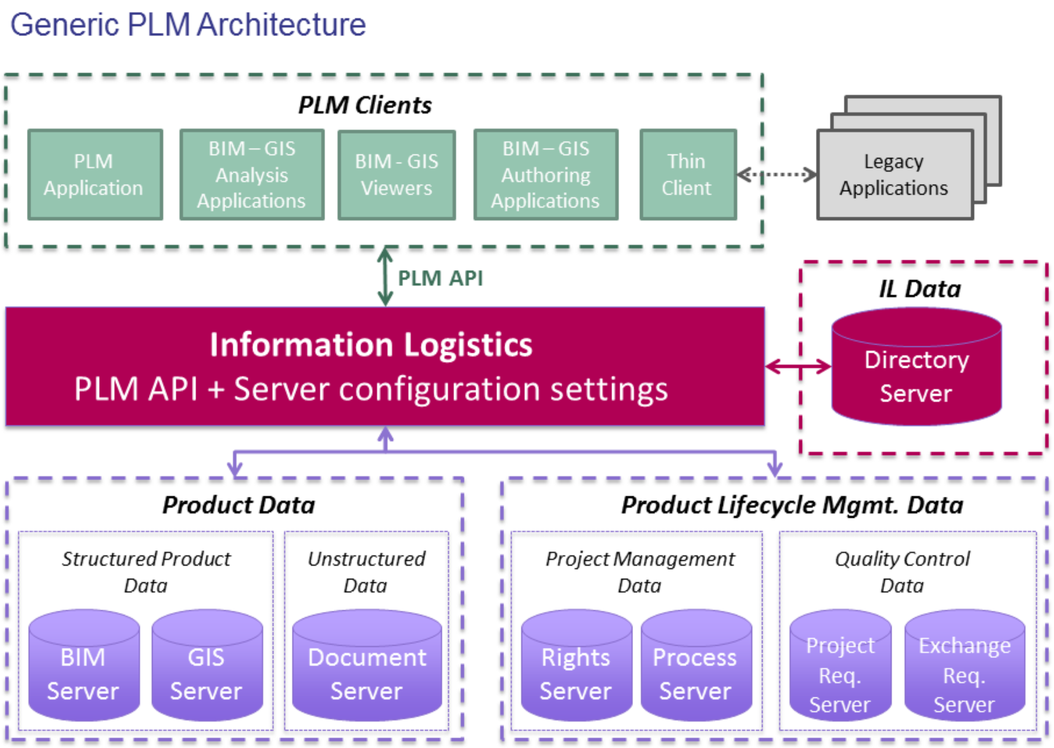

Figure 5: Generic architecture to combine BIM and PLM

This architectural diagram is and exhaustive representation of the ideal coupling between BIM and PLM based on Details of generic PLM solution functionalities. It is divided into three main parts:

PLM Clients

All kind of possible clients are represented here:

- PLM application: it could be any PLM software (like all those described in section 2.2.1 Main advantages of a PLM solution

In this section the objective is to describe the main advantages of a Project Lifecycle Management that could be useful to exploit for our BIM and GIS problematic. Indeed we may be attentive to store data in a reliable way, to easily reach and visualize the data wherever and whenever we are. Moreover we are searching for an easy configurable solution that may be capable of defining rights and processes to enable collaboration processes.

Advantage	Comment
Reliable and safe	The PLM enables to store the data and to control them ➢ Centralized and secured databases on the PLM ➢ Backup and archiving functions on the PLM ➢ The server allows each actor to follow in real time the progress of the project and to adjust it if needed
Easy access	The PLM is most of the time accessible through a web browser The PLM solution enables easy access from AEC business tools ➢ Often full-web solution ➢ Personalized workspaces (dedicated according to the users) ➢ Multiple research tools
Fast	➢ All the exchanges between the different actors of the project are centralized on the server and managed by the PLM ➢ Quick and productive collaborative work
Wherever and whenever	➢ In window office (in network or on a construction site) ➢ Whatever are the complexity of the project and the number of actors ➢ Any modification is immediately taken into account and real time (to allow possibility of reacting, of rectifying, to adjust, to specify, to exchange)
Flexible	➢ In its management of partners access rights and contributions to the project
Configuration management	The PLM enables « Product » structure management
Change process management	➢ Access rights control ➢ Workflow system
Collaboration management	➢ Shared workspaces ➢ Synchronous ans asynchronous collaboration tools
Integration with business tool	➢ CAD, CAM, Office, project Management, ERP
Visualisation 2D/3D	➢ System configuration ➢ PLM construction in relation to business needs

- Panel of some PLM applications) that would use the PLM API to access data.
- BIM-GIS Analysis Applications: BIM Analysis tools are BIM software to analyse and predict model behaviour. They are used to validate model compliance with standards/codes. They are used by the designers and consultants and are commonly used in the preliminary design phase until construction phase.
- BIM-GIS viewers: BIM involves representing a design as combinations of "objects" that carry their geometry, relations and attributes. BIM design tools allow extraction of different views from a building model for drawing production and other uses. These different views are automatically consistent, being based on a single definition of each object instance.BIM software also defines

objects parametrically; that is, the objects are defined as parameters and relations to other objects, so that if a related object is amended, dependent ones will automatically also change.

BIM-GIS Authoring Applications: BIM programs such as Autodesk's Revit, Tekla Structure or Graphisoft's ArchiCad that are used to create the initial BIM model. BIM authoring tools allow creating construction drawings and can be viewed in 3D.

Thin Client: a client machine that relies on the server to perform the data processing. Either a dedicated thin client terminal or a regular PC with thin client software is used to send keyboard and mouse input to the server and receive screen output in return. The thin client does not process any data; it processes only the user interface

Data

This bottom section of the schema represents the data which are of two types:

- The product data

Under this appellation, we have two kinds of data:

  - The structure product data (see Structured product data) made available through BIM and GIS servers. The goal of these servers is to enable collaboration between AEC actors and BIM processes for building but also across the city. These servers are intelligent central storage for BIM model, GIS data and all documentation needed by AEC professionals to help them collaborate around building processes. But they are also in charge of storing more than BIM and GIS models. They also provide BIM and GIS functions for dealing with these models. For example concerning BIM data, IFC models could be converted and stored semantically as topology, disciplines and layers. The BIM server can provide versioning methods to identify the objects that have been altered, added or removed. I can also enable rendering, clash detection, measurements, simulations or also planning and costs.
  - The unstructured data (see Unstructured data)
    This data concerns all the building information needed for collaboration between all the actors of the project: 2D and 3D plans, excel sheets, calculations, annotations and comments, visa advices… Links can be done between structure and unstructured data as far as IFC files or parts of IFC files from BIM server could be linked to documents of the unstructured data.

- The product lifecycle management data

  Under this appellation we also have two kinds of data:

  - Project management data to know:
    Access rights to knowledge and data manipulation. The access of an actor to this server goes through a login/password connection that enables a particular view of the documents linked to the function and role of the actor in the project. A BIM Manager, an architect, a design officer or a control officer will have different views of the documents and different actions on them (only consultation, contribution, annotation, creation…).
  - Process definition. In this server will be available the visa advices processes: for each phase of the project, which actor is in charge of what, what are the validation protocols. What is the communication between contracting and execution?

- Quality Control data: it concerns the data that concerns the validation of the structured product data.

Information Logistics

The intermediate section between the upper PLM clients and the bottom data is the central brick called Information Logistics. The principle is the following: a PLM client sends a request to this central BUS. Whatever the language or the type of demand, the Information Logistics can answer by soliciting the good data server in the right language. The localization of the data is known by the directory server and the translation between BIM and GIS data is made possible thanks to the semantic contained in this configuration server.

3.1.2 STREAMER PLM architecture

Due to the complexity of such an exhaustive architecture, the competences of the different partners and the multiple owner languages, choices have been made and solutions have been identified.

The final architecture proposed for STREAMER is the following:

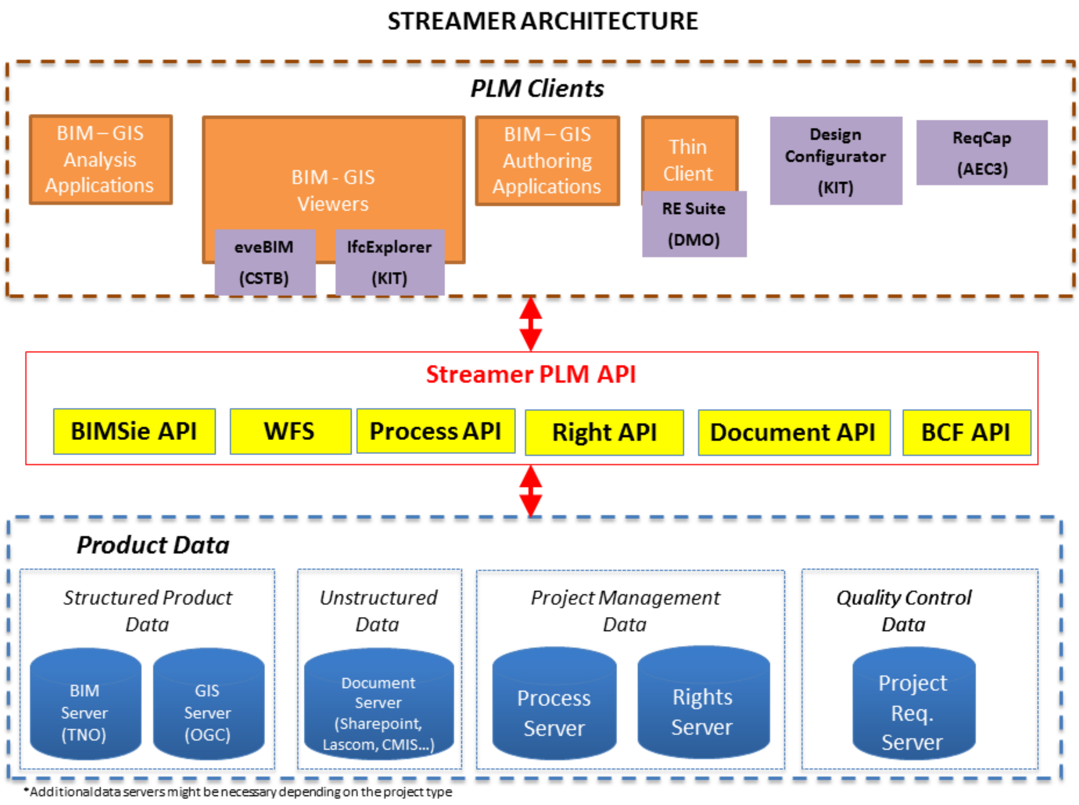

Figure 6: Restricted STREAMER architecture

The principal difference with the global architecture is that the API developed for the STREAMER project will not re-implement all languages to access servers but will use existing API like the BCF, BIMSIE or WFS.

PLM Clients

The clients identified for the STREAMER project are those represented in purple boxes:

- The two BIM/GIS viewers identified are: **eveBIM-Viewer** (CSTB) and **IfcExplorer** (KIT).
- The thin client that will be used for the STREAMER project is the **Re Suite** software from DMO which is in charge of collecting, structuring, analysing and disclosing real state information through a dashboard platform.
- The **Design configurator** from KIT is also identified as a PLM client. This software is in charge of proposing multiple configuration solutions at the design stage of a building construction or rehabilitation according to predefined requirements in order to prepare energy simulations.
- The **ReqCap** tool from AEC concerns the quality control of the data. This software might connect to the Project Requirement Server through the STREAMER PLM API to store is results like mvdXML generated files.

Information Logistics

The central generic information logistics part of Figure 5: Generic architecture to combine BIM and PLM has been decomposed into a set of more detailed APIs:

- BIMSie API (BIM Service Interface Exchange): a standard API for BIM Web Services to get BIM into the cloud
- WFS (Open Geospatial Consortium Web feature Service Interface): interface allowing requests for geographical features across the Web.
- Process API: This API manage the define and follow BIM workflows
- Right API: This API enables to manage and administer users on the different servers.
- Document API: This API describes the CRUD (Create Read Update Delete) API to manage documents
- BCF API (BIM Collaboration Format): It defines the open file XML format to support workflow communication in BIM processes.

The details of this STREAMER PLM API are clarified in appendix of the document.

Data

- The BIM Structure Product data will be accessible via BIMSie API in regards to BIM server.
- On the GIS side, WFS protocol will enable to query mapping servers to manipulate geographic objects.
- The unstructured data will be accessible via any Document server. CMIS, dropbox or also SharePoint have been discussed as possible solutions.
- The PLM data will be stored into Process and rights servers.
- All the data related to quality will be managed by the Project Requirement server.

The main efforts of the project will focus on the following developments:

- From BIM or GIS viewers, accessing the document server where all kind of document will be store (IFC files, GIS files or computation results).
- From thin client (principally RE Suite software) reaching the document server to get computation results to represent them through multiple tables or dashboards.

To achieve these tasks, rights API, processes API and document API will be the principle parts of the API that will be implemented.

3.2 PLM clients

3.2.1 PLM application

In this section is provided an example of PLM application: the eveBIM-Edition CSTB viewer that is connected to an existing PLM solution Lascom. The functionalities of the solution are presented according to the Figure 4: Example of STREAMER workflow on which we made a zoom below.

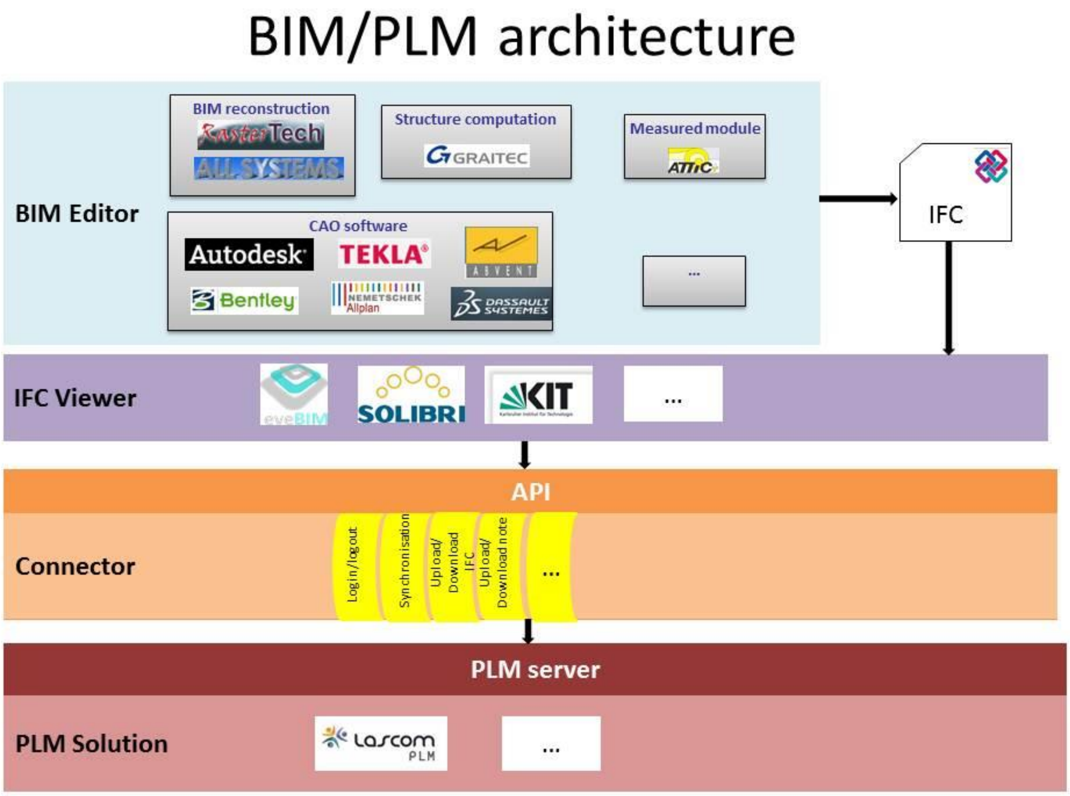

Figure 7: BIM/PLM architecture using PLM API

The previous schema is a zoom of the Figure 6: Restricted STREAMER architecture applied in the case of a PLM client which is a BIM-GIS viewer. We can see the link between BIM Editors and BIM viewer upstream of the schema and the connector is another term for PLM API.

3.2.1.1 PLM application architecture

First step: IFC generation

Whether the 3D model exists or not, the architect draft export into IFC format can be done using a BIM Editor.

Second step: IFC analysis

Once the IFC file has been generated, it can be visualized into the IFC Viewer of your choice.

Third step: Distribute the IFC file on the network

To exchange data with the other actors through a PLM solution, you have to connect your BIM Viewer to a PLM solution including an API. Some functions of this API are detailed here: login/logout, document download/upload, note upload/download methods…

The PLM server enables to store all kind of data and to manage all types of file extensions (native files, .ifc, .dwg, excel sheet, word…). At any time you can have access to the documents you need: either the last version or a previous one.

To be more efficient, only mature documents are stored on the PLM: for example during the construction of the native architect draft, the actors generally prefer to exchange the native files through a software versioning as SVN. Once the sketch has been finalised, the native file and the exported IFC files can be stored on the PLM and published to all the other actors.

You will find as much IFC files as needed on the PLM server: the original architect one, those extracted for particular analysis and computation (for example for structure, energy or also costs computation tools)…

What the PLM does not provide:

PLM and BIM server goals are often merged. This paragraph gives the main differences between both technologies to clear up any possible misunderstanding.

The main difference is that PLM is not necessarily dedicated to the BIM whereas a BIM server is able to interpret IFC data and knows how to handle this data.

Indeed a BIM Server enables to centralize the information of a construction project (like PLM software) but as the core is based on the open standard IFC, it also manage to query, merge and filter the BIM-model what a PLM solution cannot do.

3.2.1.2 eveBIM-Edition example

3.2.1.2.1 Lascom AEC BIM Edition process

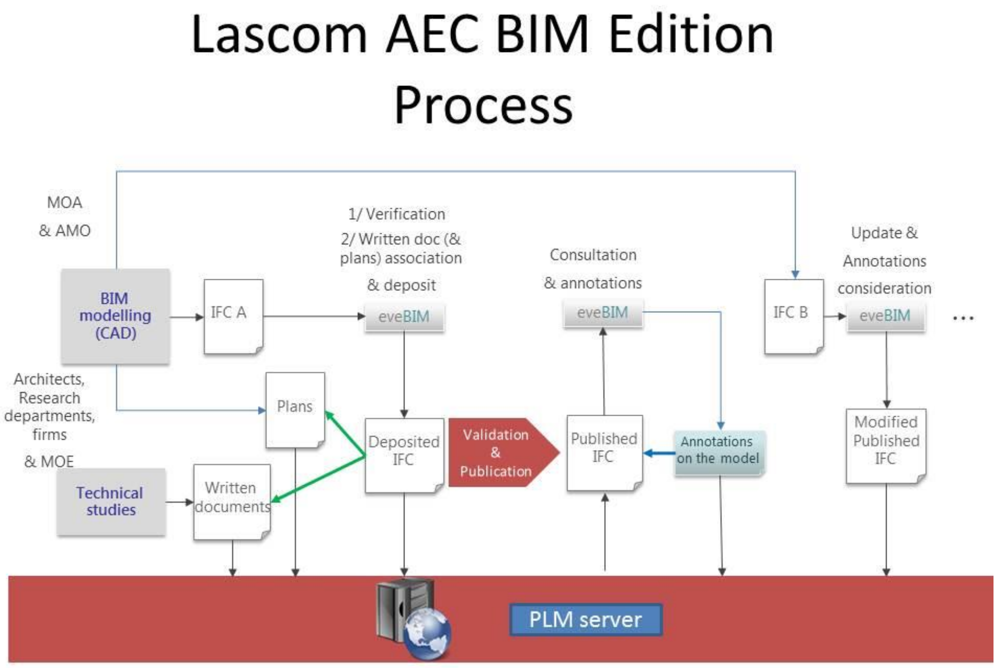

Figure 8: Lascom AEC BIM Edition process

The main process of the solution is the following:

- Each actor (architect, engineering consultants, general contractors…) uses his own BIM Modelling tool to validate the architect first draft and his own analysis tool to make calculations and simulations.
- An actor cannot modify the IFC model of another actor. However he can annotate it to highlight problems, errors, incompatibilities or simply to add a comment.
- All the files related to the BIM project are organized and stored on the PLM server so that no data will transit directly between the actors.

3.2.1.2.2 eveBIM-Edition main functionalities

The functionalities of the eveBIM-Viewer are the following:

- Visualize a set of IFC files by superposing them in the 3D view
- Have access to the characteristics of the various IFC entities (root information, property sets, local placement, representation…)
- Manipulate the models in the 3D view: zoom, walk inside, display selection…
- Do measure computations in 3D view

The connector to the Lascom PLM solution developed by the CSTB is a layer above the eveBIM-Viewer that enables to:

- Connect directly to the PLM server thanks to a dialog box connection:

A login and password will be attributed to the actor according to its positioning and its rights in the project. Once connected, he will only have access to the data he can visualize and/or modify.

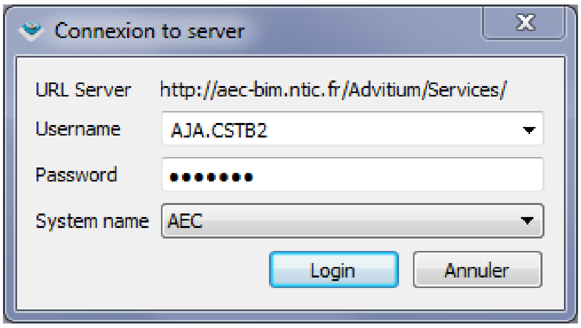

Figure 9: Connection to the PLM through eveBIM-Edition

- Upload and publish an IFC file on the Lascom PLM Server:

Once you have validated your IFC file thanks to eveBIM viewer you can upload it on the server. First you have to fill a form with all the data needed for the IFC codification (as for example the construction lot or stage of the project).

Then a new IFC object is created on the server linked with this IFC file. The state of this object is for the moment “Deposited”.

According to your rights in the project (BIM Manager or example) you will have the capacity to pusblish the file through the eveBIM-Edition software. At this moment the file will be made visible to all users.

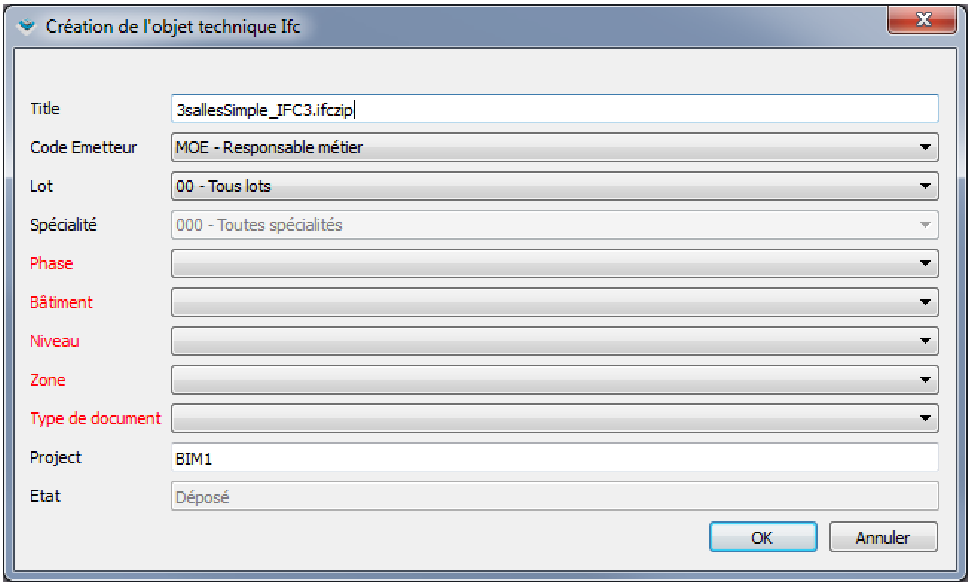

Figure 10: Upload an IFC file on the server

- Link this IFC file to a set of other PLM files (like for instance simulation excels sheets or administrative document):

You may want to attach referenced documents of the PLM to IFC files (but also to a specific entity of an IFC file) already present on the PLM. It is possible to do it using eveBIM-Edition and to store the link on the PLM. From eveBIM-Edition you will be able to open the related documents through the Lascom web interface.

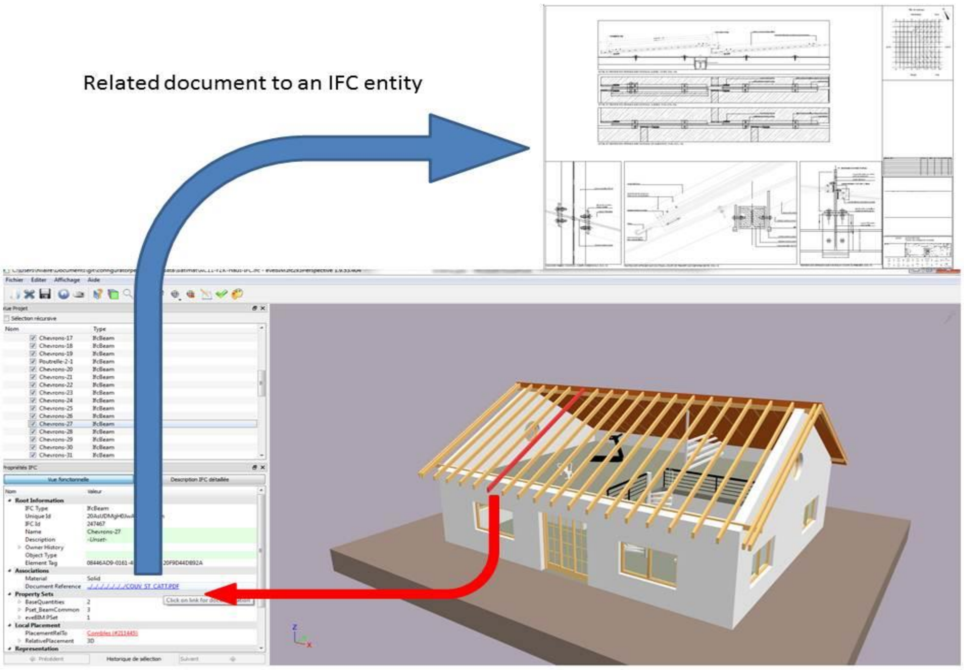

Figure 11: Associate documents to IFC file

➢ Download IFC files from the server

eveBIM-Edition proposes a view of all the IFC files from PLM server. You can open them directly from the viewer once connected to the PLM without having to go to the Lascom Web Interface. Once the file has been opened, it is locally present on your machine to enable you to work in log-of mode.

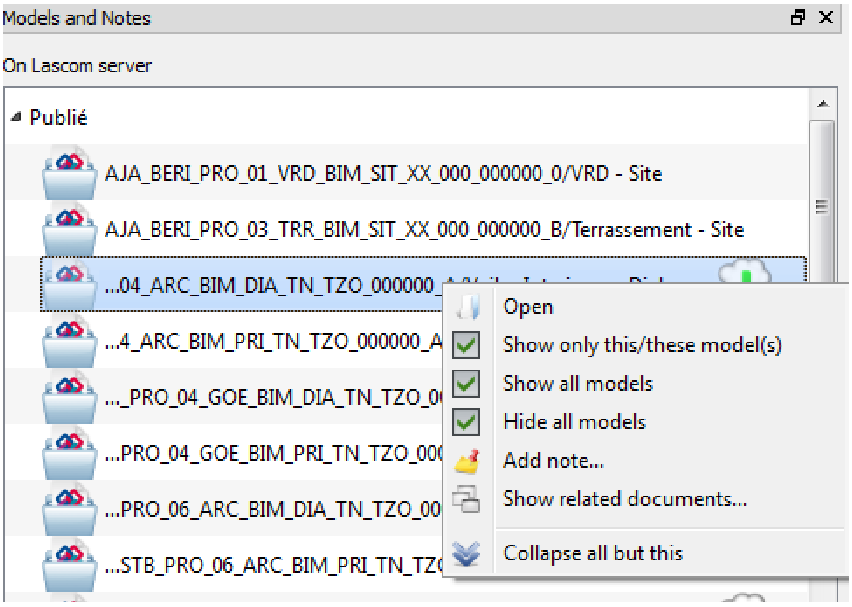

Figure 12: Open IFC file

➢ Annotate the IFC files or comment them on the PLM server

Through eveBIM-Edition you can annotate "Published" IFC files or reply to a note already deposited on the server. All the actors that have subscribed to annotations will immediately be informed that a new note or a new comment is available on the server. If the actor is concerned by the annotation, he will quickly react to adjust the model.

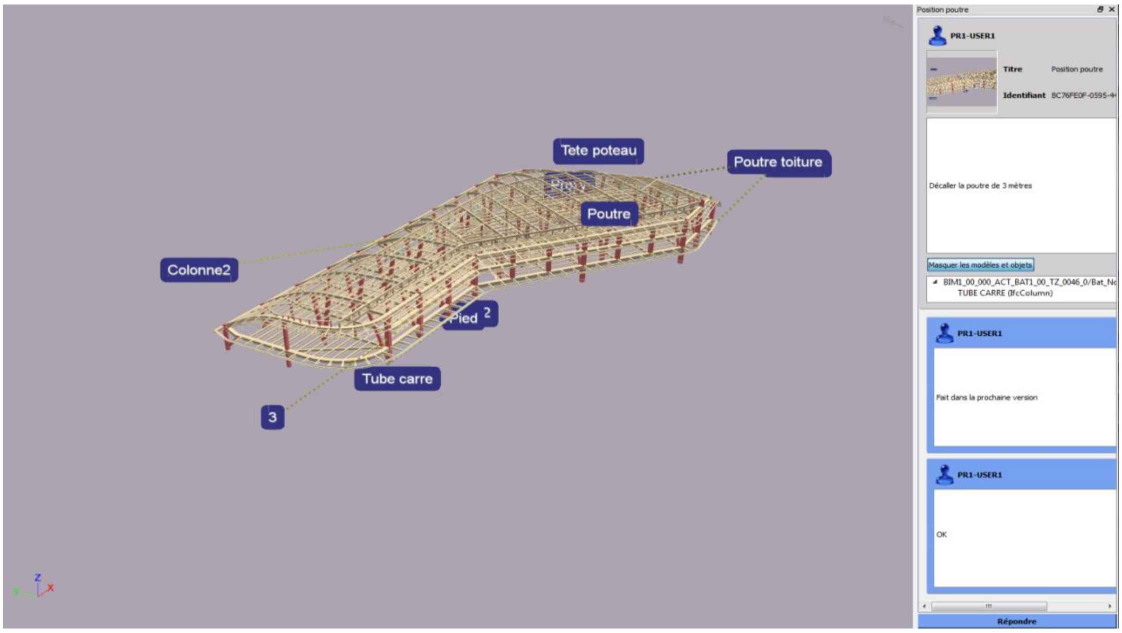

Figure 13: Note process

3.2.2 BIM-GIS analysis applications

During the design development of the building, details concerning the building's various systems must be determined to validate earlier estimate and to specify the systems for bidding, fabrication and installation. This means defining a wide range of technical information.

Buildings must satisfy many criteria of various types like structural, environmental, conditioning, electrical, energetic order…The systems required to support this criteria can be identified earlier, but their conformance to certifications and client objectives require tests and validation. Tools are more and more coming into use to do simulations and analysis.

Significant efforts are required to prepare the dataset needed to run analysis. Thanks to PLM, a more collaborative workflow will be possible, allowing different experts from different area to work together to propose the best final design according to requirements.

3.2.3 BIM-GIS viewers

One central component of PLM architecture is a viewer for the supported data formats. A viewing application needs to support the visual checking and evaluation of information, but usually not its generation, editing or processing.

In the STREAMER context, most information handled by the PLM system is generated with BIM or GIS authoring tools (see chapter 3.2.4). This especially means the data contain 2D or 3D geometry and areas related to specific location and orientation in the real world.

The STREAMER report D5.1 “State-of-the-art review of advancements and challenges in ontology research” contains a detailed overview on existing BIM and GIS viewing applications. Generally speaking, BIM - as well as GIS-viewers - needs to support at least the following functionality:

- A three-dimensional, perspective presentation of all spatially related data (the “scene”) with arbitrarily chosen viewpoints, viewing angles and viewing directions.
- A number of user-friendly strategies to interactively navigate in the scene.
- The possibility to display the scene objects with different styles. This concerns the basic visualization style (i.e. Shading or Outline), as well as the colours, line styles, and textures.
- In addition to the geometric representation of the scene objects, a structural representation of the scene (“scene graph”) should be supported, indicating functional or semantic relations between scene objects and enabling direct object access.
- A simple access to non-geometric data (e.g. attributive information, documents or time series of measure values) being related with geometric objects.

Most of the viewing applications mentioned in D 5.1 either support BIM-data (e.g. in the data formats IFC or gbXML) or GIS-data (e.g. in CityGML). However, in a PLM environment both data types occur simultaneously, and in some cases together with unstructured data like textual documents or raster images. A BIM-GIS viewer needs to support the user in merging the data and displaying then in a common scene. For this, is has to be considered that spatially related data from different sources may use different coordinate reference systems. For successfully merging these data, the PLM-system or the viewer must automatically perform the required coordinate transformations. Furthermore, a purely file-based data access, as it is supported by all existing viewers, is not sufficient in a PLM environment. Usually, all information is stored in Product Data Servers, which need to be accessed via the PLM protocol.

In the STREAMER project, many BIM viewers could be used. According to the fact we want to prototype a connection between BIM viewers and PLM solutions, we will need to reach the code to implement API methods. Consequently only viewers that are developed by STREAMER partners (CSTB and KIT) can be used. One of them is the eveBIM system, which has already been presented in chapter 3.2.1. A second viewing application satisfying the requirements mentioned above it the IFCExplorer developed by KIT (Benner et al. 2013). This software supports all STREAMER-relevant GIS and BIM data formats, and additionally is able to integrate georeferenced raster data, e.g. originating from OpenStreetMap (OSM) or GoogleMap servers, into the scene (see Figure 14: IFCExplorer scene, merging IFC, CityGML and OSM data). It is possible to merge and display all data in a common scene. Eventually required coordinate transformations are performed automatically, provided that the information on the used reference systems is available. The IFCExplorer supports the basic functionality of a BIM-GIS viewer mentioned above, and additionally has a number of analysis, checking and validation functions.

To a large extent, the IFCExplorer already supports the PLM communication interfaces foreseen for STREAMER (see chapter 3.1.2). For accessing BIM-servers, the software supports the buildingSmart Standard “BIM Server Information Exchange” (BIMsie), and for accessing GIS-servers the standards

"Web Feature Service" (WFS) and "Web Map Service" (WMS) of the Open Geospatial Consortium (OGC). Here it is to mention that the WFS-interface automatically adapts to the existing functionality of the GIS-server (see Figure 15: IFCExplorer user interface for configuring WFS requests). In a WFS-request, the user not only specifies type and range of the needed data with attributive and spatial filters, he can also choose among the available data formats and coordinate reference systems. Additional technical information on the interface standards can be found in the STREAMER report D6.5 "Advance Mapping Structures and Standards", chapter 2.4.

The IFCExplorer furthermore supports the OASIS standard "Content Management Interoperability Services" (CMIS) for the access to document management systems like Sharepoint. As soon as the specification of the STREAMER PLM API is available, it is planned to prototypically implement also this interface.

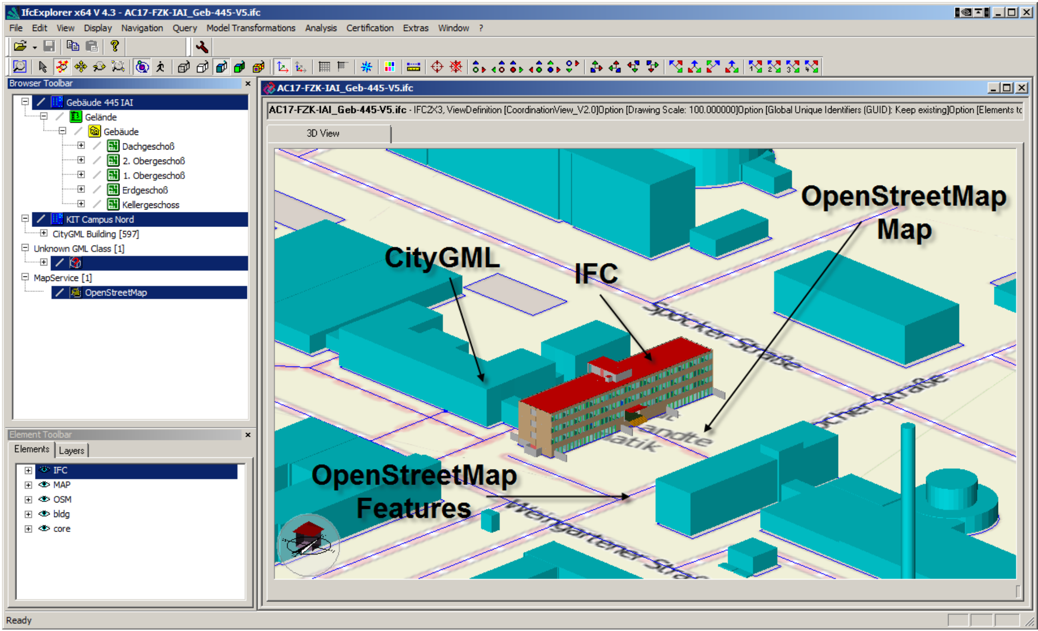

Figure 14: IFCExplorer scene, merging IFC, CityGML and OSM data

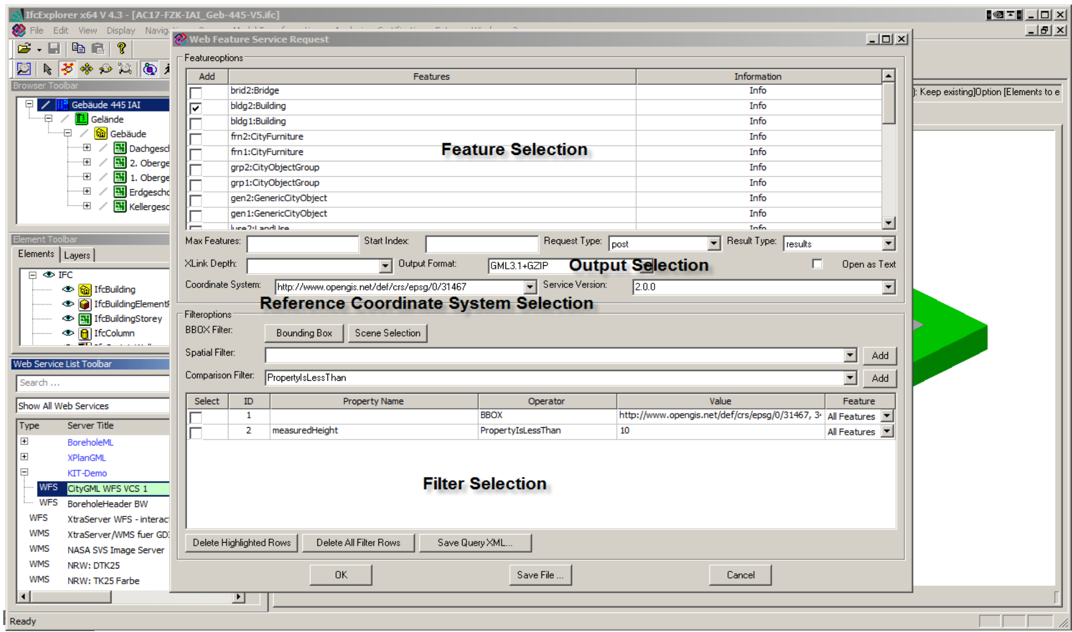

Figure 15: IFCExplorer user interface for configuring WFS requests

3.2.4 BIM-GIS authoring applications

Any information entering the PLM domain will either be acquired from existing information resources (such as mapping services, surveys, or FM inventories) or will be original content needing authoring by designers, engineering and product experts. BIM-GIS authoring applications can vary in many different characteristics: whilst some general purpose tools dominate the marketplace, there are many specific applications which are more closely tailored to a small subset of the target domain. For STREAMER, the target domain is potentially very broad, but the key requirement is that any application must be able to export to open standard data schemas and formats such as IFC.
The following paragraphs highlight some of the key dimensions along which authoring applications may be distinguished.

- Dimensionality: Most BIM authoring systems are based in 3D geometry with some 2D capability. The 2D capability is a reflection of the dominance of the plan (vertical) view. GIS authoring systems are based primarily on 2D geometry, with some 3D capability. For GIS, the creation of a continuous 2D surface manifold, divided into adjacent and hierarchical regions limits their ability to represent topologically complex building and infrastructure forms such as tunnels and bridges. However such forms may be represented as supplementary features located above or below the manifold.

- Scale: Most authoring applications have a target scale or scope in the geospatial domain: typically GIS applications focus on the Site upwards, with only indicative information within the typical curtilage of a hospital, where as BIM authoring applications may focus on the District downwards even to the level of material thicknesses.

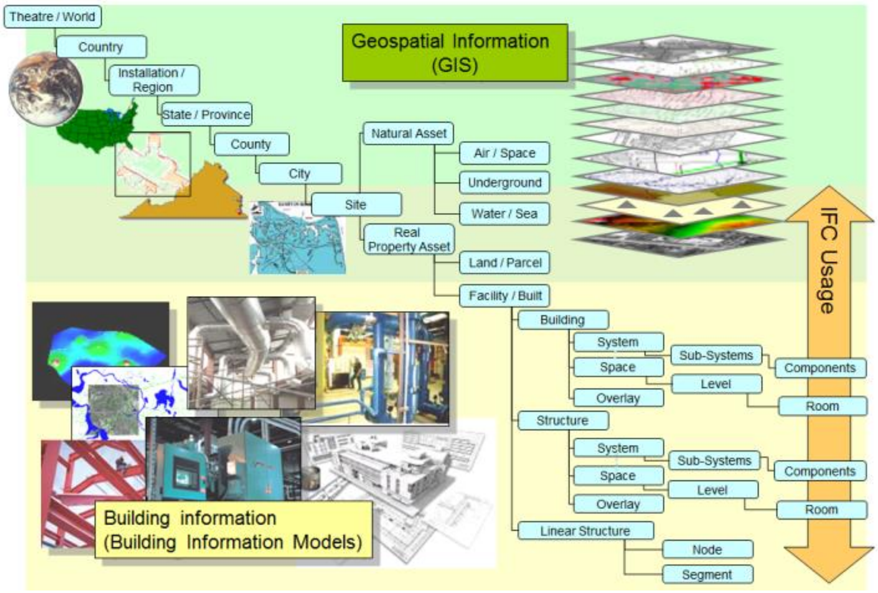

Figure 16: Scale across GIS/BIM authoring applications (OGC/buildignSMART)

- Time perspective: The difference of scale and dimensionality often creates a different focus on time: whereas GIS is applied to document the existing situation, BIM authoring applications for building and infrastructure are applied to capture and manage a change proposal within a finite project. The change may be predominantly new-build or it may be predominantly refurbishment. Given a project, some authoring applications will focus on requirements or on specific stages in the development of a project form conception through to handover and operation.

- Focus: Taken together, the scale and time perspective delimit a Facility, its Site and Project. Different applications may support the creation and development of corresponding Commencements, Locations and Tasks. For example a project planning application may enumerate all the planned tasks for a project, with some linkage to the Components involved, a 3D BIM authoring application may enumerate the physical Components with some linkage to their the Locations. To render the many Components, Locations or Tasks, these may be grouped into named Types, Systems, Zones, Regions/storeys, Packages and Job-Types. In contrast, GIS systems may offer a less structured approach to classification and grouping, and may not support the distinction between locations and components.

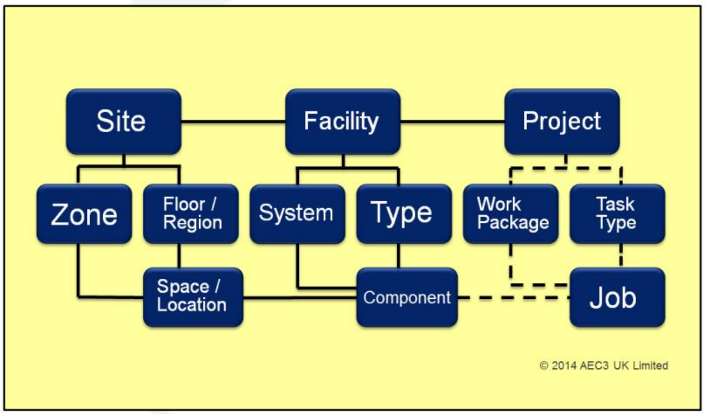

Figure 17: Authoring applications may address spatial, physical and/or process aspects (AEC3 Ltd)

- Detail: Both GIS and BIM authoring tools may use photogrammetry and imaging to provide a degree of realism, however the BIM authoring tools may include the intention to create increasingly realistic representations, especially for new-build where project outcomes need to be marketed and approved. BIM authoring applications may achieve a degree of completeness that enables automated ordering, machining and installation.

- Basis: The development of BIM authoring tools for building is focussed on creating an asset object based information model: geometry, symbols and data are necessarily attached to named objects. GIS and some BIM authoring tools for infrastructure have retained, and in some cases not progressed beyond, the ability to document indicators of boundaries and interfaces such as embankment edges. This restricts their ability to support non-graphic purposes such as measurement, analysis, reporting, and comparisons.

3.2.5 Thin client

The RE Suite is a comprehensive software tool that supports the collection, structuring, analysis and disclosure of real estate information. This software has a modular design, and it can be implemented in a modular system. The RE Suite software can be offered based on license or as SaaS contract (Software as a Service). The software is suitable for client-server environments, cloud applications, as well as stand-alone PCs.

The existing software RE Suite developed by DEMO (a partner in the STREAMER consortium) contains three tiers:

- RE Foundation
- RE Applications
- RE Cockpit

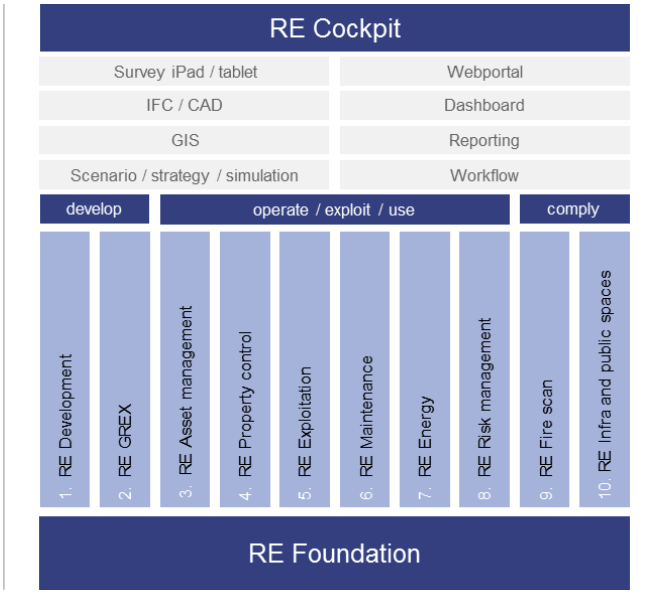

Figure 18: Main structure of the existing RE Suite software tool

The first tier is the RE Foundation, which is based on a server-solution that contains the data model as well as operates links to external sources and primary corporate information systems. The RE Foundation can manage data, such as the objects and components, as well as encode the users of the information system. The second tier of the RE Suite is the Application tier, containing applications such as Asset management, Maintenance, Energy. The third tier, the so-called RE Cockpit, contains generic user-oriented modules, such as a Dashboard, a Web Portal, GIS and IFC BIM viewers, and reporting modules. They can be implemented and used either integrated or independent from the RE Applications. The RE Cockpit modules can also be operated on mobile devices (tablets such as iPad and iPhone).

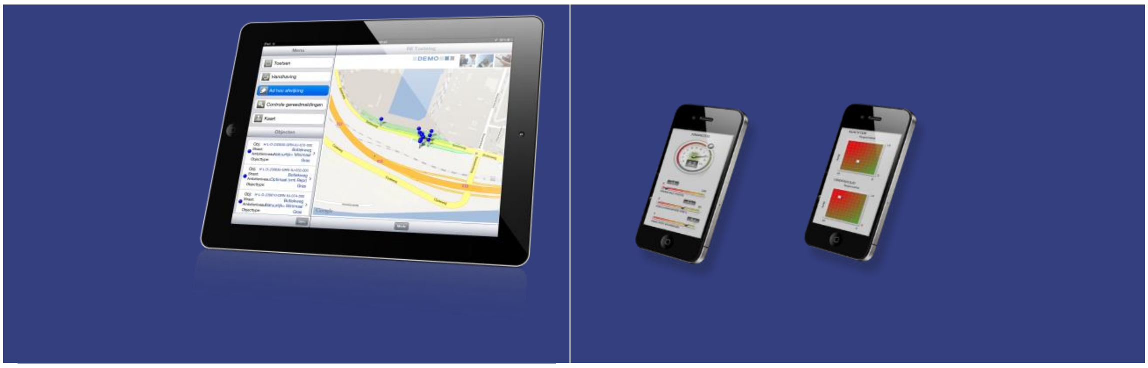

Figure 19: Illustrations of RE Suite use on mobile devices

The Dashboard is a visual representation of the most important information required to reach one or more goals, consolidated and arranged in a single screen so that the information can all be viewed at once. The most important features of a dashboard in the RE Suite are:

- All visualizations are on a single screen.
- (Key) performance indicators are visualized.
- Drill downs (list of mutations) can be shown.
- The information displayed is refreshed automatically, without user intervention.

Figure 20: Illustration of Dashboard in RE Suite

3.3 Product data

3.3.1 Structured product data

Within STREAMER context the relevant structured product data that needs to be addressed by the PLM architecture is BIM and GIS data. There are various standards capable of holding such data which are already described in D5.1 chapter 2 and D6.5 chapter 2. One aspect of the PLM is to store data and make it accessible via an interface towards the client application that connects to PLM.

In chapter 2.3.2 of D5.1 there is already a short description on data servers that exist today and are used for BIM or GIS data. In the current PLM architecture the BIMsever and Deegree server are selected to perform these tasks for respectively BIM and GIS data.

BIMserver

The BIMserver is a standalone solution that is able to store and manage BIM data, mainly expressed via the IFC data standard. Nevertheless it supports some other standards for importing and exporting data. Accessing data on the BIMserver can be done via the BIMsie API which is already described in D6.5 chapter 2.4. In the PLM architecture the clients will use this API to connect directly to the BIMserver.

Deegree server

The deegree server supports GML formats for storing data. Depending on the configuration it can be detailed to the CityGML standard which fits well to the GIS needs expected with the STREAMER project. Typical use of GIS data is making it available for at least viewing. This is why the deegree server lacks (like other GIS servers) any form of versioning of data.

Accessing data of the Deegree server can be done via 2 common OGC web service standards. The Web Map Service WMS sends as a response rendered images representing the GIS data. The area of interest and zoom level are part of the request. The Web Feature Service returns the results in a GML format. The configuration of the Deegree server as used by STREAMER will return CityGML like data.

3.3.2 Unstructured data

The unstructured data concerns all the documentation linked to the project except BIM files. The server in charge of storing this data is a EDM (Electronic Document Management). An EDM enables to organize and manage information and electronic documents within an organization.

The EDM operates mainly systems of acquisition, indexation, classification, storage, access and research for the diffusion of documents. It participates in the processes of collaboration, capitalization and information exchange.

Various solution of Document Server can be used in STREAMER project:

- **CMIS** (Content Management Interoperability): open standard managed by OASIS whose purpose is to increase the interoperability between the management systems of contents.
  CMIS supplies a common model of data covering types of files and directories with generic properties. It also describes a management system of the rights of access, contrl of version and offers the possibility to define generic relations.
- **Sharepoint**: it is a series of software for Web applications and portals developed by Microsoft. The features of products SharePoint are the management of contents, search engines, the electronic document management, the forums, the possibility of creating forms and decision-making statistics.
- **CSTB Document server** under development whose functionalities are described through the API in appendix 1.

3.4 Product Lifecycle Management data

3.4.1 Project Lifecycle Management data and services

The details of the data is described in appendix 1 through the API details.

3.4.2 Quality control data

BIM and GIS authoring applications are intended to generate internal efficiencies in terms of the cost or time required for the production and development of documentation sets. The production of multiple drawings, reports or schedules from a single information set will of itself act as a quality assurance mechanism. The use of appropriate authoring and data capture tools may enforce a degree of assurance, by disallowing unstructured or arbitrary input. In some cases the process of mapping to an external schema such as IFC may filter out or highlight non-conformant or incomplete data.

For these benefits to be compounded by re-using information across multiple stakeholders, the information (not the derivatives) must be of assured quality, so that rework, rekeying or review is minimised. This implies that checks should be introduced at the key information exchange points. It may be that both the provider(s) and the recipient(s) may have different priorities, depending on their roles: the development of well-defined information exchange requirement specification will bring together these into a single coherent requirement that can be applied to the exchange. Such exchanges may be collaborative or may be contractual.

Information exchanges are triggered with specific purposes in mind: there may be a duty to inform or consult or approve, or to develop the proposal further. Critical information exchanges may occur at gateways between specific life-cycle phases of the project (such as tender) and of the facility (such as handover).

Compliance

An agreed data standard and format such as IFC can serve to isolate the parties from specific authoring and analysis applications and checking processes, and to make the state independently verifiable. Beyond compliance to the specified schema and format, there are a number of levels of quality assurance. Most levels necessarily need a mixture of human and automated inspection.

Continuity

Whilst any individual information exchange may be specified and executed in isolation, both the receiver and recipient(s) may have a strong interest in continuity between information exchanges. This may be to support the detection of changes or in the comparison of alternative proposals. Continuity is achieved by maintaining GUIDs and names, unless there has been a genuine change to the design. Within any individual object is expected that attribute and relationship information will accumulate, so that new information may superseded older values, but the sudden disappearance of values is an issue, indicating a retrograde step.

In some cases there may be multiple independent representations of a facility that need to be converged: in this case the GUID or name may have been created independently in two places and other criteria may be needed to identify the same objects. This can arise when survey or mapping work is conducted in parallel with concept designs, or where semi-structured Facility Management information has been captured independently. Candidate methods for resolving this include their absolute location (two objects in the same spot may be identical), their unique (there is only one Site), or other characteristics.)

Completeness

The agreed scope of an exchange will be determined by the purpose. Some purposes may be progressively defined, such as cost estimates; others may be defined by an absolute requirement for a specific analysis or report. In some cases implications of a set of purposes may be grouped to define named 'Levels of Information' or 'Levels of geometry'. ISO 12911 uses a purpose driven approach, the UK 'Plain language Question' and 'digital plan of Works' uses assumed levels at the UK standard project gateways.

For an information exchange to be assured, it must relate to all (or some) of the agreed scope, typically the site, facility and project extents. Information outside of this, such as contractual relationships, commercial standing, or other context, may be included but is not normally expected. Some commercial organisations may hold facility and commercial information together, but an exchange that included such information may risk confidentiality and security.

Within the agreed scope, the information will exchange groupings of Components, Locations and Tasks. It may require manual checks to make sure that the lists of these are complete with reference to the actual or intended facility. There may be early stage checklists of accommodation or engineering approaches which can be used to verify the lists. This can be supported by data-driven checks such as does every Component belong to a System (or why else is it in the model?) or conversely does every System contain some Components. Similarly every location/space should have a region/storey, storeys without spaces may be redundant.

If the schedules of named aspects are to be exchanged, then it will be necessary that each has a unique identifier. This may be an internally generated globally unique identifier, or it may be a name that is at least locally unique, ideally within the scope of the facility, but often the name is unique only within a limited context, such as a pump P/8 which is unique within its space. Names may be supplemented with descriptive text which may serve to eliminate ambiguity.

Each of the aspects documented may suggest the association of references to labels from a chosen classification table. Ideally the chosen table will be comprehensive against the range of possible objects, and with an appropriate level of resolution as this coding may be the key semantic link between the model and other design processes. The choice of table is often dependant on locale: there are some product (Type) tables available from commercial standards, such as UNSPCC or eClass. In other cases the table of functional Systems expected is highly dependent on the type of Facility. Zone classification is less standardised. Work Package classification is usually specific to an individual constructor and published classification tables are rarely acceptable.

Any given asset object may be expected to have specific additional attributes. These may arise from the intrinsic nature of the objects (a door may need its width and height specified)) or from their extrinsic function (its position on a fire-boundary). The attributes expected may depend on the role of contributing author and the stage of the life cycles for which the information exchange is envisaged. However the IFC schema offers a small number of direct attributes and a large number of property and quantity sets for which some international consensus has been achieved. These may be supplemented by application or project specific expectations. In the case of STREAMER, the same requirements will be applied to the four pilot projects.

<u>Implementation</u>

For STREAMER, it has already been agreed that IFC and other interoperable standards will be used: compliance with IFC2x3 is readily testable by Express tools, or by using downstream viewers and applications for which the validity of the format is a pre-requisite.

STREAMER is focussed on several facilities, but the analysis of KPIs is aimed at a single design stage (early conceptual design). Continuity is therefore not expected to be a major issue, except where the assembly of diverse FM information together to make a single model is creating specific challenges.

Completeness assurance will be achieved by creating a single unified schedule of requirements that can be developed and tested by all the parallel pilot projects in WP7.

Requirements are being captured by the early work packages (WP1, WP2, WP3 and WP4). In WP5 they are being interpreted in the AEC3 “ReqCap” requirements capture tool. This tool organised requirements by authoring role, by project stage and purpose, by object type and by specific attributes. The requirements and their applicability and exceptions can be accumulated and developed. The output can be presented as checklists and reports that can act an annex to contracts and collaboration agreements, but the primary output is a set of testable requirement rules, delivered in a buildingSMART candidate standard mvdXML.

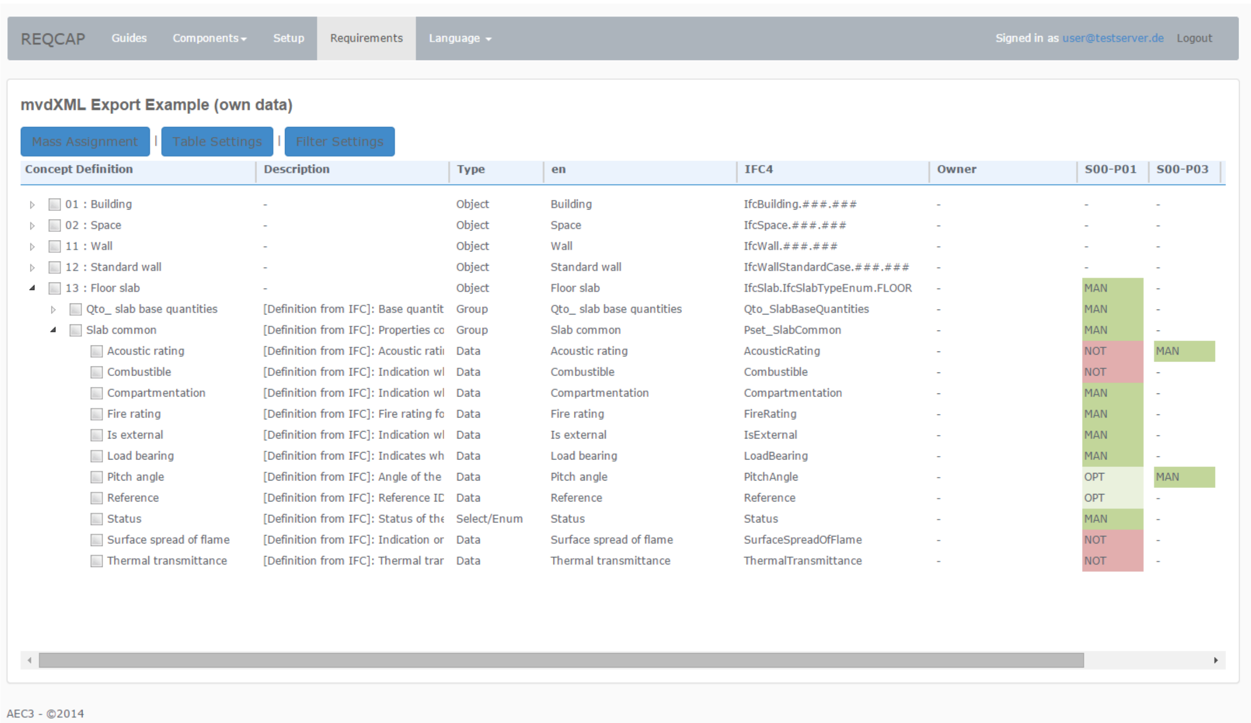

Figure 21: Requirements capture (AEC3 Ltd)

mvdXML is a concise format that contains a hierarchical list of expectations that can be tested to define expected content or to filter out unexpected content.

AEC3 ReqExplorer allows any set of IFC models to be grouped and checked against selected mvdXML requirements. The application includes a tree hierarchy, 3D view and the results panel which can highlight compliant, untested and defective objects. Issues can be reported as BCF messages. BCF messages are structured summaries of the issues found, in a format that can be used to open the authoring tools looking directly any Component or Location of concern.

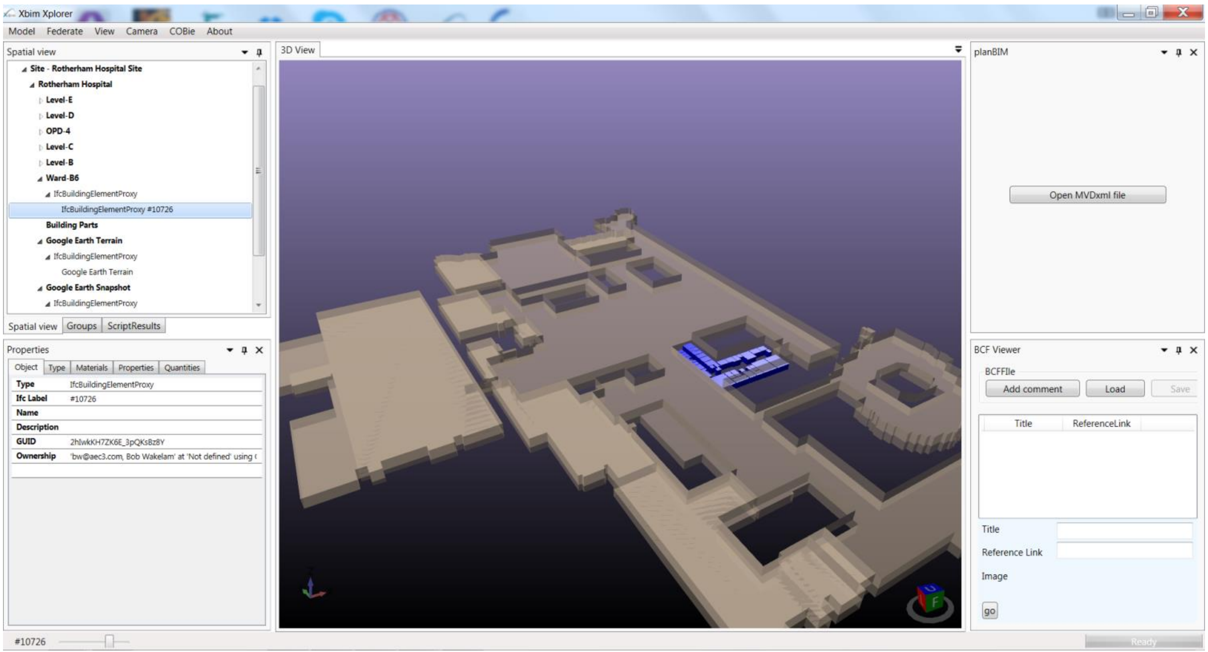

Figure 22: Requirement Explorer (AEC3 Ltd)

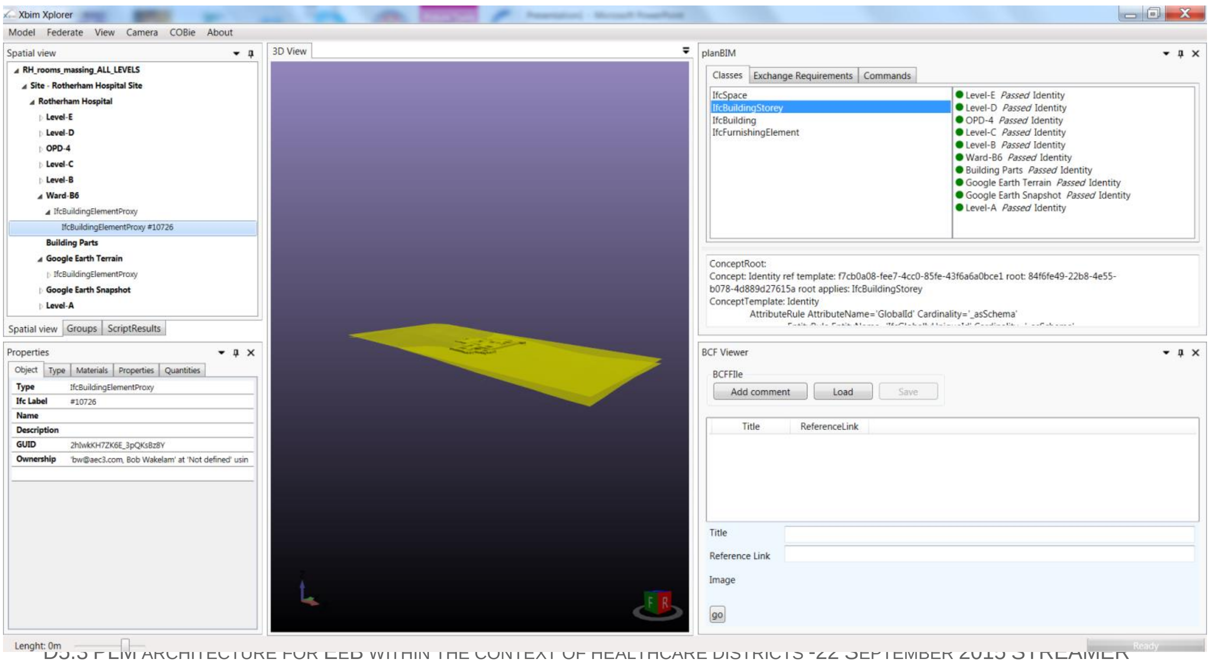

Figure 23: Checking for naming of Ifc Building Storey (AEC3 Ltd)

4. Conclusions

4.1 Relationship to other workpackages in STREAMER

The relationship with the other workpackages has been shown in paragraph 3.1.2 STREAMER PLM architecture.

In this architecture we see the link with **WP6** through the Design Configurator tool of KIT represented as a PLM client. It will connect the PLM server through the PLM API to access particular data or deposit new ones in the objective to store or get design results produced by the Design Configurator.

The same could be envisaged with the RE Suite tool from DMO resulting from **WP3**. The objective is to store all simulation data and results on the PLM server. The RE Suite application will then get the data through the PLM API to display and represent them on the DMO semantic interface.

The document also presents a link with task 5.1 ReqCap tool of **WP5** which appears in the Quality Control Data of the Product Lifecycle Management data. The data produced by the ReqCap tool will be stored in the Project Requirement Server and will appear in the PLM API as a validation process linked to a workflow.

4.2 Further work in STREAMER

The objective of this deliverable was to clarify what is behind the concept of PLM and to propose a PLM architecture based on Semantic Web technologies, coupled with the BIM, which could facilitate the collaborative work for construction or rehabilitation of Energy efficient Buildings within the context of healthcare districts.

The PLM solution proposed has been described regarding the inputs and outputs of the other workpackages and its functionalities are described via an API detailed in appendix. The objective of this API is to be as generic as possible in order to be able to connect to any PLM client in entrance and any type of data server in output.

Nevertheless they are many problematic:

- Being as generic as possible through the PLM API
- Being able to exchange data within various servers and to translate the data from a server to another one.
- Determine a good PLM that will enable to use the PLM API

In the case no fully generic PLM API can be implemented, the solution will be to reach directly the good server that contains the data instead of going through the upper layout called generic directory server.

The next step is to implement a PLM solution based on this architecture that will manage all the functionalities (or most of them) proposed. It will be the objectives of the second deliverable D5.4 of the current task.

References

[GEOINFORMATIC]

Benner et al. 3013: Benner, J., Geiger, A., Häfele, K.-H., Knüppel, H.: IFCExplorer – Ein Werkzeug für die Integration unterschiedlicher raumbezogener semantischer Daten. Geoinformatik, Heidelberg, 2013; http://www.geoinformatik2013.de/images/dokumente/gi2013_submission_13.pdf (last access 22.07.2015).

[PLM]

http://www.journeeduplm.fr/uploads/file/les_concepts_PLM2.pdf

http://www.journeeduplm.fr/uploads/jplm2011/Presentations/Atelier_Comprendre_PLM.pdf

[PLM and OMG standards]

http://www.prostep.org/fileadmin/user_upload/ProSTEPiViP/Standard-Infos/OMG_PLM_Services/PLM_Services_Ziel_und_Nutzen.pdf

[BIMSie]

https://buildingsmart.github.io/BIMSie/

[WFS]

http://www.opengeospatial.org/standards/wfs

APPENDIX 1: PLM API Services V0

1 Object model

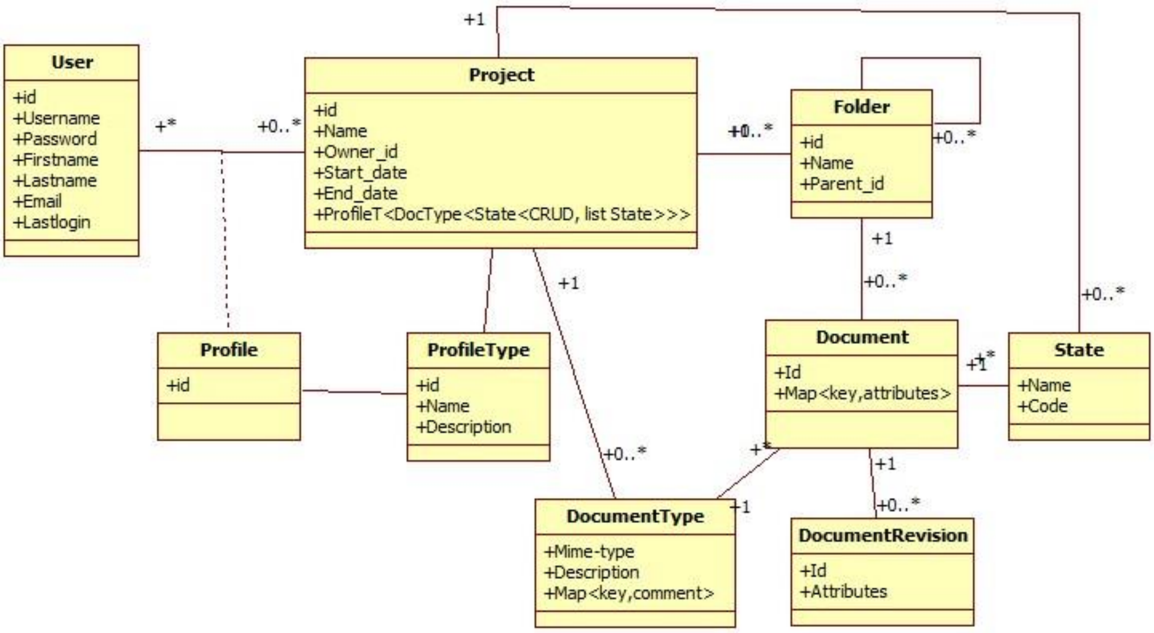

Figure 24: Web API Object Model and Services

This diagram represents the Web API Object Model and Services. It can be interpreted as following:

- A project is defined by:
  - A set of users
  - A set of profile types (BIM Manager, audit officer, project manager…)
  - A set of document types (Excel sheet, Ifc file, DWG file…)
  - A set of folders
  - A set of states (Deposit, To validate, Rejected…)
- A user takes part to a project according to his profile that corresponds to a profile type
- A document is defined by:
  - The folder in which it is stored
  - The document type it is coming from
  - A given state
  - A set of revisions

2 API definition

2.1 Right API

2.1.1 Authentification

2.1.1.1 Connection

Description
Connection of a user to the server via his username and password.

HTTP Method GET

URL plm-api/{version}/login
Example plm-api/1.0/login

JSON Structure

Name	Mandatory/Optional	Type	Description
username	Mandatory	String	The user's username
password	Mandatory	String	The user's password

Example request
HEADER {}
JSON
```
{
    "username":"Martin",
    "password":"MMa!55."
}
```

Example Response
```
{
    "token": "Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```

2.1.1.2 Deconnection

Description
Deconnection of a user to the server.

HTTP Method GET

URL plm-api/{version}/logout
Example plm-api/1.0/logout

JSON Structure

Name	Mandatory/Optional	Type	Description
token	Mandatory	String (guid)	The user's username

Example request
HEADER
```
{
    Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```
JSON
```
{}
```

Example Response
```
{
  Status 200 OK
```

}

2.1.2 Administration

2.1.2.1 Create a user

Description
Creation of a new user.

HTTP Method POST

URL plm-api/{version}/users
Example plm-api/1.0/users

JSON Structure

Name	Mandatory/Optional	Type	Description
Username	Mandatory	String	The user's username
Firstname	Mandatory	String	The user's firstname
Lastname	Mandatory	String	The user's lastname
Password	Mandatory	String	The user's password
Email	Optional	String	The user's email

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
"Username" : "Martin",
"Firstname" : "Martin",
"Lastname" : "Eric",
"Password" : "plm"
"Email" : eric.martin@gmail.com
}

Example Response
{
"Id" : "10",
"Username" : "Martin",
"Firstname" : "Martin",
"Lastname" : "Eric",
"Password" : "plm"
"Email" : eric.martin@gmail.com
}

2.1.2.2 Get user info

Description
Get the information of a user.

HTTP Method GET

URL plm-api/{version}/users/{user_id}
Example plm-api/1.0/users/10

Example request
HEADER

{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON {}

Example Response

{
"Username" : "Martin",
"Firstname" : "Martin",
"Lastname" : "Eric",
"Password" : "plm",
"Email" : eric.martin@gmail.com,
}

2.1.2.3 Modify user

Description
Modify the information of a user.

HTTP Method PUT

URL plm-api/{version}/users/{user_id}
Example plm-api/1.0/users/10

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
Password = "plm1"
}

Example Response

{
"Id" : "10",
"Username" : "Martin",
"Firstname" : "Martin",
"Lastname" : "Eric",
"Password" : "plm1",
"Email" : eric.martin@gmail.com
}

2.1.2.4 Remove a user

Description
Delete a user.

HTTP Method DELETE

URL plm-api/{version}/users/{user_id}
Example plm-api/1.0/users/10

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response

{
Status 200 OK
}

2.2 Document API

2.2.1 Project

2.2.1.1 Create a new project

Description
Creation of a new project on the server.

HTTP Method POST

URL plm-api/{version}/projects
Example plm-api/1.0/projects

JSON Structure

Name	Mandatory/Optional	Type	Description
Name	Mandatory	String	Name of the project
Owner_id	Mandatory	String (guid)	Owner of the project
Description	Optional	String	Short description of the project
Start_date	Optional	String (date)	Creation start of the project
End_date	Optional	String (date)	End of the project

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
```
{
    "Name" : "STREAMER",
    "Owner_id" : 15,
    "Description" : "Collaborative research project for EeB construction",
    "Start_date" : "2015-05-01",
    "End_date" : "2017-12-31"
}
```

Example Response
{
Status 200 OK
}

2.2.1.2 Get project info

Description
Get the information of a project.

HTTP Method GET

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/25

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}

JSON{}

Example Response

```
{
            "Id" : "25",
            "Name" : "STREAMER",
            "Owner_id" : "15",
            "Description" : "Collaborative research project for EeB construction",
            "Start_date" : "2015-05-01",
            "End_date" : "2017-12-31",
            "Users":   [24,25],
            "Folders": [12,32]
}
```

2.2.1.1 Modify the information of a project

Description

Change the information of a project.
HTTP Method PUT

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/12

Example request
HEADER

```
{
         Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```

JSON

```
{
            "Name": "STREAMER1"
}
```

Example Response

```
{
            "Id": "25",
            "Name": "STREAMER1",
            "Owner_id": "15",
            "Description" : "Collaborative research project for EeB construction",
            "Start_date": "2015-05-01",
            "End_date": "2017-12-31",
            "Members":["12","56"]
}
```

2.2.1.1 Remove a project

Description

Delete a project.
HTTP Method DELETE

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/15

Example request
HEADER

```
{
         Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```

JSON{}

Example Response

{
Status 200 OK
}

2.2.1.2 Add user to project

Description
Add a new user to a project.

HTTP Method POST

URL plm-api/{version}/profiles
Example plm-api/1.0/profiles

JSON Structure

Name	Mandatory/Optional	Type	Description
User_id	Mandatory	String (guid)	Id of the user
Profile_type_id	Mandatory	String (guid)	Id of the profile_type
Project_id	Mandatory	String (guid)	Id of the project

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
"User_id " : "10",
"Profile_type_id" : "25",
"Project_id" : "55"
}

Example Response
{
Status 200 OK
}

2.2.1.3 Remove user to project

Description
Remove a user from a project.

HTTP Method DELETE

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/50

JSON Structure

Name	Mandatory/Optional	Type	Description
User_id	Mandatory	String (guid)	Id of the user

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
"User_id " : "10"

}

Example Response

```
{
        Status 200 OK
}
```

2.2.1.4 Get user projects

Description

Retrieve a collection of projects where the currently logged on user has access to.
HTTP Method GET

URL plm-api/{version}/projects
Example plm-api/1.0/projects

Example request

HEADER

```
{
        Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}
```

Example Response


```
{
        "Projets" : [
        {
                "Id": "25",
                "Name": "STREAMER",
                "Owner_id": "15",
                "Description" : "Collaborative research project for EeB construction",
                "Start_date": "2015-05-01",
                "End_date": "2017-12-31",
                "Members":["12","56"]
        },
        {
                "Id": "12",
                "Name": "ProjectX",
                "Owner_id": "32",
                "Description" : "…",
                "Start_date": "2014-12-03",
                "End_date": "2018-12-31",
                "Members":["12","56"]
        }]
}
```


2.2.1.5 Get all the documents of a project (by type)

Description

Retrieve a collection of documents of a given document typecontained in the selected project where the currently logged on user has access to.
HTTP Method GET

URL plm-api/{version}/documents?project_id={project_id}&document_type_id={document_type_id}
Example plm-api/1.0/documents?project_id=12&document_type_id=5

Example request

HEADER

```
{
        Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
```

}
JSON{}

Example Response

```
{
        "Documents" : [
        {
                "Id": "25",
                "Description": "Computation results of energy simulation"
                "data": "…."
        },
        {
                "Id": "12",
                "Description"  :"The legal requirements for buildings."
                "data": "…."
        }]
}
```

2.2.2 Folder

2.2.2.1 Create a folder

Description
Creation of a new folder in a given project.

HTTP Method POST

URL plm-api/{version}/folders
Example plm-api/1.0/folders

JSON Structure

Name	Mandatory/Optional	Type	Description
Project_id	Mandatory	String (guid)	Id of the project
Parent_id	Mandatory	String (guid)	Id of the parent (folder or project)
Name	Mandatory	String	Name of the folder

Example request
HEADER

```
{
        Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```

JSON

```
{
        "Project_id " : "52",
        "Parent_id " : "15",
        "Name" : "AdministrativeDocuments"
}
```

Example Response

```
{
        "id " : "50"
}
```

2.2.2.2 Get folder info

Description
Get the information of a folder.

HTTP Method GET

URL plm-api/{version}/folders/{folder_id}
Example plm-api/1.0/folders/25

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response
{
"Id" : "25",
"Name" : "AdministrativeDocuments",
"Project_id" : "15",
"Parent_id" : "52",
"Folders": [12,32]
"Documents":[5,24,25]
}

2.2.2.3 Modify a folder

Description
Modify the information of a folder.

HTTP Method PUT

URL plm-api/{version}/folders/{folder_id}
Example plm-api/1.0/folders/25

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
Name = "AdministrativeDoumentsV2"
}

Example Response
{
"Id" : "25",
"Name" : "AdministrativeDoumentsV2",
"Project_id" : "15",
"Parent_id" : "52",
"Folders": [12,32]
"Documents":[5,24,25]
}

2.2.2.4 Remove a folder

Description
Delete a folder.
HTTP Method DELETE

URL plm-api/{version}/folders/{folder_id}
Example plm-api/1.0/folders/30

Example request
HEADER
{

Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response
{
Status 200 OK
}

2.2.3 Profile type

2.2.3.1 Create a profile type

Description
Creation of a new profile type.

HTTP Method POST

URL plm-api/{version}/projects/{project_id}/profiles
Example plm-api/1.0/projects/25/profiles

JSON Structure

Name	Mandatory/Optional	Type	Description
Name	Mandatory	String	Name of the profile type
Description	Mandatory	String	Description of the profile type

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
"Name " : "BIMManager",
"Description " : " Group of persons that manage BIM processus.",
}

Example Response
{
"id " : "12"
}

2.2.3.1 Get profile info

Description
Get the information of a profile.

HTTP Method GET

URL plm-api/{version}/profiles/{profile_id}
Example plm-api/1.0/profiles/12

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{}

Example Response

{
"Id" : "12",
"Name " : "BIM-Manager",
"Description " : " Group of persons that manage BIM processus.",
}

2.2.3.1 Modify a profile

Description
Modify the information of a profile.

HTTP Method PUT

URL plm-api/{version}/profiles/{profile_id}
Example plm-api/1.0/profiles/12

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
Name = " BIM-Manager "
}

Example Response
{
"Name " : "BIM-Manager",
"Description " : " Group of persons that manage BIM processus.",
}

2.2.3.2 Remove a profile

Description
Remove a profile.
HTTP Method DELETE

URL plm-api/{version}/profiles/{profile_id}
Example plm-api/1.0/profiles/12

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response
{
Status 200 OK
}

2.2.3.3 Associate user to profile type

Done in 2.2.1.1 Modify the information of a project
Description
Change the information of a project.
HTTP Method PUT

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/12

Example request
HEADER
{
    Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
    "Name": "STREAMER1"
}

Example Response
{
    "Id": "25",
    "Name": "STREAMER1",
    "Owner_id": "15",
    "Description" : "Collaborative research project for EeB construction",
    "Start_date": "2015-05-01",
    "End_date": "2017-12-31",
    "Members":["12","56"]
}

2.2.3.4 Remove a project

Description

Delete a project.
HTTP Method DELETE

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/15

Example request
HEADER
{
    Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response
{
    Status 200 OK
}
Add user to project.

2.2.4 Document type

2.2.4.1 Create a document type

Description
Creation of a new document type.

HTTP Method POST

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/25

JSON Structure

Name	Mandatory/Optional	Type	Description
Mime_type	Mandatory	String	Name of the profile type
Description	Mandatory	Object	A JSON string that describes the

			document type.

Example request
HEADER
```
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```
JSON
```
{
	"Mime_type" : "Ifc",
	"Description" :
	{
		"Phase":[
			Comment: Project phase,
			Values: [APS, APD,…]
			],
		 "Lot":[..],
		…
	}
}
```

Example Response
```
{
	"Id " : "12",
	"Mime_type" : "Ifc",
	"Description" :
	{
		"Phase":[
			Comment: Project phase,
			Values: [APS, APD,…]
			],
		 "Lot":[..],
		…
	}
}
```

2.2.4.1 Get document type info

Description
Get the information of a document type.

HTTP Method GET

URL plm-api/{version}/document_types/{document_type_id}
Example plm-api/1.0/document_types/12

Example request
HEADER
```
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}
```

Example Response
```
{
	"Id " : "12",
	"Mime_type" : "Ifc",
	"Description" :
	{
		"Phase":[
```

Comment: Project phase,
Values: [APS, APD,…]
],
“Lot”:[..],
…
}
}

2.2.4.1 Modify a document type

Description
Modify the information of a document type.

HTTP Method PUT

URL plm-api/{version}/document_types/{document_type_id}
Example plm-api/1.0/document_types/12

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
"Mime_type" : "Excel",
}

Example Response
{
"Id " : "12",
"Mime_type" : " Excel ",
"Description" :
{
"Phase":[
"Comment": "Project phase",
"Values: ["APS", "APD",…]
],
"Lot":[..],
…
}
}

2.2.4.2 Remove a document type

Description

Remove a document type.
HTTP Method DELETE

URL plm-api/{version}/document_types/{document_type_id}
Example plm-api/1.0/document_types/12

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response
{

Status 200 OK
}

2.2.5 Document

2.2.5.1 Create a document

Description
Creation of a new document.

HTTP Method POST

URL plm-api/{version}/documents
Example plm-api/1.0/documents

JSON Structure

Name	Mandatory/Optional	Type	Description
Document_type_id	Mandatory	String (guid)	Id of the document_type
Attributes	Mandatory	Object	A JSON string that describes the document.

Example request
HEADER
```
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```
JSON
```
{
	"Document_type_id": "15",
	"Attributes":
	{
		"Phase": "APS",
		"Lot": …
		…
	}
}
```

Example Response
```
{
	"id": 50,
	"Document_type_id": "15",
	"Attributes":
	{
		"Phase": "APS",
		"Lot": …
		…
	}
}
```

2.2.5.1 Get a document

Description
Get the information of a document.

HTTP Method GET

URL plm-api/{version}/documents/{document_id}
Example plm-api/1.0/documents/50

Example request

HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response

```
{
        "Revisions":[
        {
        "A":
                {
                        "id": 50,
                        "data": […],
                        "Document_type_id": "15",
                        "Attributes":
                        {
                                "Phase": "APS",
                                "Lot": …
                        }
                },
                "B":
                {
                        "id": 50,
                        "data": […],
                        "Document_type_id": "15",
                        "Attributes":
                        {
                                "Phase": "APS",
                                "Lot": …
                        }
                }
        }
}
```

2.2.5.1 Modify a document

Description
Modify the information of a document.

HTTP Method PUT

URL plm-api/{version}/documents/{document_id}
Example plm-api/1.0/documents/50

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON

```
{
        "B":
        {
                "id": 50,
                "data": […],
                "Document_type_id": "15",
                "Attributes":
                {
                        "Phase": "APD",
                        "Lot": …
```

```
                }
        }
}
```


Example Response


```
{
        "Revisions":[
        {
        "A":
                {
                        "id": 50,
                        "data": […],
                        "Document_type_id": "15",
                        "Attributes":
                        {
                                "Phase": "APS",
                                "Lot": …
                        }
                },
                "B":
                {
                        "id": 50,
                        "data": […],
                        "Document_type_id": "15",
                        "Attributes":
                        {
                                "Phase": "APD",
                                "Lot": …
                        }
                }
        }
}
```


2.2.5.2 Remove a document

Description

Remove a document.
HTTP Method DELETE

URL plm-api/{version}/documents/{document_type_id}
Example plm-api/1.0/documents/50

Example request
HEADER

```
{
        Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}
```

Example Response

```
{
        Status 200 OK
}
```

2.2.6 Document revision

2.2.6.1 Create a new revision of a document

Description
Creation of a new revision of a document.

HTTP Method POST

URL plm-api/{version}/documents/{document_id}
Example plm-api/1.0/documents/50

JSON Structure

Name	Mandatory/Optional	Type	Description
Document_type_id	Mandatory	String (guid)	Id of the document_type
Attributes	Mandatory	Object	A JSON string that describes the document.

Example request
HEADER
```
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
```
JSON

```
{
	"Data" : "…",
	"Revision_object":
	{
		"id": 50,
		"Document_type_id": "15",
		"Attributes":
		{
			"Phase": "APS",
			"Lot": …
		}

	}

}
```


Example Response

```
{
	"Data" : "…",
	"Revision_object":
	{
		"id": 50,
		"Document_type_id": "15",
		"Attributes":
		{
			"Phase": "APS",
			"Lot": …
		}
	}
}
```


2.2.6.2 Get document revision

Description
Get the revision of a document.

HTTP Method GET

URL plm-api/{version}/documents/{document_id}?{revision_id}=id
Example plm-api/1.0/documents/50?revision=A

Example request

```
HEADER
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}
```

Example Response

```
{
	"Revisions":[
	{
	"A":
		{
			"id": 50,
			"data": […],
			"Document_type_id": "15",
			"Attributes":
			{
				"Phase": “APS”,
				"Lot”: …
			}
		}
	}
}
```

2.3 Process API

2.3.1 State

2.3.1.1 Create a new state

Description
Creation of a new state of a document.

HTTP Method POST

URL plm-api/{version}/projects/{project_id}
Example plm-api/1.0/projects/25

JSON Structure

Name	Mandatory/Optional	Type	Description
Name	Mandatory	String	Name of the state
Code	Mandatory	String	System codification of the state

Example request

```
HEADER
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
	"Name":”A valider”,
	"Code":”TOVALIDATE”
}
```

Example Response

```
{
	"id": 50
}
```

2.3.1.1 Get a state info

Description
Get the information on a state of a document.

HTTP Method GET

URL plm-api/{version}/states/{state_id}
Example plm-api/1.0/states/50

Example request
HEADER
```
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}
```

Example Response
```
{
	"Id " : "50",
	"Name":"A valider",
	"Code":"TOVALIDATE"
}
```

2.3.1.1 Modify a state

Description
Modify the state of a document.

HTTP Method PUT

URL plm-api/{version}/states/{state_id}
Example plm-api/1.0/states/50

Example request
HEADER
```
{
	Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
	Code:"TO_VALIDATE"
}
```

Example Response
```
{
	"Id " : "50",
	"Name":"A valider",
	"Code":"TO_VALIDATE"
}
```

2.3.1.2 Remove a state

Description
Remove the state of a document.
HTTP Method DELETE

URL plm-api/{version}/states/{state_id}
Example plm-api/1.0/states/50

Example request
HEADER
{

Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response
{
Status 200 OK
}

2.3.2 Workflow

2.3.2.1 Create a workflow

Description
Creation of a workflow on a document.

HTTP Method POST

URL plm-api/{version}/workflows
Example plm-api/1.0/workflows

JSON Structure

Name	Mandatory/Optional	Type	Description
Project_id	Mandatory	String (guid)	Id of the project
Document_type_id	Mandatory	String (guid)	Id of the document type
Profile_id	Mandatory	String (guid)	Id of the profile
State_id	Mandatory	String (guid)	Id of the state
Available_states	Mandatory	Object	A JSON string that describes the available states of the workflow.

Example request
HEADER
{
Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
"Name":"A valider",
"Code":"TOVALIDATE"
}

Example Response
{
"id":"10",
"Project_id":"25",
"Document_type_id":"11",
"Profile_id":"55",
"State_id":"68",
"Available_states":["1","52","63"]
}

2.3.2.1 Get a workflow info

Description
Get the information on a workflow of a document.

HTTP Method GET

URL plm-api/{version}/workflows/{workflow_id}

Example plm-api/1.0/workflows/10

Example request
HEADER
{
    Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON{}

Example Response
{
    "id":"10",
    "Project_id":"25",
    "Document_type_id":"11",
    "Profile_id":"55",
    "State_id":"68",
    "Available_states":["1","52","63"]
}

2.3.2.1 Modify a workflow

Description
Modify the workflow of a document.

HTTP Method PUT

URL plm-api/{version}/workflows/{workflow_id}
Example plm-api/1.0/workflows/50

Example request
HEADER
{
    Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"
}
JSON
{
    "Available_states":["12","52","63"]
}

Example Response
{
    "id":"10",
    "Project_id":"25",
    "Document_type_id":"11",
    "Profile_id":"55",
    "State_id":"68",
    "Available_states":["12","52","63"]
}

2.3.2.2 Remove a workflow

Description
Remove the workflow of a document.
HTTP Method DELETE

URL plm-api/{version}/workflows/{workflow_id}
Example plm-api/1.0}/workflows/10

Example request
HEADER
{
    Authorization : "basic Ym9ic2Vzc2lvbjE6czNjcmV0"

}
JSON{}

Example Response

{
Status 200 OK
}