MOEEBIUS

Modelling Optimization of Energy Efficiency in Buildings for Urban Sustainability

D3.2. MOEEBIUS Common Information Model

Version number: 2.0
Dissemination Level: PU
Lead Partner: TECNALIA
Due date: 31/10/2016
Type of deliverable: R
STATUS: Delivered

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 680517

Published in the framework of:

MOEEBIUS - Modelling Optimization of Energy Efficiency in Buildings for Urban Sustainability

MOEEBIUS website: www.moeebius.eu

Authors:

Borja Tellado, Alberto Armijo, Asier Mediavilla, Ander Romero, Pablo de Agustín - TECNALIA

Yashar Kalantari, Georgios Kontes - THN

Tsatsakis Konstantinos - HYPERTECH

Javier Biosca, Javier Royo – SOL

Revision and history chart:

VERSION	DATE	EDITORS	COMMENT
1.0	31/10/16	TECNALIA	Submitted to the EC
2.0	23/12/16	TECNALIA	Revisited version to accommodate to D3.1 structure

Disclaimer:

Table of content

List of tables

List of figures

Glossary

Acronym	Full name
AEC	Architecture, Engineering & Construction
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
BDAE	Building Dynamic Assessment Engine
BEPS	Building Energy Performance Simulation
BIM	Building Information Model
BMS	Building Management System
CIM	Common Information Model
DAE	Dynamic Assessment Engine
DAFM	Demand Aggregation, Flexibility and Management
DER	Distributed Energy Resource
DFMPC	Distributed Fuzzy Model Predictive Control
DoA	Description of Actions
DSM	Demand Side Management
DSS	Decision Support System
EPW	EnergyPlus Weather File
ESB	Enterprise Service Bus
ESCO	Energy Service Company
FDD	Fault Detection and Diagnosis
FM	Facility Manager
gbXML	Green Building XML schema
GIS	Geographic Information System
GUI	Graphical User Interface
HPXML	Home Performance XML
HVAC	Heating Ventilation Air Conditioning
IAQ	Indoor Air Quality
IDD	Input Data Dictionary
IFC	Industry Foundation Classes
IoT	Internet of Things
JSON	JavaScript Object Notation
KML	Keyhole Markup Language
KPIs	Key Performance Indicators
LCA/ LCC	Life-Cycle Assessment/ Life-Cycle Cost
MOEEBIUS	Modelling Optimization of Energy Efficiency in Buildings for Urban Sustainability
MQTT	Message Queue Telemetry Transport
OPENADR	Open Automated Demand Response
OSM	OpenStreetMap
PM	Particulate Matter
REST	Representational state transfer
SHP	ESRI Shapefile
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
VOC	Volatile Organic Compounds
VR	Virtual Reality
WSDL	Web Services Description Language
WSN	Wireless Sensor Network
XML	eXtensible Markup Language

1 Executive summary

The MOEEBIUS project introduces a Holistic Energy Performance Optimization Framework that enhances current static and dynamic modelling approaches and delivers innovative simulation tools, which, on the one hand, describe real-life building operation complexities in accurate simulation predictions that significantly reduce the "performance gap" and, on the other hand, provide continuous optimization of building energy performance as a means to further mitigate and reduce the identified "performance gap" in real-time or through retrofitting.

This document is the outcome of the task "*T3.2 Common Information Model Definition (M7-M12)*", framed under work package "*WP3 - MOEEBIUS Architecture and Data Model Definition (M7-17)*". The goal of this document is the definition of the MOEEBIUS Common Information Model (CIM), which will describe the high level domain model comprising the basic elements (semantic concepts, events, relations, interfaces, etc.) and will support the information needs of the components underlying the MOEEBIUS project.

The CIM model establishes the basis upon which further information modelling work will be performed in tasks "*T3.3 DER and District Heating Modelling (M10-M15)*", "*T3.4 Occupants Comfort Modelling (M10-M15)*", "*T3.5 Indoor Air Quality Modelling (M10-M15)*" and "*T3.6 Local and Global Energy Performance Modelling (M9-M17)*". The results of these tasks will be leveraged in "*WP4 Data Acquisition and Management*" & "*WP5 MOEEBIUS Simulation-Based Dynamic Assessment Environment*" and will facilitate the integration of the different components under the common MOEEBIUS framework.

The document builds upon the holistic business scenario "**DSS towards the establishment of a sustainable building level and district level environment"** presented in "*D2.4 Functional and Non-functional requirements of the MOEEBIUS framework and individual components*". The goal of this business scenario is to leverage a holistic and dynamic modelling and simulation / optimization approach that enables (1) Improved predictions on the basis of more accurate and dynamically updated Building Energy Performance Simulation (BEPS) models, (2) Real-time building and district performance optimization through control and predictive maintenance and (3) Optimized retrofitting decision making on the basis of improved performance predictions. Taking the holistic scenario as reference, the document further elaborates on the data exchange requirements envisaged for each of the components. In this regard, connections to the relevant sections in the building industry reference data models are provided.

This is the second version of this deliverable, which has been revisited for accommodating it to the structure of *D3.1 MOEEBIUS Framework Architecture including functional, technical and communication specifications,* once D3.1 has been released.

2 Introduction

The MOEEBIUS framework [1] will support the operation and maintenance of buildings and districts through the configuration and integration of an innovative suite of end-user tools and applications, enabling:

1 Improved and real-time Building Energy Performance Simulation (BEPS) Assessment and performance optimization (during the operation and maintenance phase)
2 Optimized retrofitting decision making on the basis of improved and accurate performance predictions
3 Real-time peak-load management optimization at the district level

The provision of a robust technological framework will enable ESCOs, Aggregators, Maintenance Companies and Facility Managers the creation of attractive business opportunities in evolving and leveraging highly competitive energy services markets. While the overall MOEEBIUS reference architecture is reported in D3.1, here we are providing the common information model among the different system components of MOEEBIUS framework. The MOEEBIUS framework will be validated in 3 large-scale pilot sites, located in Portugal, UK and Serbia, incorporating diverse building typologies, heterogeneous energy systems and spanning diverse climatic conditions.

2.1 Scope of this Deliverable

This deliverable presents the results of the task "*T3.2 Common Information Model Definition (M7-M12)*". It highlights the basic information requirements that will be needed by the components of the MOEEBIUS system architecture [3] at building and district level. Taking into account the input/output requirements derived from the requirements analysis driven by the end-users, and the architecture definition, a domain model was conceived. This model, referred to as the Common Information Model (CIM), will provide the MOEEBIUS components with data / information provision services that support their activities.

Following the approach developed in "*T3.1 MOEEBIUS Architectural Design and Technological/ Functional Specifications - Communication interfaces and protocols definition*", some of the identified communication protocols and data modelling approaches that are more suitable to be leveraged in data exchange processes between the MOEEBIUS components are presented.

Subsequently, some building industry reference data models are presented. Based on the input/output requirements for the MOEEBIUS components described in [3], some of the most prominent data model standards (at building and district level) were selected and applied to comply with their data requirements. Thus, some data classes were defined for each of the components based on the above mentioned standards. In order to leverage interoperability and "open" access,

commonly used standards will be considered and extended as necessary to fit the use case requirements.

Finally, the MOEEBIUS data management/exchange framework is presented. This framework, acting as a data services server, will provide methods to access the Common Information Model. It describes the way the MOEEBIUS components will handle the information they require.

Although this report is delivered according to the DoA in Month 12, the MOEEBIUS components in terms of data requirements will follow an iterative process until the middleware and the components are integrated into the pilots. Consequently, this deliverable can be considered as a working document that will guide and track the data requirements of the different components that comprise the MOEEBIUS framework.

2.2 Deliverable Structure

The deliverable is structured and organized in the following chapters:

- Chapter 2 is the introduction of this document, outlining the main objectives and the purpose of the report
- Chapter 3 presents the Common Information Model Context, including the Data Services Server philosophy and the data exchange process in the frame of the MOEEBIUS middleware components
- Chapter 4 presents the communication and data modelling approaches that will be followed to leverage interoperability between the components in terms of data provision
- Chapter 5 describes the data classes that comprise the Common Information Model, which will be implemented and made available through a data services server in order to enable data provision to the different MOEEBIUS components. A description of the standards that apply at building and district level is provided along with discussion on the relevance of the application of these standards to support the MOEEBIUS approach.
- Chapter 6 defines the MOEEBIUS data services server, which will be the application that facilitates the data exchange with the different modules according to the Common Information Model derived from the standards and the component input/output requirements
- Finally, in Chapter 7 conclusions are drawn for this report, focusing on the interconnection of T3.2 with the upcoming work in the project.

3 Common Information Model Context

The next subchapters explain the ideas behind the Common Information Model (CMI) and the data exchange process around the MOEEBIUS semantic model. On the other hand, a contextualization of the Common Information Model within the whole MOEEBIUS platform (middleware + services) is presented based the holistic business scenario shown in "*D2.4 Functional and Non-functional requirements of the MOEEBIUS framework and individual components*".

3.1 CIM / Data Services Server Philosophy

The idea behind the Common Information Model (CIM) and the associated data management / exchange framework is to replace module-2-module interactions by a common model + common API approach (see Figure 1). The data exchange framework or data services server will provide a single data access point for all the components. Specifically, the data management / exchange framework will be part of the Data Acquisition and Management Layer middleware (see Figure 5) and will provide semantically enhanced interfaces available to the rest of MOEEBIUS components.

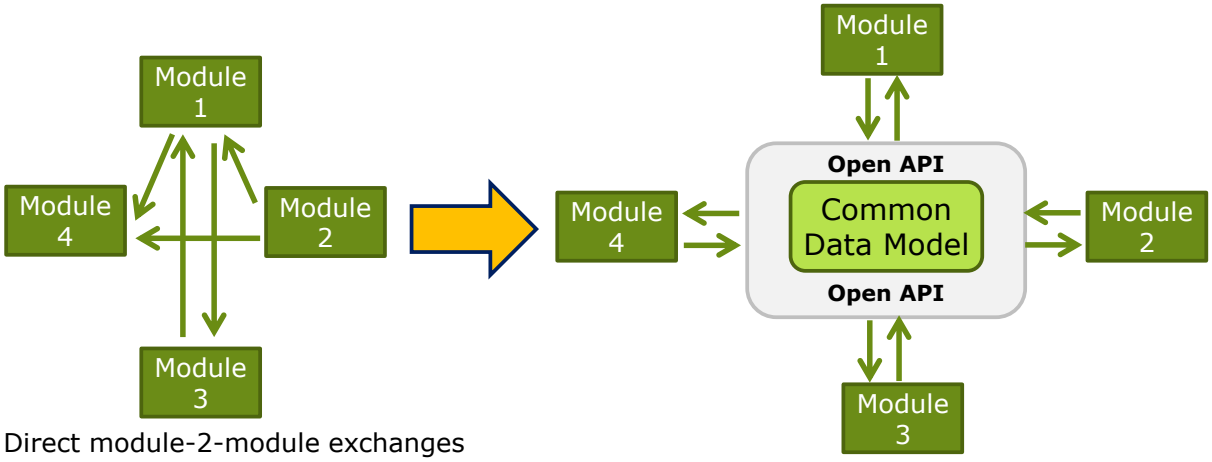

Figure 1 Module-2-module interactions vs. Common API approach

Thus, the integration pattern of the data exchange framework within the service orchestration supported by the middleware provides two communication possibilities as depictured in Figure 2. Option 1 entails the provision of data by the data services server to the components through direct end point (EP) calls. Another option is to consider the data services server as a component and orchestrate the composition of services through an ESB (part of the middleware). This implies that the provision of data is handled by the service bus.

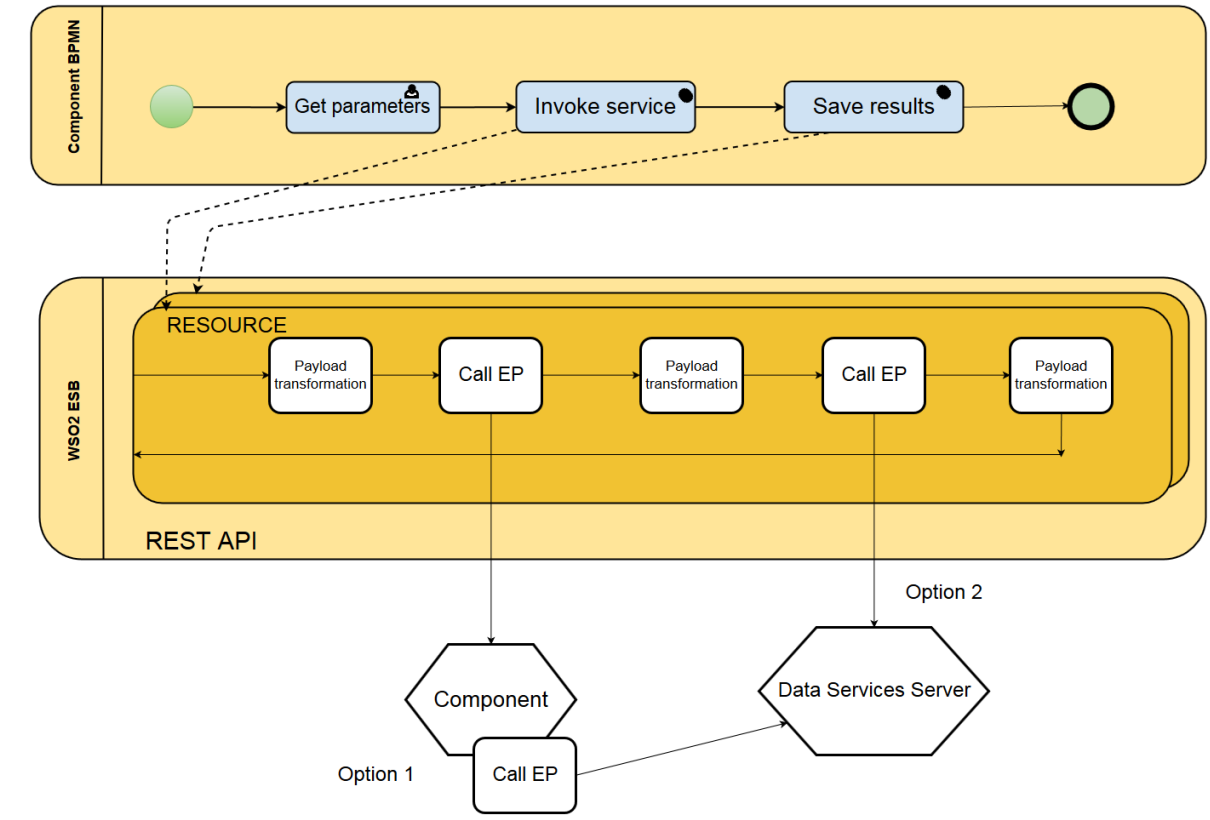

Figure 2 Module interaction with the data services server

3.2 Data exchange process

The data exchange process between modules and the service bus entails the next steps:

1. Identify data (input) requirements for each module (component). The specification, architecture and requirement deliverables (D3.1 and D2.4) are the starting point for the identification of these data exchange requirements
2. Identify which module (or external source) produces the data (see Figure 3)

Figure 3 Data exchange procedure

3. As an important assumption, we consider that the data internally handled by a specific component is not part of the Common Data Model. Furthermore, some data delivered by a component that is to be directly consumed by another component neither is considered part of the Common Data Model.

A methodology to identify the required data by the components is to split the component models into sub-models and perform an individual analysis, as depictured in Figure 4.

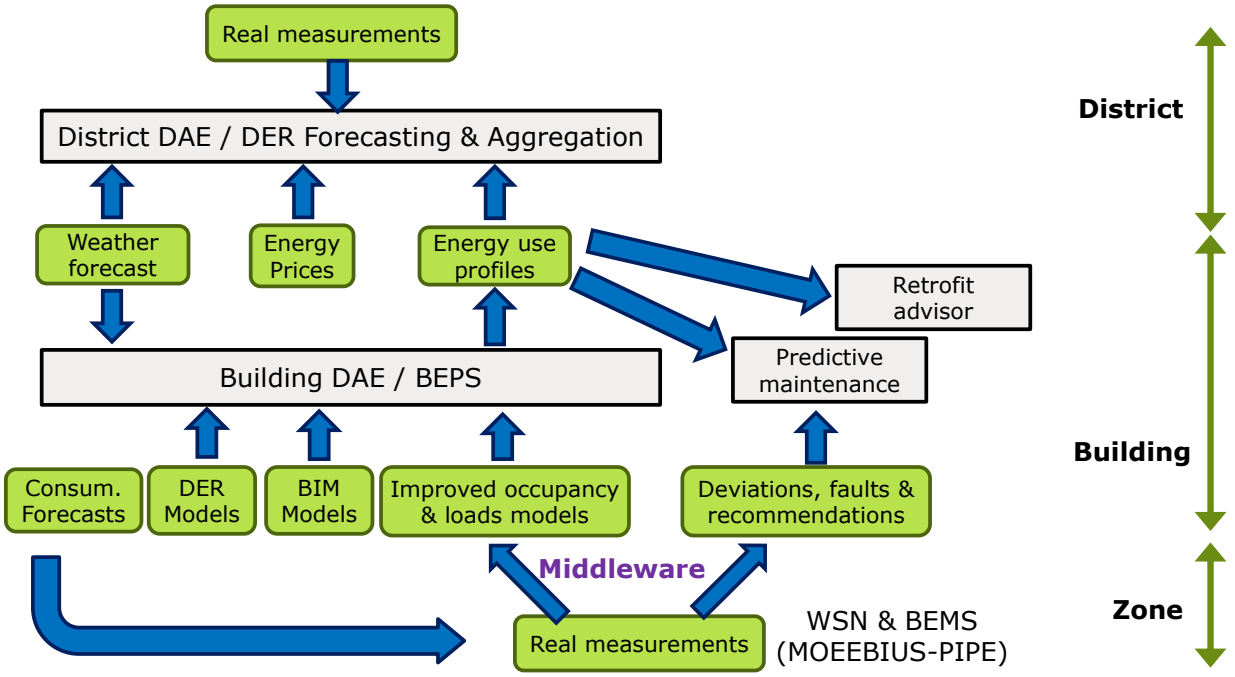

Figure 4 Conceptual data flow and components

In this regard, following the methodological framework for the definition of the MOEEBIUS architectural elements presented in D2.4, the following holistic business scenario was derived: **Business Scenario 5 – DSS towards the establishment of a sustainable building level and district level environment** (see Figure 5). The goal of this business scenario is to leverage a holistic and dynamic modelling and simulation / optimization approach that enables:

1. Improved predictions on the basis of more accurate and dynamically updated Building Energy Performance Simulation (BEPS) models
2. Real-time building and district performance optimization through control and predictive maintenance
3. Optimized retrofitting decision making on the basis of improved performance predictions

This business scenario is considered the integrated scenario and combines the different MOEEBIUS component functionalities to deliver the core business capabilities identified by the end-users during the requirement elicitation.

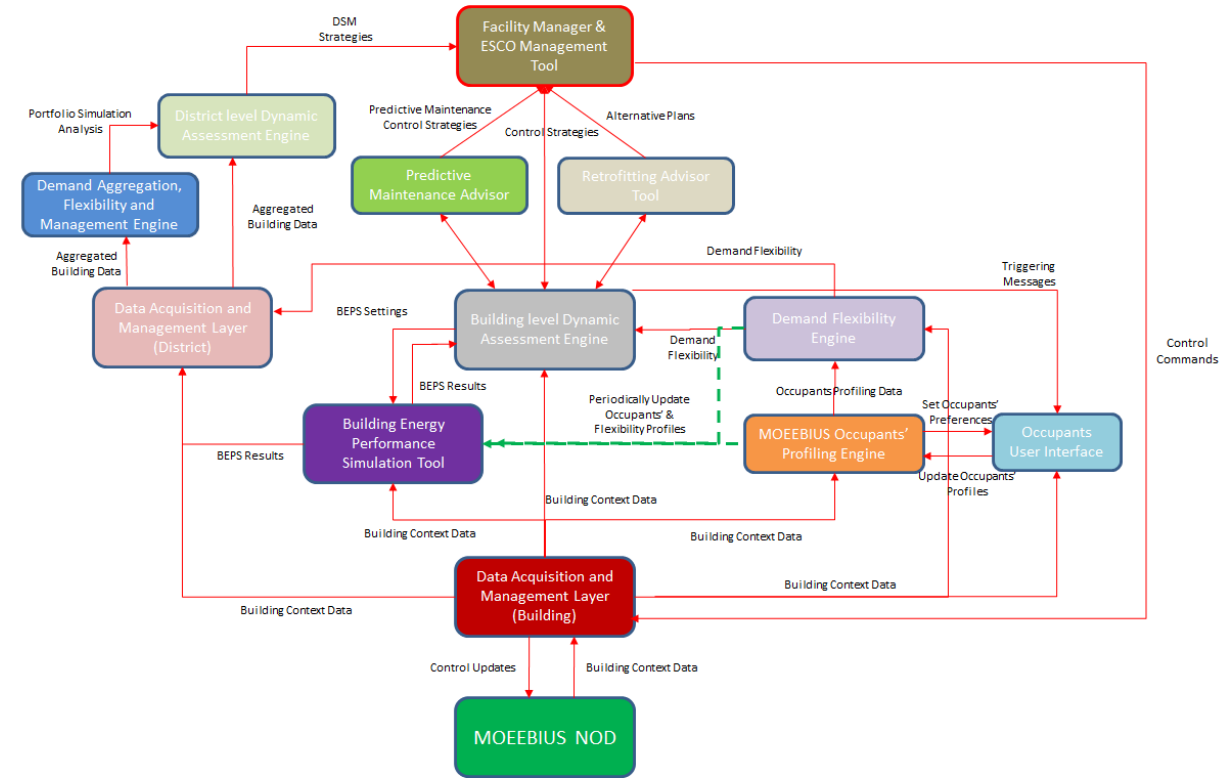

Figure 5 Conceptual holistic MOEEBIUS Business Case

According to Figure 5, the different data outputs of the application layer components are inputs to be reported to the **MOEEBIUS DSS business component**, i.e. the **Facility Manager & ESCO Management Tool** (refer to Figure 3 in D2.4 that shows the dynamic view of the components – business chain analysis).

- The **District Level Dynamic Assessment Engine** provides the DSS with DSM control strategies derived from the simulations delivered from the Demand Aggregation, Flexibility and Management Engine to support Demand Response scenarios and peak load management based on real life signals. These signals are to be received as aggregated data by the Data Acquisition and Management Layer (District) component. The common data model elements that will support this component are:
  - CityGML standards to aggregate building information and model DER elements
  - Energy price models
  - DSM models
- The **Building level Dynamic Assessment Engine – BDAE** (through the innovative Distributed Fuzzy Model Predictive Control capability) provides the list of alternative control strategies related to building operation to bridge the gap between simulated vs. monitored data. If the deviation is not solved, the Predictive Maintenance Advisor component o and / or the Retrofitting Advisor Tool enter into play. The common data model elements that will support this component are the next ones:

  - Occupancy behavioural profiles to model accurate simulation schedules
  - Dynamic DER models comprising Demand flexibility and elasticity profiles
  - Sensor related data models leveraged by the Data Acquisition and Management Layer
- The **Predictive Maintenance Advisor** component delivers predictive maintenance control strategies to the DSS to ensure the prompt operation of building systems by means of identifying a deteriorated performance of HVAC equipment. This component interfaces with the Building Dynamic Assessment Engine Fault Detection and Diagnosis capability to obtain identification of deviations of some metrics through real measurements vs. Building Energy Performance Simulation Tool (BEPS) analysis. This component will use the common data model elements supported by the BEPS and the Building level Dynamic Assessment Engine
- The **Retrofitting Advisor Tool** delivers the best fitted retrofitting projects to ensure the prompt building operation at macro level. This component interfaces with the Building Dynamic Assessment Engine Fault Detection and Diagnosis capability to identify sub-optimal performance and with the BEPS to retrieve different alternatives about building performance. Like in the Predictive Maintenance advisor, this component will leverage the common data model elements supported by the BEPS and the Building level Dynamic Assessment Engine
- The **Building Energy Performance Simulation Tool** provides enhanced simulation on the basis of more accurate MOEEBIUS models, such as weather forecasting profiles, occupancy behavioural profiles and flexible DER models as inputs to the Building Level Dynamic Assessment Engine and to the Retrofitting Advisor Tool. Specifically, this component will leverage the data model schemas defined in the next sections:
  - IFC4 and gbXML standards to model building information and short term weather forecasts related to building operation
  - Occupancy behavioural profiles to model accurate simulation schedules
  - Dynamic DER models comprising Demand flexibility and elasticity profiles

This business analysis highlights the role of the individual MOEEBIUS components as part of the MOEEBIUS integrated framework. The workflow analysis shows the high level dynamic view of MOEEBIUS platform, defining the core interactions among system components. This analysis and interfaces definition facilitate the definition of a data management / exchange framework that will enable the semantic integration of the different MOEEBIUS components by means of a Common Information Model. Thus, the data management framework will be a part of the Data Acquisition and Management Layer middleware that will provide semantically enhanced interfaces available to the rest of MOEEBIUS components.

4 Protocols and data models

Considering interoperability, scalability and flexibility of the MOEEBIUS framework, the Internet of Things paradigm was followed while analysing and evaluating the suitability of main standard-based **communication protocols** and **data modelling** approaches. In this section, the protocols that are going to be used for communication and the modelling language which is going to be employ for architecture design are discussed. In terms of communication with web services SOAP, WSDL, XML, REST, and JSON are going to be used. For messaging the platform uses AMQP and MQTT protocols. And finally, UML modelling language is going to be employed for modelling of the architecture design. In the following sections we will discuss each of these protocols and modelling languages in detail.

4.1 Communication protocols

4.1.1 Web Services Description Language (WSDL) 1.1

According to W3C, WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate, however, the only bindings described in this document describe how to use WSDL in conjunction with SOAP 1.1, HTTP GET/POST, and MIME.

4.1.2 Simple Object Access Protocol (SOAP)

According to W3C, SOAP Version 1.2 (SOAP) is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation specific semantics.

4.1.3 Extensible Markup Language (XML)

Based on W3C the Extensible Markup Language (XML) is a subset of SGML that is completely described in this document. Its goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML.

4.1.4 Representational state transfer (REST)

RESTful web services are one way of providing interoperability between computer systems on the internet. REST-compliant web services allow requesting systems to access and manipulate textual representations of web resources using a uniform and predefined set of stateless operations. In a REST web service, requests made to a resource's URI will elicit a response that may be in XML, HTML, JSON or some other defined format. The response may confirm that some alteration has been made to the stored resource, and it may provide hypertext links to other related resources or collections of resources. Using HTTP, as is most common, the kind of operations available include those predefined by the HTTP verbs GET, POST, PUT, DELETE and so on. By making use of a stateless protocol and standard operations REST systems aim for fast performance, reliability, and the ability to grow, by using reusable components that can be managed and updated without affecting the system as a whole, even while it is running. [2]

4.1.5 JavaScript Object Notion (JSON)

Based on W3C JSON is commonly used as an exchange format between Web client and backend services. Enabling HTML forms to submit JSON directly simplifies implementation as it enables backend services to operate by accepting a single input format that is what's more able to encode richer structure than other form encodings (where structure has traditional had to be emulated).

4.1.6 Message Queue Telemetry Transport (MQTT)

Based on mqtt.org the founders of MQTT protocol, MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium.

4.1.7 The Advanced Message Queuing Protocol (AMQP)

Based on amqp.org the founders of AMQP, The Advanced Message Queuing Protocol (AMQP) are an open standard for passing business messages between applications or organizations. It connects systems, feeds business processes with the information they need and reliably transmits onward the instructions that achieve their goals.

5 MOEEBIUS Components

The common information model developed inside the MOEEBIUS project compiles from the referenced information models the relevant aspects to create a common information model compatible with other already existing building and district level design or management tools. The next sections analyse the different available information models and standards and describes how these models are applied in MOEEBIUS.

5.1 Building Energy Performance Simulation Tool

Several options are available that leverage models for building information management (BIM) and energy efficiency. While these specifications are not only concerned with building energy efficiency, they partially or fully define the data models that can be used to capture this information. Among them, three consolidated specifications are the most promising ones.

The common data model definition takes as reference already existing city level and building level data modelling information models, such as gbXML, CityGBML, or IFC.

5.1.1 Industry Foundation Classes (IFC)

The general concept of Building Information Modelling (BIM) is commonly used to denote the process of creation, sharing, use and re-use of digital information about a building or built asset throughout its entire lifecycle, from design through procurement, construction and beyond, i.e. into its operation and management. The introduction of the BIM concept into the building construction industry is led by buildingSMART International® (bSI)[1], a non-profit organization which promotes the digital transformation of the built environment through creation and adoption of open, international standards.

Among the different standards included under the BIM umbrella for representing products, processes, dictionary data and collaboration, the most popular one is the IFC (Industry Foundation Classes) data schema, accepted as ISO 16739. It is the leading standard for facilitating data exchange of a building or asset between different software tools during the whole life-cycle using vendor-neutral format, i.e. not controlled by any software vendor. There have been several evolutions of the model, being the current one the IFC4[2] model.

There are two possible well established implementations for the IFC format: SPF (STEP[3] Physical File) or XML that are in the industry. Although XML is increasingly used, STEP is by large the most widely adopted format by software developers. It

[1] http://buildingsmart.org/standards/technical-vision/
[2] http://www.buildingsmart-tech.org/specifications/ifc-releases/ifc4-release/buildingSMART_IFC4_Whatisnew.pdf
[3] https://en.wikipedia.org/wiki/ISO_10303-21

is also more compressed than XML, which is an advantage for big model exchanges.

IFC is structured in four main layers presented in the following figure (Figure 6).

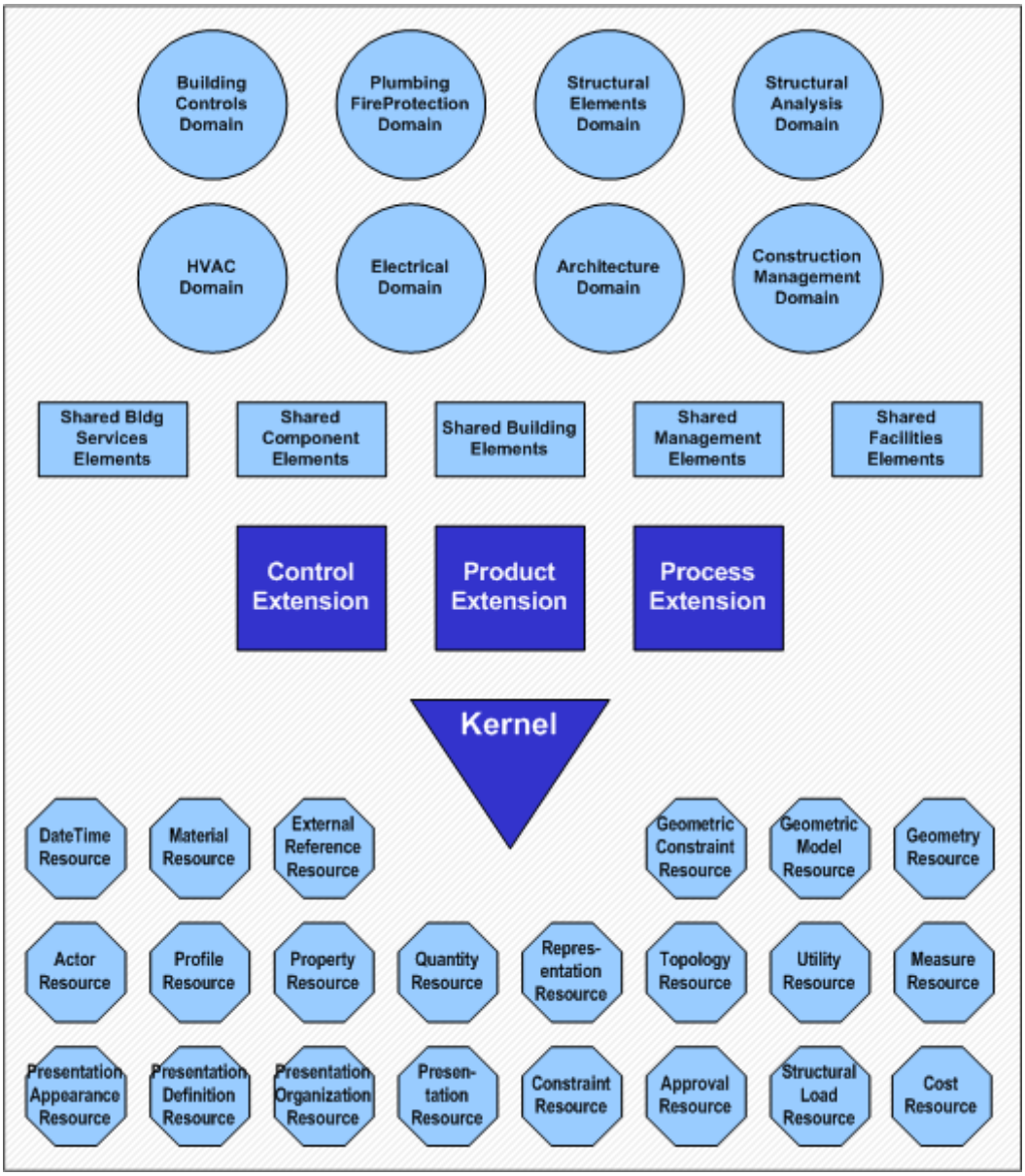

Figure 6 Structure of the IFC model

The Resource Layer is the lowest one in the IFC model architecture. As a base, resources are independent of any other classes in the model. The resource classes compile basic attributes such as geometry and any measurements, quantity, type of material, duration of construction or costs.

The Core Layer provides the kernel and a number of extensions on the core layer and contains entities for products, relationships and processes.

The Interoperability Layer defines objects that are common across AEC interdisciplinary applications within a set of modules or entity categories.

The Domain Layer, also known as Application Layer is the highest one in the IFC Object Model and it provides modules for AEC specific applications targeting concrete domains.

5.1.2 Green Building XML schema (gbXML)

The Green Building XML schema, or "gbXML", was developed to facilitate the transfer of building information stored in CAD-based building information models, enabling interoperability between disparate building design and engineering analysis software tools aiming at helping architects, engineers, and energy modelers to design more energy efficient buildings[4]. Today, gbXML has the industry support and wide adoption by leading Building Information Modeling (BIM) vendors including Autodesk, Trimble, Graphisoft, and Bentley. With the development of export and import capabilities in over 40 engineering and analysis modeling tools, gbXML has become a defacto industry standard schema. Its use dramatically streamlines the transfer of building information to and from architectural and engineering models, eliminating the need for time consuming plan take-offs. This removes a significant cost barrier to designing sustainable and energy efficient buildings. It enables building design teams to truly collaborate and realize the potential benefits of Building Information Modeling.

In June of 2000, the gbXML schema was submitted by a company called Green Building Studio for inclusion in aecXML(TM), the industry-led initiative launched by Bentley Systems. Shortly thereafter, gbXML became the draft schema for the Building Performance & Analysis Working Group.

In 2009, gbXML was spun off from Green Building Studio to become a stand-alone entity and things took off then: funding was secured, the schema was drastically improved, a new website was launched, and a community of thousands of architects, engineers, and energy modelers attended live webinars explaining the benefits of gbXML. Today, gbXML is funded by organizations such as the U.S. Department of Energy, the National Renewable Energy Lab (NREL), Autodesk, Bentley Systems, and others.

With the development of export and import capabilities in several major engineering modeling tools, gbXML has become a defacto industry standard schema. gbXML allows intelligent solutions for the design, certification, operation, maintenance, and recycling of buildings.

5.1.3 Home Performance XML (HPXML[5])

In addition to gbXML, a second consolidated specification is **Home Performance XML (HPXML)**[6], created by the Building Performance Institute (http://www.bpi.org/). The specification includes two proposed standards:

[4] http://www.gbxml.org/About_GreenBuildingXML_gbXML
[5] http://www.hpxmlonline.com/

- BPI-2100-S-2013[7], the Standard for Home Performance-Related Data Transfer (informally known as Home Performance XML, or the HPXML standard) provides requirements for an extensible mark-up language (XML) standard data transfer protocol that can be used to transfer home performance-related data between any party involved in a home performance program, including contractors, program administrators, utilities, and government agencies
- BPI-2200-S-2013[8], the Standard for Home Performance-Related Data Collection, designed to facilitate the exchange of information and data among all actors in the home performance industry by providing a standard vocabulary for describing terms related to buildings, energy consumption, and energy conservation measures. Each of the data elements defined in BPI-2200 can be transferred via HPXML.

5.1.4 BIM model implementation in MOEEBIUS

IFC is the current leading standard for BIM, since it is supported by a large number of software applications, including the most popular and adopted BIM authoring tools from main vendors (Autodesk, Nemetschek/AllPlan, Bentley, Trimble…) and for all project phases, especially for design and construction.

Since some of the activities of MOEEBIUS demonstrators are related to improving building energy performance assessment on the basis of enhanced BEPS models that include retrofitting measures and accurate weather and occupancy profiles, the use of BIM seems essential, and IFC and gbXML are the candidate formats to address the interoperability aspects between the users and tools of the MOEEBIUS platform. The current version of IFC is the IFC4 version, which is already supported by many tools and seems the most appropriate one to be used. Future extensions (IFC5/IFCAlignment) are very promising from the building and city integration perspective, but will not be available during the project evolution, although should be monitored for further evolutions of MOEEBIUS platform.

Consequently, in order to deliver a MOEEBIUS BEPS model, the **IFC4 standard** will be leveraged. Some of the elements comprising that standard will be translated into a database schema that will support the data / information exchange between MOEEBIUS components. approach based on some elements supported by the IFC standard. This schema will support the generation of enhanced simulation models that allow for more accurate representation of the real-life complexities of the building. In turn, these accurate models will leverage improved Building Energy Performance Simulation (BEPS).

---

[6] https://hpxml.nrel.gov/

[7] http://www.hpxmlonline.com/uploads/HPXML_BPI-2100-S-2013-Standard-for-Home-Performance-Related-Data-Transfer_20131115.pdf

[8] http://www.hpxmlonline.com/uploads/HPXML_BPI-2200-S-2013-Standard-for-Home-Performance-Related-Data-Collection_20131115.pdf

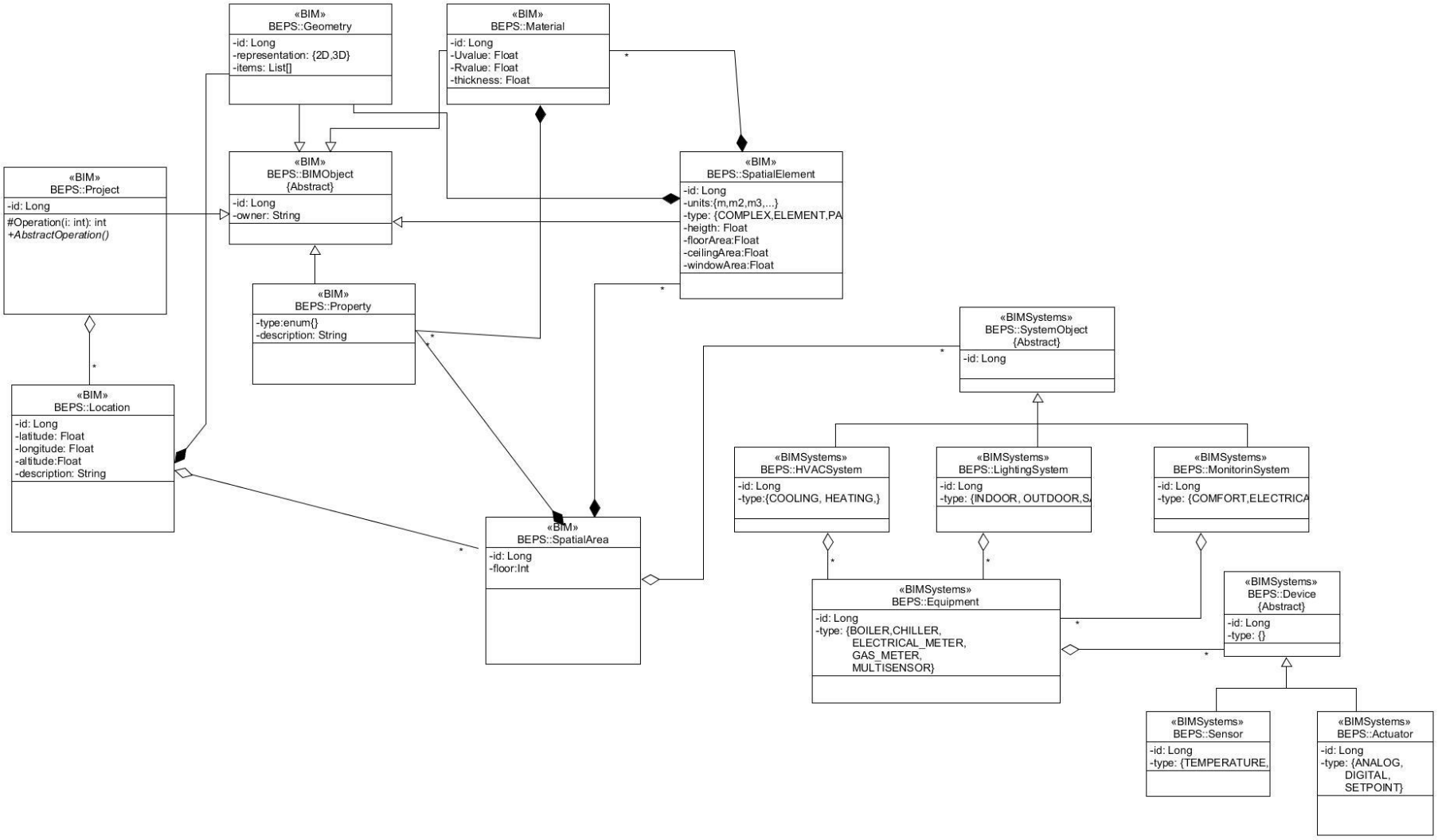

Figure 7 BIM main entities hierarchy

On the other hand, some of the **gbXML** elements and EPW IDD[9] (EnergyPlus Input Data Dictionary) can be leveraged to support the modeling of weather conditions and forecasts. These elements can be incorporated to the CIM to support the use of updated weather models by the BEPS module. Thus, extending the gbXML v5.10 schema with the fields from EPW IDD, an entity hierarchy was created, as shown in Figure 8. shows that the schema provides placeholders to store, for each location, the parameters related to the WEATHER_DATA section of an EPW file or data related to the GROUND TEMPERATURE section

[9] http://bigladdersoftware.com/epx/docs/8-3/auxiliary-programs/energyplus-weather-file-epw-data-dictionary.html

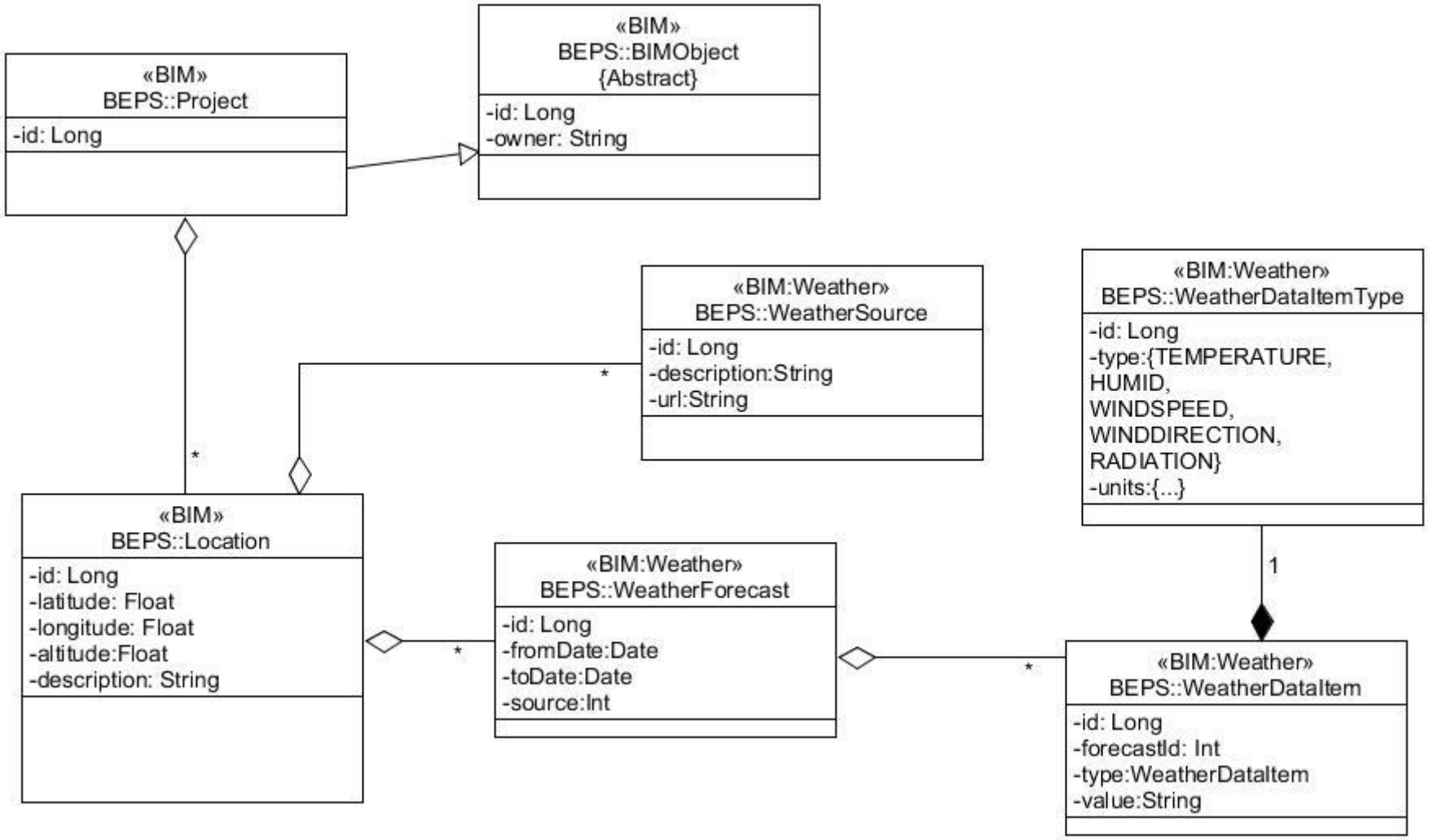

Figure 8 UML schema for supporting weather forecast model

The simplified building and environmental model introduced before main entities are described below.

- **Project**: High level concept that links all the database entities related to a single management domain. The concept enables to duplicate data base entries with different meaning (e.g. summer time in northern hemisphere versus southern hemisphere)
- **Location**: Conceptual aggregation of buildings that share same environmental conditions, management policies or geographical area.
- **SpatialElement**: Minimun spatial element in building distribution. It is important to mention that the building itself is a SpatialElement.
- **SpatialArea**:Aggregation of spatial elements from the functional or management point of view (e.g SpatialElements with same temperature set point)
- **SystemObject**: Abstract entity for implementation of HVAC, Lighting or Monitoring systems in buildings. Openings, such doors or windows would be modelled as SystemObjects tool
- **Device**: Abstract entity, base class, for Sensor and Actuators instantiation. This entity would include common features for Sensors and Actuators
- **Sensor/Actuator**: Straight forward entity for modelling sensors and actuators.
- **Equipment**: Functional aggregation of sensors and actuators (e.g. thermostat are equipment that include sensor and actuators)
- **OutdoorObject**: Abstract entity for instantiation of weather forecast, ground temperature, energy prices, or third party.
- **WeatherForecast:** Most relevant outdoor object. It handles the weather data modelling.
- **Calendar**: Entity for user that link comfort models (Scenes) to SpatialAreas for each day or period of day.
- **Scenes**: Reference entity for comfort models (temperature, relative humidity, expected occupancy)
- **Geometry**: Polynomial description of spatial elements that enables energy model simulations.
- **Material**: Construction details for components of spatial elements as the geometry are key entities for energy performance simulations

The entities listed above describe the data model entities from the data base implementation view. In the next chapters, the conceptual approach is going to be used. In this context for example ***SpatialElements*** will appear as ***BuildingZone*** or ***Building*** depending of the application domain, this means that SpatialElements are the generalization/abstraction of the BuildingZone or Building functional entities. The same approach is followed for ***Scence*** as abstraction for ***OccupancyProfile*** or ***ComfortConditions*** functional entities.

5.2 Occupants Profiling Engine

Following the definition of the MOEEBIUS BIM that defines the skeleton of building model, we proceed with the modeling of dynamic building elements as incorporated in the MOEEBIUS framework. The goal of this section is to define the data structures associated with the functionalities of MOEEBIUS Occupants' Profiling Engine. As presented in D2.4 & D3.1 there are two subcomponents that consist of the Occupants' Profiling Engine.

Starting with MOEEBIUS Occupancy profiling engine, the role of this component is to retrieve raw sensor data from different end sources and to provide:

- Real time occupancy data following the process in raw streams of sensor data
- Accurate occupancy profiling model data to be further incorporated in BEPS tool

For the 1st data type we are adopting the generic **sensor event model** and thus the focus is on the definition of the static occupancy profiling data structure.

As a root class of the model, we are considering the spatial elements (building zones) defined in BIM model. Then, the occupancy profiling class is associated with the building zone to model the respective occupancy patterns. The class diagram for the model is presented in the following figure.

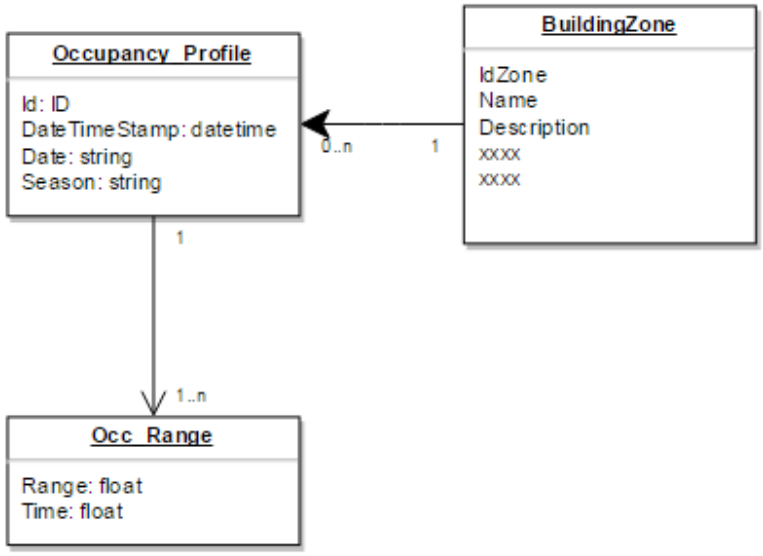

Figure 9 Occupancy Profiling Class diagram

The different data attributes of the model are further specified:

- **ID:** A unique ID for the specific instance of occupancy profile
- **dateTimeStamp**: Date and time stamp for the extracted occupancy profiling model

- **date**: Defines the type of day refer to. The enumerated values of this parameter will be updated according to the extracted occupancy patterns. Typical values are: weekend, weekday, special events, bank holidays
- **season**: Defines the season that metrics refer to. The typical taxonomy is: Autumn, Winter, Spring, Summer
- **range**: The range of occupants in a building zone per a specified time frame (e.g. per hour).
- **time**: The associated time frame.

The details (algorithmic framework and internal data modelling) for the extraction of typical occupancy profiles are presented in D3.4 "Occupants Comfort modelling".

Along with typical occupancy patterns, the goal of this section is to extract behavioural profiles, understanding comfort (dis)satisfaction boundaries, while considering also health aspects in the profiling process. The extraction of the typical behavioural profiles (thermal/ visual) is delivered per building zone and the association with BIM spatial elements is defined, as presented in the following class diagram:

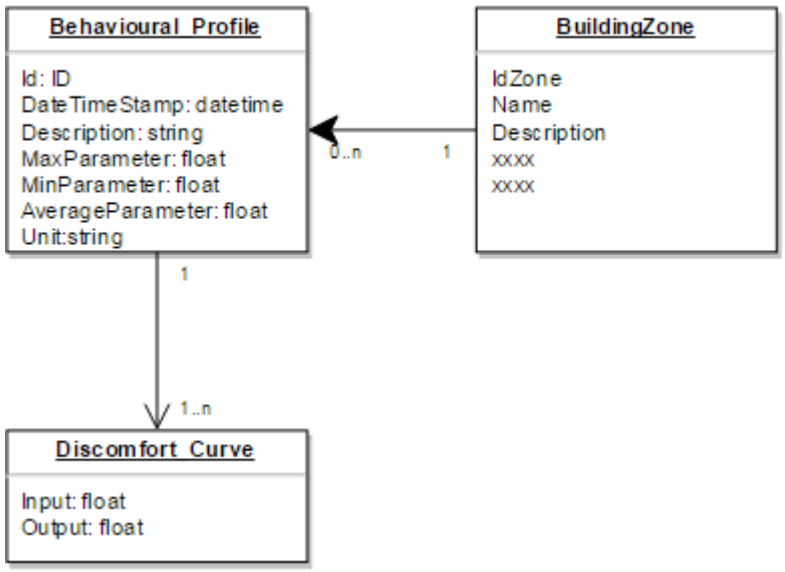

Figure 10 Behavioral Profiling Class diagram

The data attributes of the model are further specified:

- **ID**: A unique ID for the behavioural profile extracted
- **dateTimeStamp**: Date and time stamp for the extracted behavioural profiling model
- **Description**: The type of behavioural profile defined: thermal/visual
- **MaxParameter:** The parameter/value for the upper boundaries of the comfort type examined.

- **MinParameter:** The parameter/value for the lower boundaries of the comfort type examined.
- **Average Parameter:** The parameter/value for the average value of the comfort type examined
- **Unit:** The unit type for the parameter/value of comfort type examined
- **DiscomfortCurve:** The non-parametric vector that defines the discomfort utility function as a function of input environmental conditions
    - **Input:** The input parameter/value for the comfort type examined
    - **Output:** The output parameter/value for the comfort type examined, expressed in terms of utility function

Further to the extraction of zone level behavioural profiles, we further specify typical operational profiles for device types. In that case, the association with device class is provided as the root class for the model:

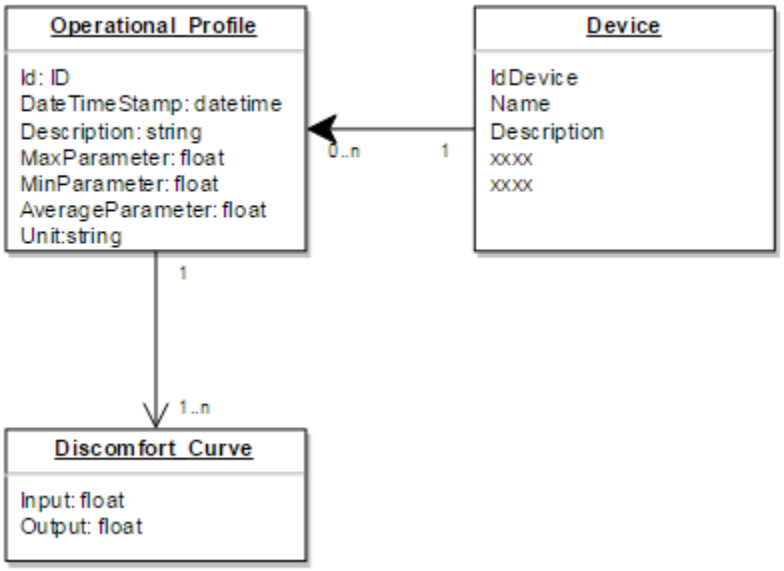

Figure 11 Operational Profiling Class diagram

The details (algorithmic framework and internal data modelling) for the extraction of typical behavioural profiles are presented in D3.4 ”Occupants Comfort modelling”.

5.3 Demand Aggregation, Flexibility and Management Engine

Following the definition of MOEEBIUS CIM associated with behavioural profiling engine, we further proceed with the definition of data models associated with MOEEBIUS **Demand Flexibility** framework. The demand flexibility model involves flexibility, aggregation and management engines. The demand flexibility model enables:

- To provide semantically enhanced DER profiles by taking into account DER operational and contextual data. The detailed DER modelling that will further enable the definition of DER parameters is provided in D3.3
- To deliver Context-Aware flexibility Profiles, reflecting real-time demand flexibility as a function of multiple parameters, such as time, device operational characteristics, environmental context/conditions, energy costs, occupant comfort preferences and health/ hygienic constraints.
- To deliver Context-Aware aggregation profiles, reflecting energy demand synergies among buildings with complementary usage patterns.

Therefore, we are introducing as part of the MOEEBIUS CIM the term of demand flexibility. This new concept is associated with each controllable device type (part of the BIM), reporting the demand flexibility potential of each device. The next figure provides the extended view of BIM model to further incorporate "demand flexibility" data structure:

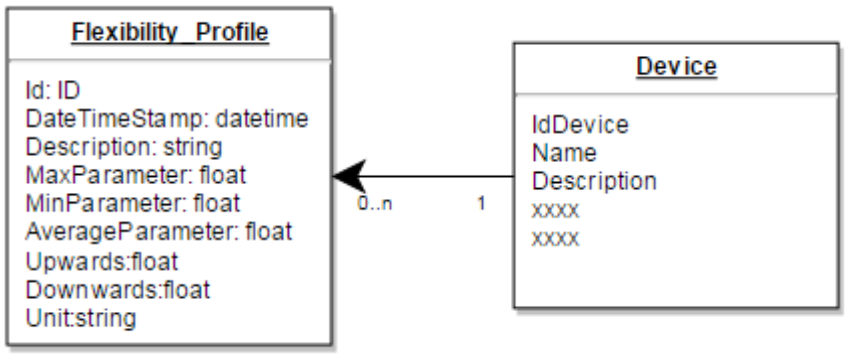

Figure 12 Demand Flexibility Profiling Class diagram

Where:

- **Id:** A unique ID for the flexibility profile model defined
- **DateTime**: the date and time for the extracted model parameters
- **Description:** A short decrption about the extracted flex profile
- **MaxParameter:** The parameter/value for the upper boundaries of the demand flex offered by this specific device.
- **MinParameter:** The parameter/value for the lower boundaries of the demand flex offered by this specific device.

- **Average Parameter:** The parameter/value for the average demand flex value of the device examined
- **Upwards:** The current value for the upwards demand flex value offered by this specific device.
- **Downwards:** The current value for the downwards demand flex value offered by this specific device.
- **Unit:** The unit type for the parameter/value of demand flex

We proceed with an abstract modelling representation of demand flexibility as a new entity addressed in the MOEEBIUS to facilitate the implementation of the different business scenarios examined. We have to point out that "demand flexibility" is an additional metric for each controllable device type and thus normally is handled as an event data type:

- Time: timestamp of metric value
- Measurement ID: A unique ID that expresses "demand flexibility" value
- Value: The actual value about demand flexibility
- Unit: The unit of the value (Wh)
- Reliability: The accuracy of the metric value retrieved

Finally, and in lack of low level information from building environment, high level price elasticity profiles are defined. These profiles express demand flexibility as a function of energy prices and the respective model is presented:

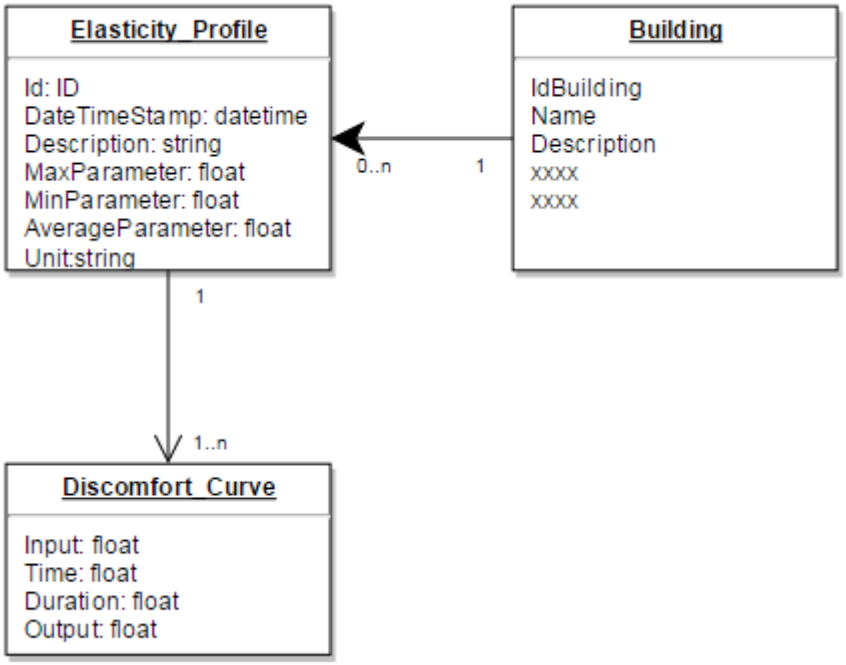

Figure 13 Demand Elasticity Profiling Class diagram

Where:

- **Id:** A unique ID for the elasticity profile model
- **DateTime**: the date and time for the extracted model parameters
- **Description:** A short decrption about the extracted elasticity profile
- **MaxParameter:** The parameter/value for the upper boundaries of the elasticity term
- **MinParameter:** The parameter/value for the lower boundaries of the elasticity term
- **Average Parameter:** The parameter/value for the average demand elasticity
- **Unit:** The unit type for the parameter/value of demand elasticity (percentage)
- **Input:** The price value as the input parameter for elasticity
- **Time:** The date time associated with an elasticity value
- **Duration:** The total time duration associated with an elasticity value
- **Output:** The demand elasticity value associated with the input model parameters

The demand flexibility data (aggregated at building level or per device type) are further available to the different MOEEBIUS building and district level components for further exploitation.

5.4 District Level Dynamic Assessment Engine

District level information models represent georeferenced data of city areas. Most used and developed tools for representation of district information are 2D GIS systems. Current trends search for tools for representation of district areas through 3D models. A 3D district model consists of the information of digital elevation model, buildings, land use, vegetation and infrastructure, such as bridges and roads, among others. The main characteristic of these models is the ability to store all information of a district in a single data model, which facilitates its use and interoperability. These models can provide, operate and manage district related data, which can be used in different applications such as urban sustainability plans, disaster management, traffic planning, security, telecommunications, navigation, and tourism.

5.4.1 GIS Systems (SHP)

GIS (Geographic Information Systems) systems are the most mature and used systems for the representation of geographically referenced data used to solve problems of spatial planning and management in diverse fields as urban planning, geology, agriculture, management of incidents and natural disasters, sociology, transport, environment and many others.

The information is organized in layers of graphical entities, as displayed in Figure 14. Each of the layers has georeferenced information of one or more relevant attributes which represent a virtual model of the spatial reality (Earth). The following figure shows an example of some common data layers in GIS systems.

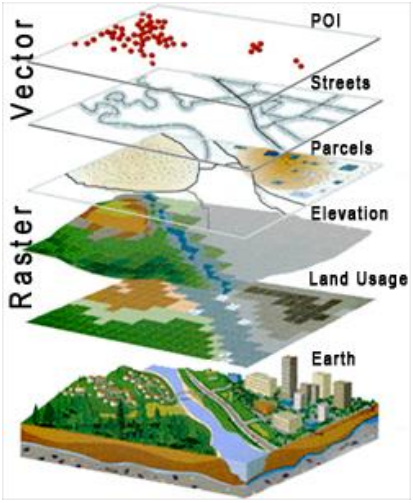

Figure 14 Data layers in a GIS system

The graphical representation of a layer can be of two types: Raster: Raster data type consists of rows and columns of cells, representing generally a rectangular grid, with each cell storing a single value. Examples of raster data are aerial imagery, orthophotos, scanned images (JPEG, GIF, TIFF ...). Vector: Geographical features are often expressed as vectors. Each of the geometries represented by a vector is linked to a row in a database that describes its attributes.

5.4.2 OpenStreetMap (OSM)

OpenStreetMap[10] is a collaborative project to create a free editable map of the world started in 2004. Main motivation for the development of OSM was the unavailability and restrictions in the usage of official map information. Information in OSM is divided into several thematic layers. There is a straight forward link between thematic areas in OSM and the other main data models (e.g. CityGML). Some of the main elements of the thematic areas are: buildings, transportation, land use, natural resources, etc. The main advantage of OSM is the availability of data that has not been yet collected in the official maps, it is open source and free of use. However, the quality of the information cannot be fully guaranteed, due to the voluntarily of the contributions and the inexistence of sophisticated quality control procedures.

5.4.3 3D Cities in Google (KML)

The best known example of 3D cities is Google. 3D city model represented in Google is a geometric model without semantic information associated with this geometry.

The file format used to represent 3D cities in Google is KML (Keyhole Markup Language). KML is an XML schema and file format for modelling and storing geographic features such as points, lines, images, polygons, and models that can

[10] https://www.openstreetmap.org/

be viewed in Google Earth, Google Maps and other applications. This file format is one of the most popular formats for geometric data exchange due to the adoption of Google Earth. It is based on COLLADA, which was defined with the purpose of making easy to transport 3D assets between applications.

KML is an official standard defined by the OGC. It may contain geographic vector data as well as raster data in a georeferenced data model allowing also the inclusion of high resolution textured 3D models.

5.4.4 CityGML

CityGML[11] is an open data model based on XML format for storage and exchange of virtual models of 3D city defined by the OGC[12]. The aim of the development of CityGML was to reach a common definition and understanding of the basic entities, attributes, and relations within a 3D city model (Kolbe T. H. 2009). What is especially important is that it allows the reuse of the same data in different fields of application. The extension of the 3D model with semantic information is the main feature of the CityGML data model. CityGML extends the geometric information with additional information on the use and function of objects through an ontology defined in this data model. The aim of this approach is to ensure the integrity and validity of the model city to be useful for administrative use. CityGML is based on GML (Geography Markup Language).

The main characteristics of CityGML Data Model are: *Multi-scale Modelling:* CityGML supports different levels of detail (LoD). The LoD are necessary to adjust the level of detail of the information to the requirements of each application. CityGML represents the same object with different LoD simultaneously (see Figure 15).

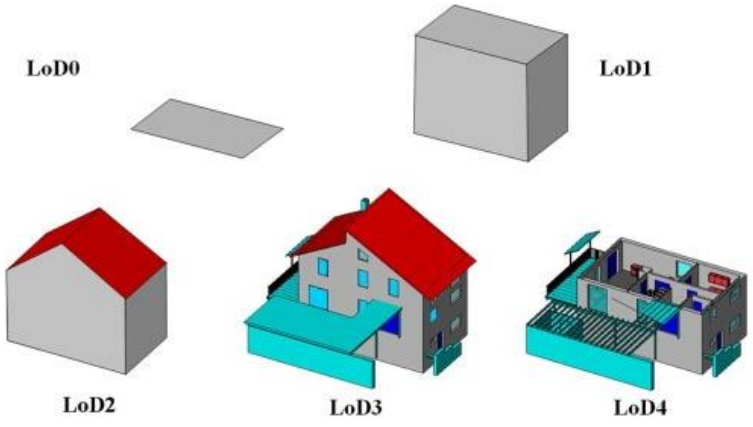

Figure 15 LoDs in CityGML (Kolbe et al. 2012)

Coherence between semantics and geometry: CityGML represents the graphical appearance of city models and also semantic and thematic properties, taxonomies

[11] http://www.opengeospatial.org/standards/citygml
[12] http://www.opengeospatial.org/

and aggregations. The following figure (Figure 16) shows the coherence between semantics and geometry in CityGML.

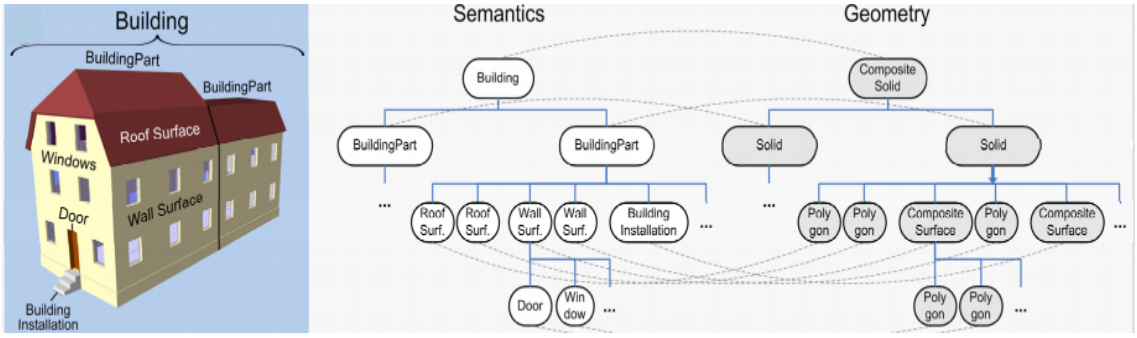

Figure 16 Coherence between semantics and geometry in CityGML (Kolbe et al. 2012)

Extensibility and Interoperabilitty: The most common and powerful way for the extension of the CityGML data model is through the Application Domain Extension (ADE): ADE extensions specify the additions to the CityGML data model required for a specific domain. Such additions include the introduction of new features to existing classes of CityGML. Regarding to the interoperability there are many tools that allow the exportation to CityGML. The following figure (Figure 17) shows already identified workflows for the generation of CityGML files.

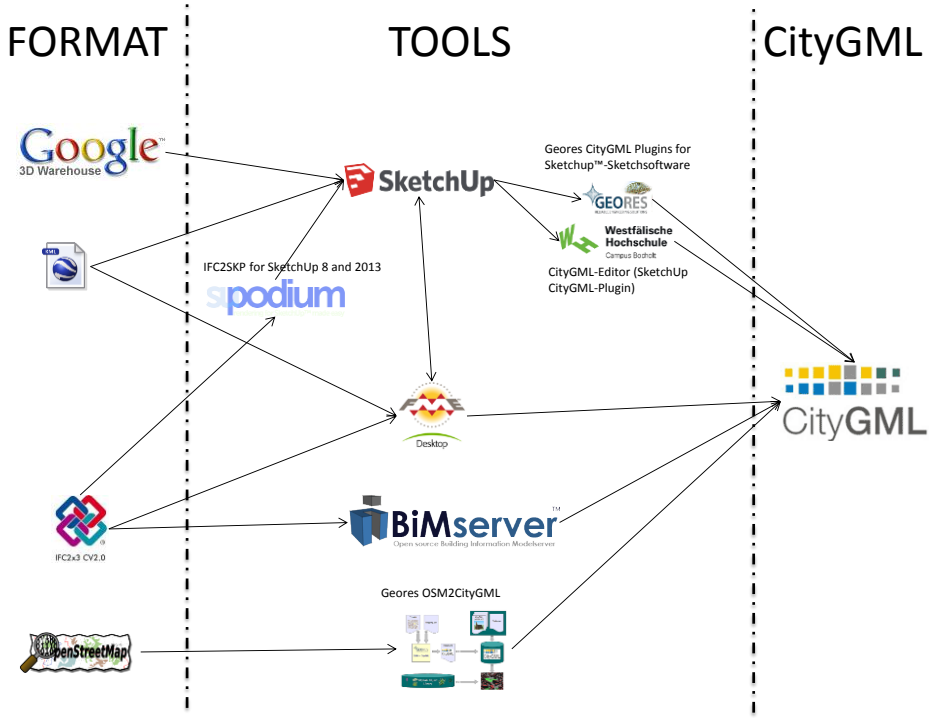

Figure 17 CityGML Interoperability with tools

5.4.5 District level implementation for MOEEBIUS

Regarding the implementation of a Common Information Model at district scale, the next data models can be leveraged to build a district model.

CityGML will be mainly considered and extensively applied in MOEEBIUS, since it is currently the most prominent standard data model for the representation of district areas in 3D. It will be used for modelling the geometry and some of the semantic properties of the static elements of the city model.

The delivery of the MOEEBIUS district management framework, special focus should be delivered on the definition of the common data model among the components that consist of the district side of the proposed framework. The root point for the definition of this model extension is the static map representation (Geography Markup) of the portfolio, considering that way the **CityGML protocol**. **CityGML** is a common information model for the representation of sets of 3D urban objects. It defines the classes and relations for the most relevant topographic objects in cities and regional models with respect to their geometrical, topological, semantical and appearance properties. In MOEEBIUS project, we select the elements required for map representation and we further extend the structure taking into account MOEEBIUS use cases and requirements.

The data classes that will be used from the CityGML standard will represent the distributed energy resources required by the district level MOEEBIUS components and will provide a relationship to the detailed building models based on the IFC standard. These classes will be based on the 3D City Database[13] (3DCityDB), which is a free Open Source package consisting of a database schema and a set of software tools to import, manage, analyse, visualize, and export virtual 3D city models according to the CityGML standard. The database schema results from a mapping of the object oriented data model of CityGML 2.0 to the relational structure of a spatially-enhanced relational database management system (SRDBMS). The 3DCityDB supports the commercial SRDBMS Oracle (with 'Spatial' or 'Locator' license options) and the Open Source SRDBMS PostGIS (which is an extension to the free RDBMS PostgreSQL). 3DCityDB makes use of the specific representation and processing capabilities of the SRDBMS regarding the spatial data elements. It can handle also very large models in multiple levels of details consisting of millions of 3D objects with hundreds of millions of geometries and texture images.

Figure 18 below shows the CityGML Core Thematic Model, where the different district resources can be defined. The base class of all the thematic classes is the _CityObject. The class attribute of the CityObjectGroup describes the classification of the distributed resources, e.g, road, track, buildings, energy generation facilities, district heating facilities, etc. The subclasses of _CityObject comprise the

[13] http://www.3dcitydb.net

different resources of a district model. Some resources that may not be included in the schema can be defined under a generic district object model (GenericCityObject). In order to create a Common Information Model for MOEEBIUS that merges the building and district perspective, the 3DCityDB provides a reference of a 3D object to its corresponding object in an external data-set. For instance, a building modelled under CityGML can be related to a building modelled in IFC format. This feature will allow the extraction of basic or detailed information of a resource from the Common Information Model depending on the information needs of the modules.

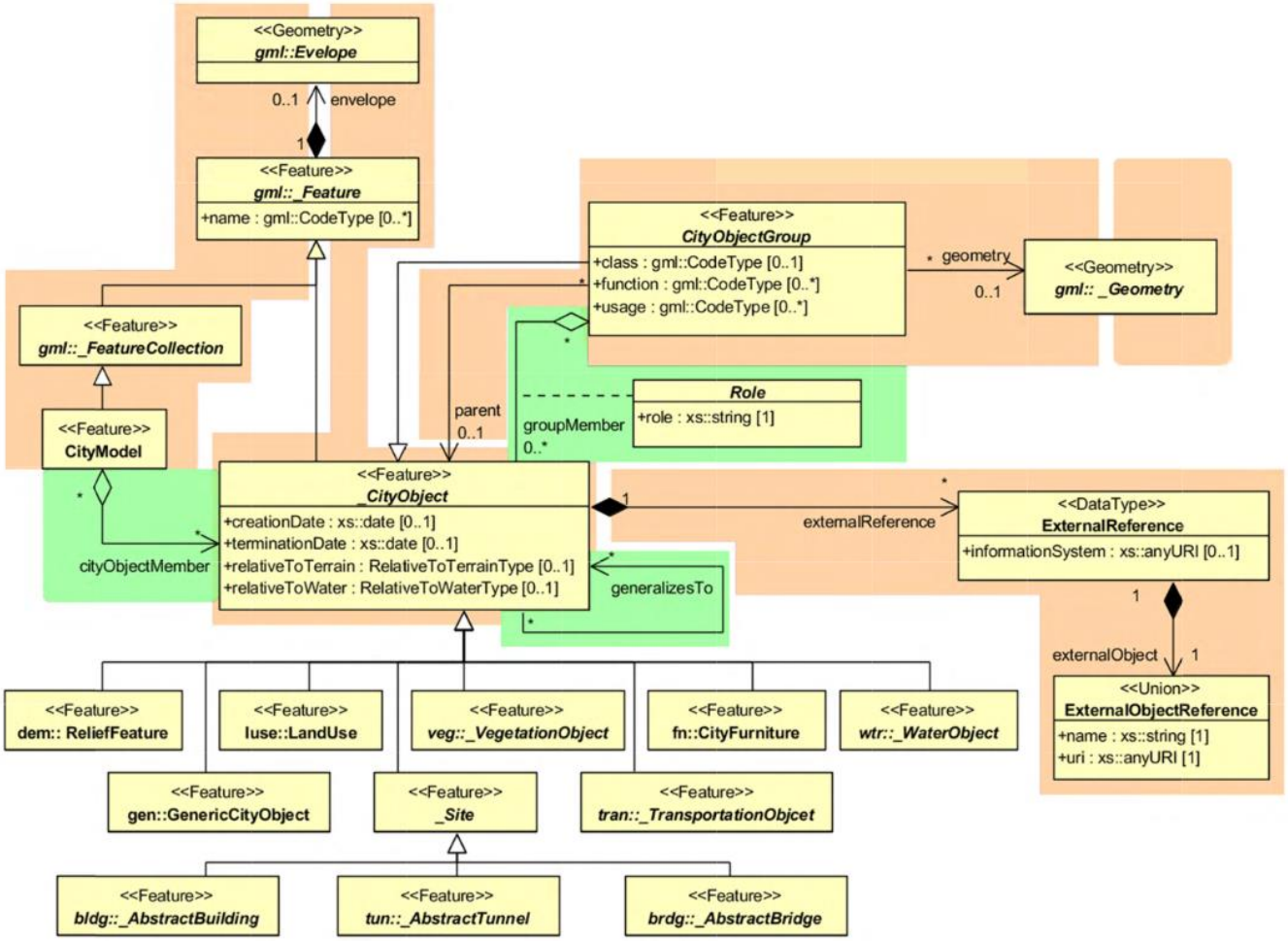

Figure 18 CityGML Core Model and Thematic classes

Following the selection of the static district level elements, we proceed by modelling the dynamic attributes required for the definition of the MOEEBIUS district management framework.

There are three main data concepts defined for the different functionalities offered by the MOEEBIUS district management framework:

- Management of Metric values, retrieved from building environment to support the implementation of ESCo/Aggregator business services
- Management of energy prices, required for the implementation of DSM strategies and portfolio management
- Implementation of DSM strategies which is the control functionality triggered by ESCo/Aggregator.

The rest of this section is focusing on the definition of these 3 main pillars that define the MOEEBIUS district level common information model.

5.5 Predictive Maintenance Tool

The predictive maintenance (PM) module provides capabilities for systematic monitoring of equipment, materials and HVAC system performance. Performance deviation can be identified by monitoring, tracking and projecting the values of various performance **indicators**. In MOEEBIUS framework the following two types of key performance indicators (KPI's) will be used:

- **Equipment degradation (ED) indicators** used for systematic comparison of the actual equipment performance with the predicted performance (of boiler heat exchangers, pumps, fans, and AHU and fan-coil filters).
- **Control performance monitoring (CPM) indicators** used for monitoring the quality of closed-loop control achieved by means of local (PID) controllers. The CPM indicators quantify the magnitude of typical control problems (oscillations, fluctuations, predictability, frequent ON-OFF switching) by analyzing the time series data of **process variables** (temperatures, pressures), **disturbances** (outdoor temperature, hot/chilled water temperature), **actuator commands** (e.g. valve position command, fan speed command, damper position digital command), **setpoint**s and equipment **statuses** (ON/OFF, multistate).

This approach was selected due to the fact that there are two main factors responsible for equipment performance degradation: (A) **physical degration** (due to fouling, scaling, dirt accumulation and other long-term deterioration processes); (B) **degradation caused by improper control** (e.g. wrong control strategy may force an ON/OFF valve to repeatedly open and close, thus causing wear of actuators and dramatic reduction of operating lifetime of valves, dampers and other actuators).

The indicators can be further classified based on the HVAC **equipment type** (e.g. boiler KPI's, air handler KPI's, pump KPI's) as well as the overall **algorithmic approach** to calculate a given indicator (e.g. using a deterministic formula, using data-driven and machine learning approaches, using reference performance data provided by the HVAC manufacturer, etc). This KPI classification is creating an

additional level of complexity in specifying the **data requirements** of each indicator. To overcome this problem, each KPI will specify its data requirements by providing the following list of requirements:

- **Required and optional datapoints** (i.e. sensor data, commands, setpoints, equipment statuses, etc) from various levels of building hierarchy (e.g. diagnosed equipment itself, building, serviced zone, etc)
- **Required and optional static parameters** of diagnosed entities (equipment, building, zone, etc), such as nominal power, nominal volumetric flow, maximum capacity, maximum operating temperature, etc.
- **Referential operation curves** (commonly provided by HVAC manufacturers), such as boiler efficiency curve, fan characteristic curve, etc.

Note that **Building Controls Domain** and **HVAC Domain** are the key components from the Domain (Application) Layer of the IFC4 standard that will be extensively utilized by the predictive maintenance module. The PM module will utilize the following **entities** (from the simplified building and environmental model described in 5.1.4 BIM model implementation in MOEEBIUS):

- **Equipment**
- **Sensor**
- **Actuator** – particular important entity for the CPM KPI's
- **Calendar**

The **CPM-KPI's** are computed at every time instant. Their trend will be analyzed over time to suggest the predictive maintenance action to be taken in order to prevent wear out of a device and potential failure of the actuated mechanical component (modulating valve, ON/OFF valve, ON/OFF damper). Below are examples of **typical faults** detectable by monitoring the CPM-KPI's:

- Poor Control Performance due to Tuning
- Oscillating Process Variable and Command
- Oscillating Process Variable due to Oscillatory Setpoint
- Command Oscillating between Bounds

The **ED-KPI's** will be based on the actual measurement and/or manufacturer data (e.g. reference efficiency curves, performance curves measured during equipment commissioning) to evaluate the actual performance of the device (e.g. boiler running at the efficiency 5% lower than expected due to likely fouling, or filter clogging).

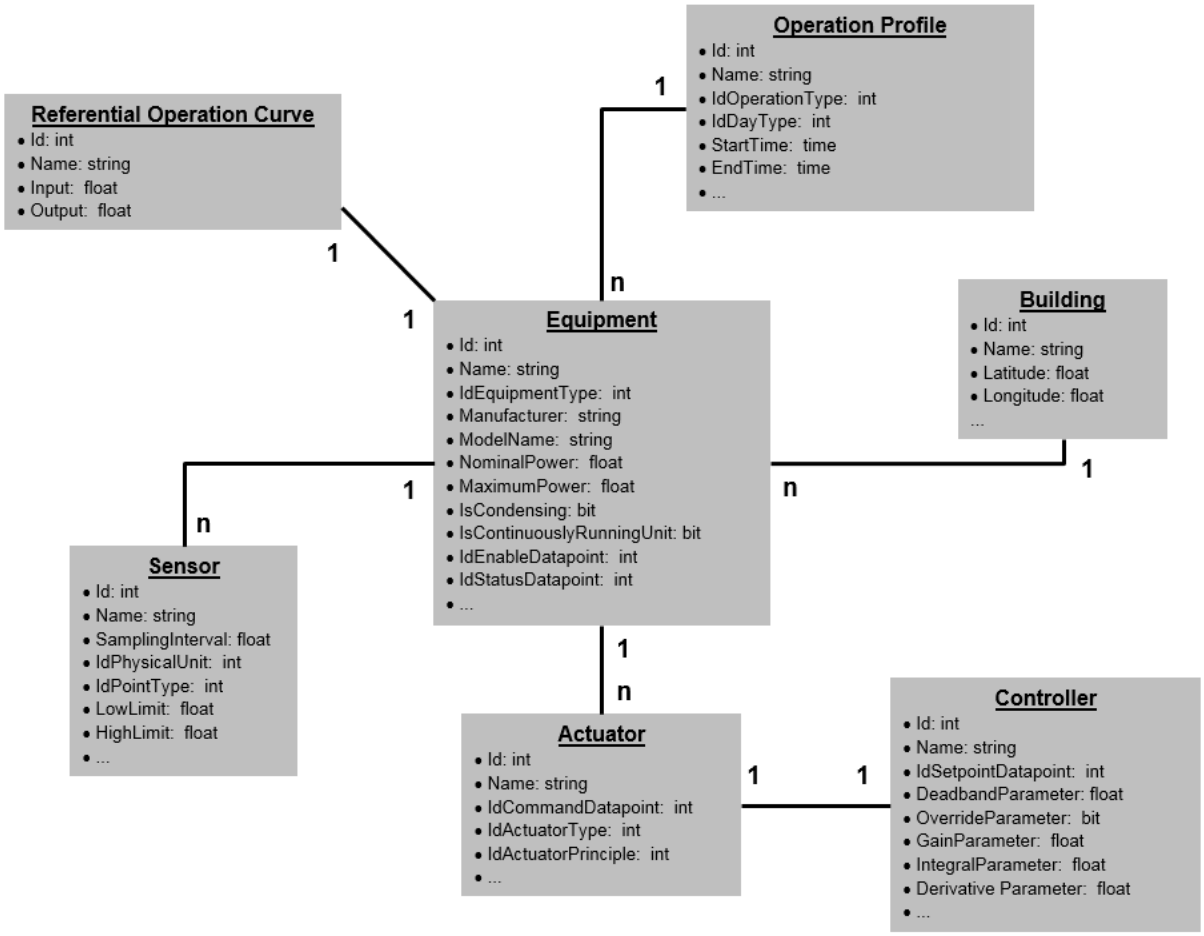

Figure 19 PMe class diagram

The following attributes are common to all entities:

- **Id**: A unique numeric identifier (of Building, Equipment, Sensor, etc)
- **Name**: A text description and/or label (of Building, Equipment, Sensor, etc)

In the folowing text, examples of speficic entity attributes will be specified; however, this list is primarily for informative purposes and is not meant as a complete list of attributes (additional attributes will be added later as needed).

Attributes of the **Building** entity:

- **Latitude**: Numeric representation of building latitude
- **Longitude**: Numeric representation of building longitude

Attributes of the **Equipment** entity:

- **IdEquipmentType**: Numeric identifier specifying the type of equipment (e.g. 1 = Air Handling Unit; 2 = Boiler)
- **Manufacturer**: String specifying the equipment manufacturing company (e.g. Trane, Uniconfort)
- **ModelName**: String specifying the equipment model (e.g. CFC-1500)

- **NominalPower**: Parameter specifying the equipment nominal power (if applicable)
- **MaximumPower**: Parameter specifying the equipment maximum power (if applicable)
- **IsCondensing**: Boolean parameter specifying whether the equipment is utilizing the condensing mode (if applicable)
- **IsContinuouslyRunningUnit**: Boolean parameter specifying whether the equipment is supposed to run continuously (if applicable).
- **IdEnableDatapoint**: Numeric identifier of a datapoint that represents the enable (clear-to-run) command for the equipment
- **IdStatusDatapoint**: Numeric identifier of a datapoint that represents the run status of the equipment

Attributes of the **Sensor** entity:

- **SamplingInterval**: Parameter specifying the sampling interval of measured sensor data in seconds (e.g. 900)
- **IdPhysicalUnit**: Numeric identifier specifying the physical units of the sensor (e.g. 1 = Degree Celsius; 2 = Percent; 3 = Pascal)
- **IdPointType**: Numeric identifier specifying the type of datapoint measured by the sensor (e.g. 1 = Boiler hot water supply temperature; 2 = Zone temperature; 3 = Outdoor temperature)
- **LowLimit**: Numeric parameter specifying the minimum value of the measured range
- **HighLimit**: Numeric parameter specifying the maximum value of the measured range

Attributes of the **Actuator** entity:

- **IdCommandDatapoint**: Numeric identifier specifying the datapoint representing the command signal to the actuator
- **IdActuatorType**: Numeric identifier specifying the actuator type (e.g. 1 = Modulating; 2 = On/Off; 3 = Off / Low / High)
- **IdActuatorPrinciple**: Numeric identifier specifyig the actuator physical principle (1 = Electric; 2 = Pneumatic; 3 = Hydraulic)

Attributes of the **Controller** entity:

- **IdSetpointDatapoint**: Numeric identifier specifying the setpoint (desired value) of the controlled variable
- **DeadbandParameter**: Numeric parameter specifying the value of the deadband parameter (e.g. 1 °C)
- **OverrideParameter**: Boolean parameter specifying whether the controller is in the override mode (true/false)
- **GainParameter**: Numeric parameter specifying the gain value of the local control loop

- **IntegralParameter**: Numeric parameter specifying the integral value of the local control loop
- **Derivative Parameter**: Numeric parameter specifying the derivative value of the local control loop

Attributes of the **Operation Profile** entity:

- **IdOperationType**: Numeric identifier specifying the operation type (e.g. 1 = Off; 2 = Running; 3 = Stand-by; etc).
- **IdDayType**: Numeric identifier specifying the day type (e.g. 1 = Monday; 2 = Tuesday)
- **StartTime**: Time stamp specifying the start of the operation profile (e.g. 9:30)
- **EndTime**: Time stamp specifying the end of the operation profile (e.g. 17:30)

Attributes of the **Referential Operation Curve** entity:

- **Input**: Numeric parameter specifying the input value of the input parameter (e.g. 70°C of return water temperature). Note that multiple input parameters are typically used (e.g. 70°C of return water temperature and 95% of firing rate in case of boilers).
- **Output**: Numeric parameter specifying the output value of the output parameter (e.g. 95% efficiency). Note that multiple outputs are sometimes also available.

The predictive maintenance KPI's will be available to different MOEEBIUS building components. Main integration will be with a Virtual Reality environment for Predictive Maintenance enabling the fusion of BIM information (3D BIM Mapping), equipment and components life-cycle data, discrepancies and anomalous behaviour detection and diagnosis algorithms along with mobile device sensors to offer a highly intuitive interface for maintenance managers.

The system integrating 3D virtual environment along with mobile device tracking for spatial and temporal contextualisation of equipment and components life-cycle data has been under development at Cork Institute of Technology with a number of successful pilot applications. In the scope of MOEEBIUS project the system will be further extended to integrate with the MOEEBIUS predictive maintenance tool allowing Maintenance Experts (or ESCOs/ Facility Managers) to identify weakly performing components, not by analysing a series of data and graphs in their control panels, but through the screen of their mobile device.

The virtual reality environment will reuse existing 3D rendering engine, which supports most of command 3D data formats such as XML based COLLADA (collaborative design activity, *.dae) and text based Autodesk Drawing Exchange Format (*.dxf). The spatial contextualisation of displayed information for Maintenance Experts will be enabled by cost effective technology based on

Bluetooth low energy battery powered RFID tags (known as IBeacons) compatible with most of widespread mobile devices and phones. The RFID will be placed at key location at a premise possible collocated with maintained equipment's and on main hallways to allow maintenance people navigation and displayed data contextualization. The visualization will show all the devices in the close surrounding with their status (e.g. replace filter).

5.6 Retrofitting Advisory Tool

The Retrofitting Advisory Tool allows the pre-assessment of alternate solutions for components in existing buildings. These alternates are first and foremost additions and substitutions regarding monitoring and control of infrastructure in these buildings. Thanks to the chosen methodology it is however easily possible to extend the portfolio with more involving options like partial substitutions of chosen materials, e.g. different window frames or doors.

5.6.1 Dependencies

The retrofit advisor tool is depending on the **BEPS** and the **DEA** modules for its input. Basically the tool is able to perform what-if scenarios basd on substitutions of building components that are in its turn selected with the help of applied artificial intelligence. The Core semantic ingredient of the tool regarding its input is called the **RCO (Retrofit Calculation Option)** and is in essence a wrapper that takes the existing vocabulary that is already described in this document regarding both elements coming from the Building information Model (=BIM) like windows and the type of glazing together with the associated parameter sets describing heat and sound permeability. The wrapper is necessary to hold the meta-information the tool gathers when applying these in calculated alternatives.

When successfully applying an RCO the metrics with respect to its relative improvement and the original option it replaces are stored, so in upcoming scenario's this information steers the selection of the different alternatives for any component that can be substituted by a newer or better alternative.

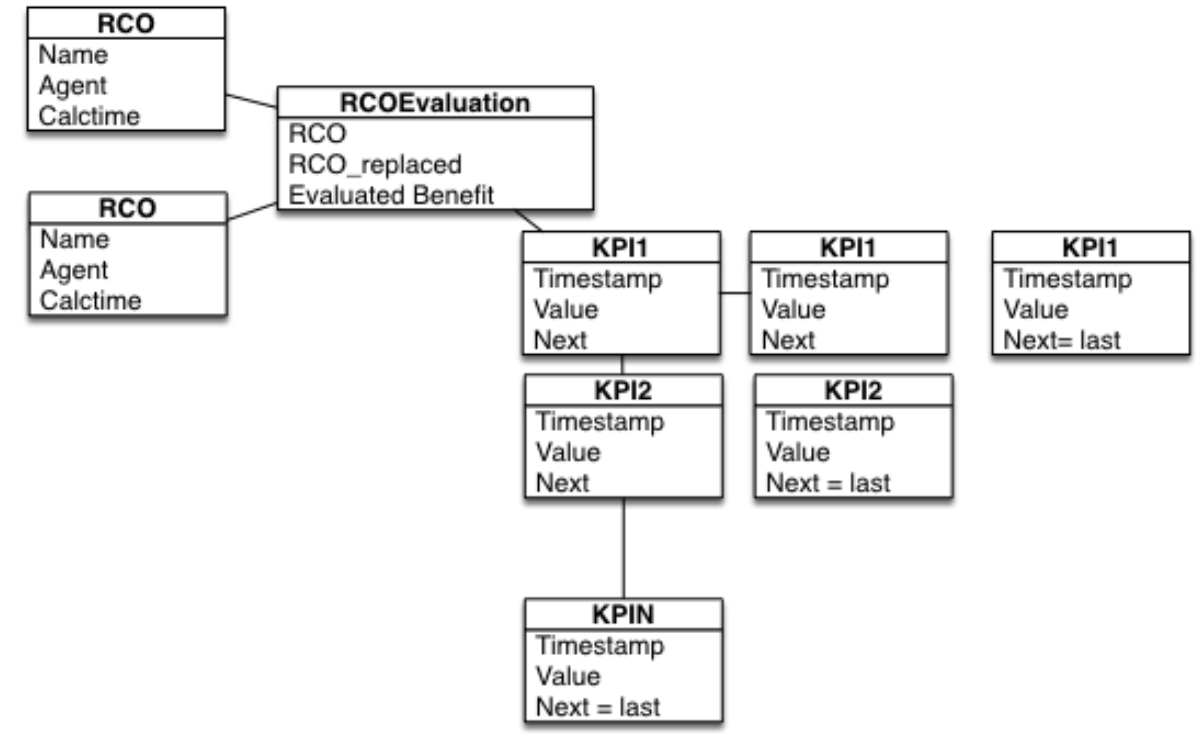

Figure 20 Retrofitting KPI evaluation modeling

Main attributes in the Retrofit Calculation entity:

- ***name*** : Unique identifier for the retrofitting option
- **Agent reference(URL)**: Address of MOEEBIUS services that are involved in the calculation process.
- ***Calculation time***: Calculation window or period in which the retrofitting operation will be evaluated.

Every RCO may have been a part of calculations already performed by the Retrofitting Advisor Tool. The tool maintains an internal annotation system. This system contains records with the following structure:

Main attributes in RCO_Evaluation entity:

- ***RCO***: List of Retrofitting options to be considered for the overall evaluation process:
- ***Evaluated benefit:*** List of KPIs to be considered for the retrofitting evaluation.

Due to the expected calculation time needed during simulations and the preference for being able to assess as many options as possible in the given time not all KPI's are evaluated at all times. Especially as the majority of them are usually not that relevant to the evaluation requested. Only those KPIs considered relevant will be recorded.

5.6.2 Outputs

The Retrofit Advisor tool will be separately controlled by the user and have a web based interface. A constituent part of that interface will be a browser for the coded timelines as described above.

5.7 Facility Management and ESCO Management Tool

5.7.1 Management of energy prices

One of the main objectives of the project is to incorporate dynamic pricing schemas at the business strategies of Aggregator/ESCo, and thus special focus is delivered on modeling these specific data attributes. As presented in D2.4, the DAFM module will be integrated with an external component, named as **Dynamic Pricing Simulation Engine**. The role of the component is constantly collect and analyse energy price data, following market dynamic fluctuations, and further identify how user behaviour is transformed on the basis of variable Electrical Peak Demand, Electrical Energy (Usage), Reactive Demand Tariffs, or through the offering of Incentives and Rebates. The goal of this section is to model the different price related elements defined MOEEBIUS project.

Again the starting point for this analysis is the static CityGML representation and the building class as part of the portfolio of Aggregator/ESCo. Each building of the portfolio is associated with:

- Energy Contract that defines the retailer price schema of each prosumer of the portfolio
- DR Contractual Agreement that defines the Contractual Agreement for participating in DSM strategies triggered by the Aggregator.

The analysis starts with the definition of "**contract"** model schema, adopted by IEC CIM standardization (see Figure 21).

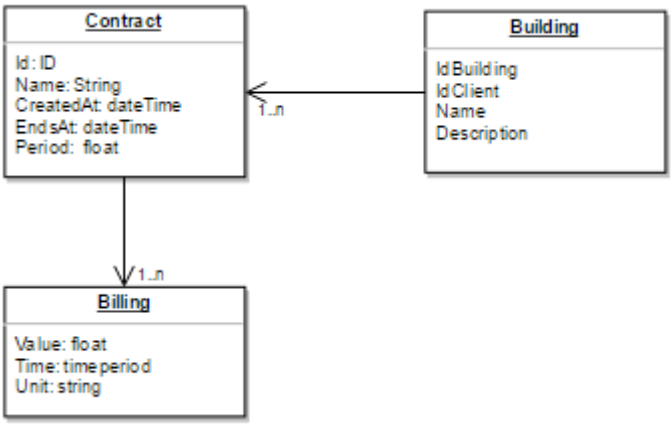

Figure 21 Billing Contract Class diagram

We have to point out that the above data structure is adopted for both retailer pricing and implicit DSM strategies implementation.

Then, the focus is on modeling the different DR Contractual Agreements towards the implementation of DSM strategies and the compensation for participating at explicit DSM strategies (automated DR). As there is no standardized approach for modeling DR compensation, a state of the art DR program running in the U.S is considered as reference

- In this context, the DR services operator provides participants with free smart home devices for enrolling in the program.
- What related to maximum participation on DR services, the idea is to follow the structure of existing commercial projects and set a maximum number of 12-15 activations per user during a 6-month period.

Taking into account the above limitations, the details of the model are presented in the following class:

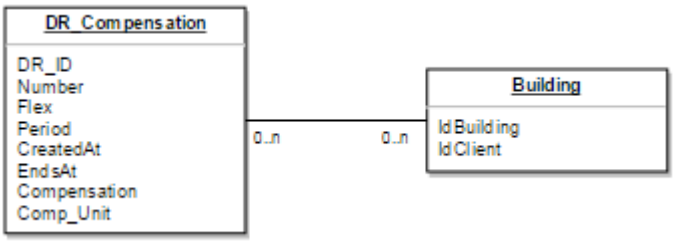

Figure 22 DR Compensation Class diagram

The different fields of DR Contractual class are described:

- **DR ID:** A unique identifier for the DR program (defined by the Aggregator)
- **Number** The total number of events associated with this DR program
- **Flex:** The total amount of flexibility defined by the contract
- **Period:** The active period for this DR program (month, year)
- **CreatedAt:** The starting date for the DR program
- **EndsAt:** The end date for the DR program
- **Compensation:** The end users compensation for participating in a DR program
- **Comp_Unit:** The unit type for the DR compensation value (*euro/ KWh* or *euro* or *dimensionless*)

We have clearly distinguished the energy prices modeling from the metrics modeling as this is one of the main innovations of the project, to incorporate a **Dynamic Pricing Simulation Engine** and further evaluate the impact from different tariff policies.

5.7.2 Implementation of DSM strategies

The very last functionality offered by the MOEEBIUS district management framework is the selection and further implementation of the best fitted DSM/DR strategies. For DSM strategies implementation, we are adopting an industry (DSM services) specific standard.

One of the main objectives of the project is to define a standardized framework for dynamic Virtual Power Plants (VPP) formulation with enhanced aggregated

flexibility capabilities to participate in event-, time-, location- or price-based Demand Side Management Strategies. Towards this direction, OPENADR data model is considered as the basis for implementation of district level DSM strategies. A short summary of OPENADR standard is provided, focusing on the aspects that are defined in business scenarios and uses cases of the project.

OpenADR 2.0, an internationally-recognized standard for Automated Demand Response (ADR), defines the interaction between an ADR server and client, setting that way the interfaces for standardized implementation of Demand Response Strategies. Open Automated Demand Response (OpenADR) consistently conveys the DR signals to consumers (business or residential) from the system operator, facilitating a timely and predictable response, while allowing choices by the end consumer. Automating DR provides consistency in communicating the DR event while facilitating faster responses that translate into greater energy savings and more options for combining energy resources, which helps keep prices lower for everyone. In addition, automating DR also provides pricing continuity between wholesale (generation and transmission) and retail (distribution) markets. For electricity providers to be able to communicate DR and Distributed Energy Resources (DER) signals with each other and with customers, it is necessary to have a common language understood by all parties. The OpenADR standard provides both a **common language** and a **common platform** for all providers and consumers of DR through the functions and features of a Demand Response Automation Server (DRAS).

Open ADR has two primary entities: A Virtual Top Node (VTN) that can initiate Demand Response events and a Virtual End Node (VEN) that can participate in an event (i.e., shed load). VTNs and VENs can be implemented in a hierarchical relationship, such as a utility playing the role of a VTN sending events to an aggregator that receives the event as a VEN then propagates the event downstream playing the role of a VTN, as shown in the next figure.

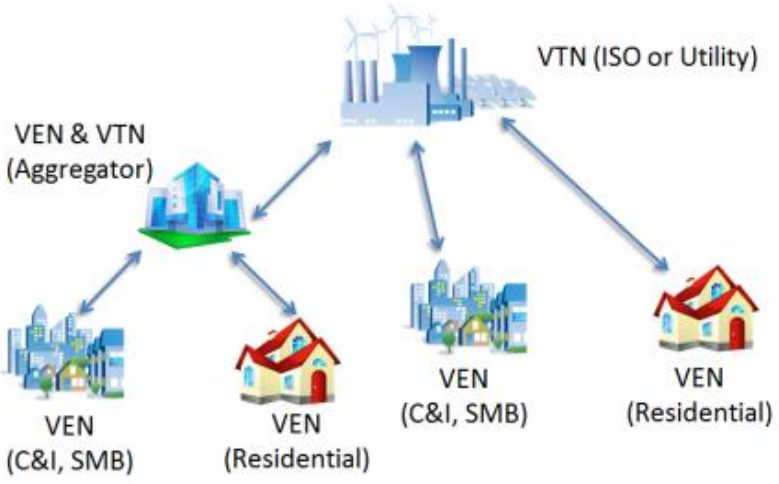

Figure 23 OPENADR Business Framework

VENs are associated with one or more resources that have a load profile that can be modified. Each VEN has some application logic associated with it, such as an Energy Management System that has been pre-programmed to initiate a specific set of actions in response to signals contained in the Demand Response event payload sent by the VTN. Note, however, that the application logic is NOT part of OpenADR. **The actionable information contained in a Demand Response event can include values such as energy pricing, load dispatches, simple levels, and a wide variety of other signal types, all associated with specific future time intervals**. OpenADR supports the following services related to Demand Response interactions:

- EiEvent Service - Enables VTNs to send events and for VENs to optIn or optOut of events
- EiReport Service - Enables VTNs and VENs to declare their reporting capabilities, request reports from each other, and to deliver both one shot and periodic reports.
- EiOpt Service - Enables VEN to declare temporary availability schedules
- EiRegistration Service - Establishes a relationship between a VEN and VTN, but does not include enrollment.

The focus here is on the incorporation of data modeling aspects, considered for the adaptation of the core OpenADR services (EiEvent& EiReport) in MOEEBIUS platform. We first identify the types of DR signals to be considered in MOEEBIUS project, taking into account business requirements.

Signal Category	Name (signal Name)	Type	units	Description
Price of electricity	ELECTRICITY_PRICE	price	currency/kWh	This is the cost of electricity expressed in absolute terms
	ELECTRICITY_PRICE	priceRelative	currency/kWh	This is a delta change to the existing price of electricity
	ELECTRICITY_PRICE	priceMultiplier	None	This is a multiplier to the existing cost of electricity
These instructions are used to set the load to values that can be expressed in terms of the desired load	LOAD_DISPATCH	setpoint	powerXXX	This is used to dispatch loads to a specific amount
	LOAD_DISPATCH	delta	powerXXX	This is used to dispatch loads to some offset from an agreed upon baseline. Note that the baseline may be the current power consumption
	LOAD_DISPATCH	multiplier		This is used to dispatch loads as some multiple of power against some agreed upon baseline. Note that the baseline may be the current power consumption

Table 1 OpenADR DR Signals

There are 2 types of services examined in the project, price based DR events (where an electricity price DR signal is triggered to the end users) and auto DR event service (where a direct load command is triggered to Building BMS). These are the high level principles considered on modeling "EiEvent Service" data parameters.

Events are generated by the VTN and sent to the VEN using the oadrDistributeEvent payload containing one or more events described by the oadrEvent element. Some events require a response and others do not as indicated by the oadrResponseRequired element in the event description. If a response is required, the VEN acknowledges its opt-in or out-out disposition by responding with an oadrCreatedEvent payload containing eventResponse elements matching each oadrEvent. If no response is required, the VEN must not reply with oadrCreatedEvents (or oadrCreateOpt) payloads for this event. The detailed data attributes specified for OpenADR EiEvent service are presented:

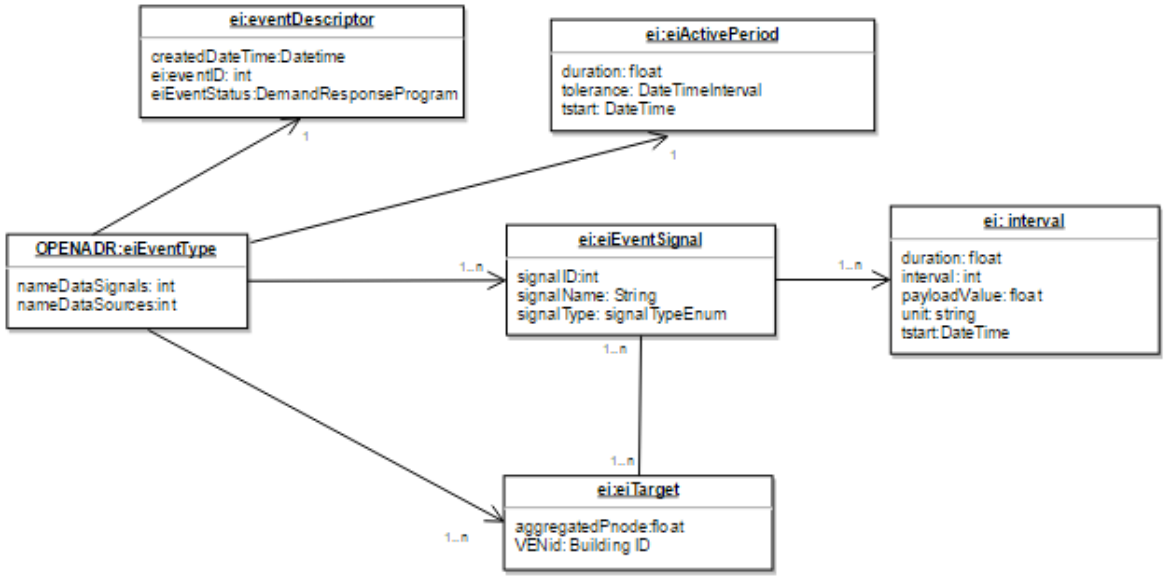

Figure 24 EiEvent class diagram

Where,

eiEventType: A root class for the DSM strategy triggered by the ESCO/Aggregator

eventDescriptor: A description about the type of the DSM strategy triggered

- createdDateTime: The date and time created
- eventID: A unique ID for the DSM strategy
- eventStatus: A selection from the DSM strategies examined in the project.

ActivePeriod: The overall period of the DSM campaign

- duration: Total duration of DSM strategy

- tstart: The start time of DSM strategy
- tolerance: the tolerance level on the date time period of the DSM strategy

eiEventSignal: The actual DSM strategy triggered by ESCO/Aggregator

- signalID: the unique ID for the signal
- signalName: the name of the signal
- signalType: The type of the signal, a selection from:{type-serials}

eiTarget: The customers participating at the DSM campaign

- aggregatedPnode: The amount of flexibility requested by the VEN
- VEN ID: The unique ID of the customer participating in DSM strategy

interval: The details of the eiEventSignal

- duration: total duration of the interval
- interval: a counter for the interval of eiEventSignal
- payload: the DSM signal from Aggregator to prosumers
- unit: The unit of the payload {currency/kWh or power}
- tstart: the start time for the interval

The same approach is considered also for OpenADR 2.0b Report Service. Each report type is intended to represent a certain set of reporting functionality that is supported by either a VEN or a VTN that claims to support that type. There are several sub-categories of Data Reports as described below.

- DATA REPORTS HISTORY – This is a type of data report in which the history of the data point values is logged and can be subsequently requested. These include the following specific types:
  - HISTORY_USAGE – these are logs of usage data that are typically logged by VEN's and can be queried by the VTN
- DATA REPORTS TELEMETRY – The term telemetry in the context of OpenADR refers to data that is reported periodically in real time and includes the following specific report types:
  - TELEMETRY_USAGE – this is usage data that is periodically reported from the VEN to the VTN in real time.

From the services supported by the protocol, we consider "Send Reports" for MOEEBIUS project. The reports sent in the oadrUpdateReport payload using the **EiReport schema** with information about DR participation. A simplified version of OPENADR EiReport schema is selected for MOEEBIUS project.

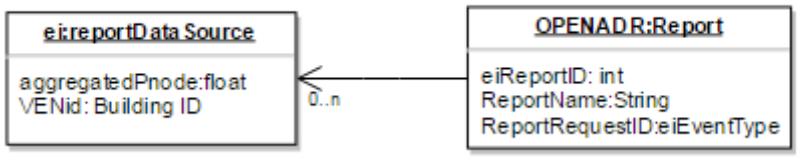

Figure 25 EiReport class diagram

Where,

eiReportID: A unique ID for the report provided by the end of the DSM strategy

ResponseName: A short description about the type of the report

ReportRequestID: The associated EiEvent triggering the DSM campaign

aggregatedPnode: The amount of demand flexibility actually offered by the VEN

VEN ID: The unique ID of the customer participating in DSM strategy (Building ID as defined in DIM)

5.8 Sensor related data models

MOEEBIUS framework, as any other framework that target physical systems management, has to be feed by field data coming from sensors or actuators. Prior to define the data format that MOEEBIUS will adopt to upload sensor and actuator values to the MOEEBIUS Quest layer, a summary of the initiatives that try to generalize the integration of field values with cloud frameworks will be described.

Sensor Web Enablement[14] encompasses a set of standards promoted by the OGC to enable developers to make all types of sensors, transducers and sensor data repositories discoverable, accessible and usable via the Web.

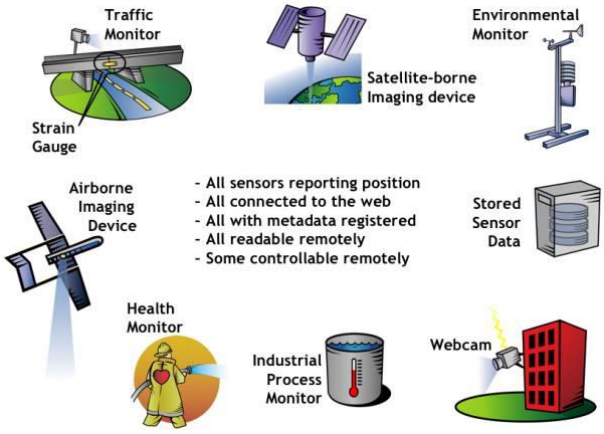

Figure 26 Data covered by the Sensor Web Enablement standards

[14] http://www.opengeospatial.org/ogc/markets-technologies/swe

The Sensor Web Enablement framework, also referred to as Semantic Sensor Web, which is intended for making visible, accessible and usable via web through standard protocols any type of data produced by sensors, transducers and sensor data repositories. Thus, it is a particular implementation of the Internet of Things (IoT) concept.

For that purpose, the **Sensor Model Language**[15] (**SensorML**) is defined, with the aim to provide a robust and semantically-tied means of defining processes and processing components associated with the measurement and post-measurement transformation of observations. This includes sensors and actuators as well as computational processes applied pre- and post-measurement. The main objective is to enable interoperability, first at the syntactic level and later at the semantic level (by using ontologies and semantic mediation), so that sensors and processes can be better understood by machines, utilized automatically in complex workflows, and easily shared between intelligent sensor web nodes.

In parallel, the OGC also defines the **SWE Service Model Implementation**[16] Standard. This standard currently defines eight packages with data types for common use across OGC Sensor Web Enablement (SWE) services. Five of these packages define operation request and response types. The packages are:

- Contents – Defines data types that can be used in specific services that provide (access to) sensors
- Notification – Defines the data types that support provision of metadata about the notification capabilities of a service as well as the definition and encoding of SWES events;
- Common - Defines data types common to other packages;
- Common Codes – Defines commonly used lists of codes with special semantics;
- DescribeSensor – Defines the request and response types of an operation used to retrieve metadata about a given sensor;
- UpdateSensorDescription –Defines the request and response types of an operation used to modify the description of a given sensor;
- InsertSensor – Defines the request and response types of an operation used to insert a new sensor instance at a service;
- DeleteSensor – Defines the request and response types of an operation used to remove a sensor from a service. These packages use data types specified in other standards. Those data types are normatively referenced herein, instead of being repeated in this standard.

Other standards included in the SWE are:

- Observation & Measurements (O&M)[17]: XML implementation for the OGC and ISO O&M conceptual model. It defines XML schemas for observations, and for features involved in sampling when making observations.

[15] http://www.opengeospatial.org/standards/sensorml
[16] http://www.opengeospatial.org/standards/swes
[17] http://www.opengeospatial.org/standards/om

- Sensor Observation Service (SOS)[18]: This standard defines a Web service interface which allows querying observations, sensor metadata, as well as representations of observed features. Descriptions of the sensors are represented in SensorML and measured values in O&M.
- Sensor Planning Service (SPS)[19]: Defines interfaces for queries that provide information about the capabilities of a sensor and how to task the sensor.

Ontologies have been widely used as semantic models for the representation of heterogeneous sensor data. Apart from SensorML described above, some of the most relevant ontologies describing sensors, their capabilities and observations are: Ontosensor [Goodwin C., Russomanno D.J. (2006)] and CSIRO [Neuhaus H., Compton M. (2009)] are the ones covering the wider range of sensing concepts, organizing sensors and data into a hierarchical structure.

5.8.1 Relevance of sensor standards for MOEEBIUS

The set of standards included in the SWE Framework represents the most well defined standard way to make accessible via Web the sensors and data captured by a sensor network infrastructure. Several of the existing frameworks and IoT architectures are compatible with some of the SWE standards (e.g. FIWARE support UL2.0, which is a simplification of SensorML). Within the SWE suite of standards, recently approved is SensorThings[20], which defines a standard way to interconnect IoT devices, data servers, and applications over the Web through a REST-like API. Within the project, the different sensor models that will not be provided by the middleware will be considered and translated into data classes based on the MOEEBIUS component data requirements.

6 MOEEBIUS data management/exchange framework

As previously discussed, the Common Information Model describes the data format and the interfaces used to exchange information between the different modules involved in MOEEBIUS. The implementation of a common model for data exchange guarantees the scalability and flexibility of the MOEEBIUS framework. Moreover, MOEEBIUS is not considered as a standalone and isolated framework and to this end it needs to interoperate with other systems involved in building or district level management. In this context, standardization of interfaces and data access mechanisms is required for ensuring MOEEBIUS's interoperability and "open" approach.

The MOEEBIUS data management and exchange framework (MOEEBIUS data service server) proposed in this section leverages interoperability through the standardization of interfaces, data access and the use of standard based data formats.

[18] http://www.opengeospatial.org/standards/sos
[19] http://www.opengeospatial.org/standards/sps
[20] http://www.opengeospatial.org/projects/groups/sensorthings

The data backed up by the common information model will be organized in databases, rooted on well-known standard BIM and BEM data models, which will support the development of the elements that comprise the MOEEBIUS knowledge domain, such as climatic elements, building materials, energy resources, etc. For this purpose, the next standards will be used and extended as necessary; gbXML, IFC, GIS and CityGML. The extensions will allocate information elements required to modeling DER and district heating elements (T3.3), occupants comfort profiles (T3.4) and indoor air quality models (T3.5).

The data provided by the MOEEBIUS data management/exchange framework will be accessed as web services data services by the different components. Thus, data will be accessible via REST and SOAP interfaces and basic CRUD operations will be available so as to let the components interact with the databases. Figure 27 shows the data management architecture, which is able to expose both SOAP and REST services to deal with various sources of data.

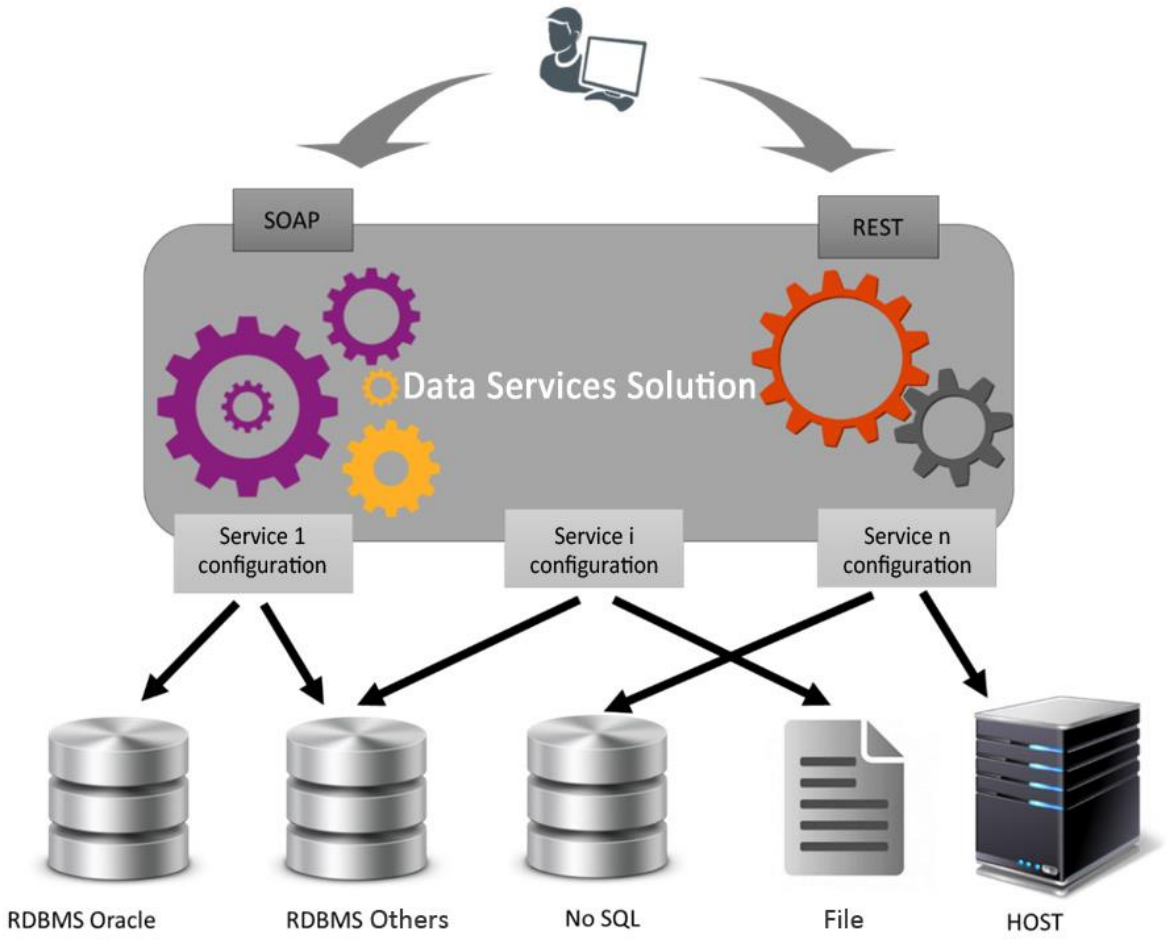

Figure 27 Data management/exchange framework architecture

Any data access, either from a traditional relational database, NoSQL or data files (XML, CSV, etc.) or even custom warehouses can be configured with the data management framework. It supports the next features:

- Data Sources: RDBMS (via JDBC), CSV, Excel, Cassandra, Google Spread-sheets, RDF or even any web page via "scraping"
- Transportation: Exposed services can be accessed via HTTP and / or HTTPS

For each service there are very interesting options available, such as:

- Transactions: Each data service can expose operations that can be executed in a transactional way, being able to accept the changes or abort (roll-back)
- Validations: For instance, during an insertion, the service can validate the entry (thresholds values, length, etc.)
- Nested operations: The data service can be built upon nested transactions between different operations exposing the result with a single service
- Relate different data sources: Users can cross information from different tables and / or diagrams exposing the information as a single data set
- Throwing events: The services can launch events to perform tasks or other operations

Summarizing, the MOEEBIUS data management/exchange framework will be a repository of common data will hold the data shared by several MOEBIUS modules. The data schemas will be aligned with existing BIM and BEM standards, allowing the interoperability with other systems involved in building or district level management.

7 Conclusions

This deliverable describes the Common Information Model (CIM) format and the MOEEBIUS data management/exchange framework, which will support the data model by means of enabling data exchange between the components.

The implementation of a common model for data exchange based on well-known standards guarantees the scalability and interoperability and "open" approach of the whole MOEEBIUS framework.

Regarding the data modelling requirements, the next elements that will build-up the CIM were identified:

- **Measurements and Verification (M&V) elements**, including a set of energy related KPIs at district level and building level, including metrics to monitor the performance of the MOEEBIUS components are described. These metrics will enable the definition of business and technical objectives within the use cases and the follow-up of the measurements that pursue these intended goals. On the other hand, some of the metrics will be used to monitor the performance of the MOEEBIUS framework itself. Some of these metrics include, for example, the time response of the components or the number of service calls per time unit
- **Building level elements (BIM elements)**, which will support the generation of enhanced simulation models in order to allow the accurate representation of the real-life complexity of the building and its surroundings. These BIM elements will be based on IFC4 and gbXML data standards. On the one hand, MOEEBIUS will leverage and extend the IFC4 standard, where necessary, to support the architectural, structural and thermal design of the buildings. On the other hand, some of the elements from the gbXML schema will be reused to support the description of updated weather conditions, forecasts and ground conditions
- **Occupancy and behavioural elements**, which will support the data requirements of the Occupants profiling engine. The building loads connected to facilities usage will be updated with sensor data to achieve an accurate occupancy profiling model, based on occupancy patterns. These patterns will be further incorporated in the Building Energy performance Simulation (BEPS) tool. The occupancy profiling data will be associated to the building zone elements according to the BIM model above
- **Energy Resources Operational profile elements**, which will support the data requirements of the MOEEBIUS Demand Flexibility Engine. Energy resource profiles will reflect real-time demand flexibility as a function of multiple parameters, such as device operational characteristics or energy costs. The profiles of the energy resources will be associated to building zones of the BIM
- **District level elements**, which will enable the delivery of the MOEEBIUS district management framework, as an aggregation of building level elements

along with Distributed Energy Resource (DER) elements, such as district heating facilities, distributed renewable energy generators or storage devices (infrastructure). These distributed / common resources will be based on the CityGML protocol, and specifically, on the 3D City Database. These elements will provide a relationship to the BIM models based on the IFC standard

- **Automated Demand Response elements**, which will leverage district management strategies based on electricity prices and load dispatches. These demand response resources will be based on the OpenADR 2.0 standard that provides a common language and platform for all providers and consumers of demand response through the capabilities of a Demand Response Automation Server (DRAS), i.e. energy savings by means of combining energy resources, which maintain prices low for all the involved agents

All in all, these elements will support an improved and realistic Building Energy Performance Simulation (BEPS) and the implementation of Automated Demand Response strategies by means of aggregating building level elements.

The document presents the following holistic business scenario defined in D2.4: **Business Scenario 5 – DSS towards the establishment of a sustainable building level and district level environment** (see Figure 5). The goal of this business scenario is to leverage a holistic and dynamic modelling and simulation / optimization approach that enables:

1. Improved predictions on the basis of more accurate and dynamically updated Building Energy Performance Simulation (BEPS) models
2. Real-time building and district performance optimization through control and predictive maintenance
3. Optimized retrofitting decision making on the basis of improved (LCA / LCC-based) performance predictions

Taking the holistic scenario as reference, the document further elaborates on the data exchange requirements envisaged for each of the components. In this regard, connections to the relevant sections in the building industry reference data models are provided.

As a final point, the MOEEBIUS data management/exchange framework is presented. This framework, acting as a data services server, will provide methods to access the elements of the Common Information Model. It describes the way the MOEEBIUS components will handle the information they require.

The first version of this deliverable was submitted on its due date on M12, stating that it would be fully aligned with architecture and middleware requirements through an iterative process. After releasing D3.1, the present deliverable has been revisited to be consequently structured. This second version of the deliverable can be considered a stable one that might be updated after final integration into the pilots

8 References

[1]. MOEEBIUS Description of Action, MOEEBIUS Project (680517)

[2]. "Web Services Architecture". World Wide Web Consortium. 11 February 2004. 3.1.3 Relationship to the World Wide Web and REST Architectures. Retrieved 29 September 2016.

[3]. D3.1 MOEEBIUS Framework Architecture including functional, technical and communication specifications

[4]. D2.3 MOEEBIUS Energy Performance Assessment Methodology