D6.2 PLATFORM ARCHITECTURE AND DESIGN

Building and integrating the platform and its modules

Project title	**Collaborative Recommendations and Adaptive Control for Personalised Energy Saving**
Project acronym	**enCOMPASS**
Project call	**EE-07-2016-2017 Behavioural change toward energy efficiency through ICT**
Work Package	**WP6**
Lead Partner	**SMOB**
Contributing Partner(s)	**PMI, CERTH, GRA, PDX, SUPSI**
Security classification	**Public**
Contractual delivery date	**31/10/2017**
Actual delivery date	**30/10/2017**
Version	**1.6**
Reviewers	**PMI (C. Pasini, P. Fraternali), CERTH (S. Krinidis)**

History of changes

Version	Date	Comments	Main Authors
0.1	11/07/2017	Defining Table of Content	L.Caldararu (SMOB)
0.2	12/07/2017	Defining document concepts and vision	S Calit (SMOB)
0.3	01/09/2017	Elaborating content of SMOB, intial version	S Calit (SMOB), L Caldararu (SMOB)
0.4	10/09/2017	Elaboration of POLIMI input and integration in the document	C. Pasini (PMI), S Calit (SMOB)
0.5	12/09/2017	Elaboration of SUPSI input and integration in the document	A Rizzoli (SUPSI), S Calit (SMOB)
0.6	29/09/2017	Added content for Notification Engine	S Calit (SMOB)
0.7	02/10/2017	Elaboration of GRA input	B.Hidasi (GRA)
1.0	02/10/2017	Compiled content from all Partners	S Calit (SMOB)
1.1	13/10/2017	Refine of content according to review telco	S Calit (SMOB)
1.2	17/10/2017	Update general architecture with new component structure	S Calit (SMOB)
1.3	18/10/2017	Update Notification Engine component	L Caldararu (SMOB)
1.3.1	18/10/2017	Update Notification Engine component	S Calit (SMOB)
1.3.2	18/10/2017	Updated Utility Console, first round of Quality Check	C. Pasini (PMI)
1.3.3	20/10/2017	Formatting, grammar check, updates following Project review meeting	S. Calit (SMOB)
1.3.4	23/10/2017	Update Component – Use Case mapping, list of services	S. Calit (SMOB)
1.3.5	24/10.2017	Updated Inference Engine and Efficiency Console for Building Managers component diagrams	S Krinidis (CERTH)
1.3.6	25/10/2017	Added Document Structure chapter	S Calit (SMOB)
1.3.7	25/10/2017	Updated Use Case list	C Pasini (PMI)
1.3.8	26/10/2017	Included Consumption and Sensor Data acquisition in the set of synchronised processes.	S Calit (SMOB), L Caldararu (SMOB)

1.3.9	27/10/2017	Quality Check	C Pasini (PMI)
1.4.0	27/10/2017	Updated Use Case – Component mapping table and Component – Services mapping table	S Calit (SMOB)
1.4.1	27/10/2017	Updated Use Case – Component mapping table and Component – Services mapping table	S Calit (SMOB)
1.4.2	27/10/2017	Adjustments after Quality Check by PMI	L Caldararu (SMOB)
1.5	30/10/2017	Final Revision	P.Fraternali (PMI)

Disclaimer

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 723059.

TABLE OF CONTENTS

Table of Figures

EXECUTIVE SUMMARY

This document is the Deliverable **D6.2: Platform architecture and design**, which, according to the Description of Work (DoW) has to design and describe:

- *information and data model (Chapter 4 –Information View);*
- *platform components (Chapter 1 – System View and Chapter 3 Logical View);*
- *services (Chapter 5 – Process View);*
- *applications (Chapter 3 – Logical View);*
- *communication protocols (Chapter 5 – Process View);*
- *integration mod (Chapter 5 – Process View).*

The document also intends to serve as an orchestration instrument among Partners, to understand each-other the place, role and play in Platform implementation.

There are no less than 14 Partners involved in the project, each of them having specific tasks to accomplish to meet in the most lean and efficient way, the Platform requirements in terms of: functionality, usability, scalability, security and reusability.

So one of the main purposes of the document, among many other, is to endow each Partner with such a tool that allows him to clearly understand what is expected from him and what to expect from other Partners. In the most precise way, the focus of the document will be on orchestration and integration among Platform components and between Platform and external 3rd parties. We'll leave the implementation details a bit loose but will enforce the interfaces each component will implement.

This document is related to following deliverables:

- *D2.1 – Use Cases and Early Requirements (M8)*: platform should be designed to accommodate all Use Cases described in this deliverable, a mapping between Uses Cases and processes implemented by the Platform is presented in Chapter 6 – Process View';
- *D2.2 – Final Requirements (M12)*: this deliverable will be used to refine the Architecture and Design of the Platform;
- *D2.3 – Functional System Specifications (M18):* this deliverable will be used to further refine the design of the Platform.
- *D3.1 – Datasets with Context Data and Energy Consumption Data (M12):* this deliverable puts in Platform context, the energy consumption database which is detailed in D3.1 deliverable.

This deliverable will be updated to be in sync with related deliverables and will serve as the reference document to the design and development of the three prototypes of the Platform:

1. V1.0 Initial Prototype (M18);
2. V2.0 Second Prototype (M24);
3. V3.0 Release Version (M36).

The structure of the document is presented below.

Chapter 1 – Introduction

This chapter places the document in the current enCOMPASS Project context, how it relates to other deliverables. Due to the fact that the project scope includes 3 Pilot deployments with 3 different Utility

companies, there are technical and legal deployment constraints that the Platform implementation MUST consider. This chapter presents details constraints and their direct impact in platform architecture .

This chapter also presents the modelling methodology of Platform architecture, the definitions and acronyms across the document.

Chapter 2 – System View

This chapter presents a structural overview of platform architecture as a Service Oriented Architecture. It also details each architectural layer: Data, Business and Consumer, their role and interactions with other layers.

Chapter 3 – Logical View

This chapter presents structure of components, applications or systems, that will build up the Platform and their internal composition. Each Platform component is detailed by its internal components using UML Component View diagrams. Description of functionality provided by each internal component and how they interact, completes the description of each Platform component.

Chapter 4 – Information View

This chapter presents the DataModel of Central Database and Data Model of each component. UML Entity Relationship diagrams were employed for graphical representation. For an Architecture and Design document, data model was not detailed in detailed technical implementation: data type, data size, foreign keys but we rather focused on the data structure and relationships.

Chapter 5 – Process View

This chapter is focused on the design of Platform component interactions. Starting from the Use Cases defined in "D2.1 Use Cases and Early Requirements", this chapter verifies that each Use Case is covered by the proposed Architecture and also propose the minimal list of services each component should implement. This chapter also details design solution for services synchronization and long running processes synchronization.

Chapter 6 – Deployment View

This chapter presents the deployment model of the Platform. It identifies the Packages to be deployed, the minimal number of machines to be deployed, the software infrastructure needed for each server that will run a Platform package. This chapter also presents the security design solutions either implemented in the Platform or put in place at the deployment site.

Chapter 7 – References

This chapter enlist all external intellectual books, open source projects, studies that were used to design the Platform architecture and also build this document.

1 INTRODUCTION

1.1 DEFINITIONS AND ACRONYMS

Project Stakeholders

Utility Company – Energy provider company that will implement a pilot of the Platform.

End User – Person who will use Client applications of the Platform. Could be: householder, student, teacher, Building Manager.

Partner – Project Consortium member company.

Frameworks and technologies

SOMA - Service-oriented modelling and architecture

IFML – Interaction Flow Modelling Language

RM-ODP – Reference Model of Open Distributed Processing

GWAP – Games with a Purpose

MD5 – algorithm that produces a 128-bit hash value. Used to check data integrity.

1.2 MODELLING METHODOLOGY

We elaborated the Platform architecture using the View Model framework. We used a mix of 4+1 Zachman (Zachman) and RM-ODP (RM-ODP) implementations of the general View Model framework for the overview representation of the Platform.

We chose the most relevant viewpoints from the above-mentioned View Model frameworks and depicted these viewpoints using the most appropriate UML views. Each time needed we used custom views (non-UML) that better illustrated one or more specific topics, still with respect to relevance for the addressed stakeholder.

Architectural Models/Views chosen:

- Logical View (Component View);
- Information View (Data Model View);
- Process View (Services View);
- Physical View (Deployment View).

We also employed SOMA to better represent the System View of the Platform (IBM, Service Oriented Analysis and Design).

We finally obtained a clear and understandable representation of the system for each of stakeholders of project consortium: business analysts, mathematicians, project managers, developers, architects so that they be able to deliver their specific tasks in a perfect correlation with each other, contributing to the project level deliverables and objectives.

Also, an important objective of this document is to deliver a clear and complete image of the system to other interested 3rd parties: EU Project Officers, future interested Utility companies, local authorities.

1.3 CONSTRAINTS

We elaborated the Platform's architecture and design with the respect to several constraints:

1. **User profile, consumption and sensor data to be stored locally on the servers of each Utility Partner;**
2. Heterogeneous pilot Utility Partners existing systems: WVT (Greece), SES (Swiss), Haßfurt (Germany);
3. Heterogeneous End-User typology, detailed in D2.2 – Final Requirements:
   - Residential
     - Household user;
   - Schools
     - Teacher;
     - Primary school student;
     - Vocational group representative;
     - Vocational school students;
   - Public Buildings
     - Workers in public buildings;
     - Manager of public building.
4. Accommodate in a consistent manner: technological differences among Utility companies, attitude, cultural and educational differences among End Users;
5. Clear partitioning of concerns:
   - structural;
   - functional.

.

2 System View

Technical implementation of enCOMPASS Platform is based on a layered architecture. Each layer is designed with respect to separation of concerns principles. The proposed architecture is organized in four distinct layers:

- Data layer
  - Data acquisition;
  - Data/object layer;
- Business process layer;
- Consumer layer.

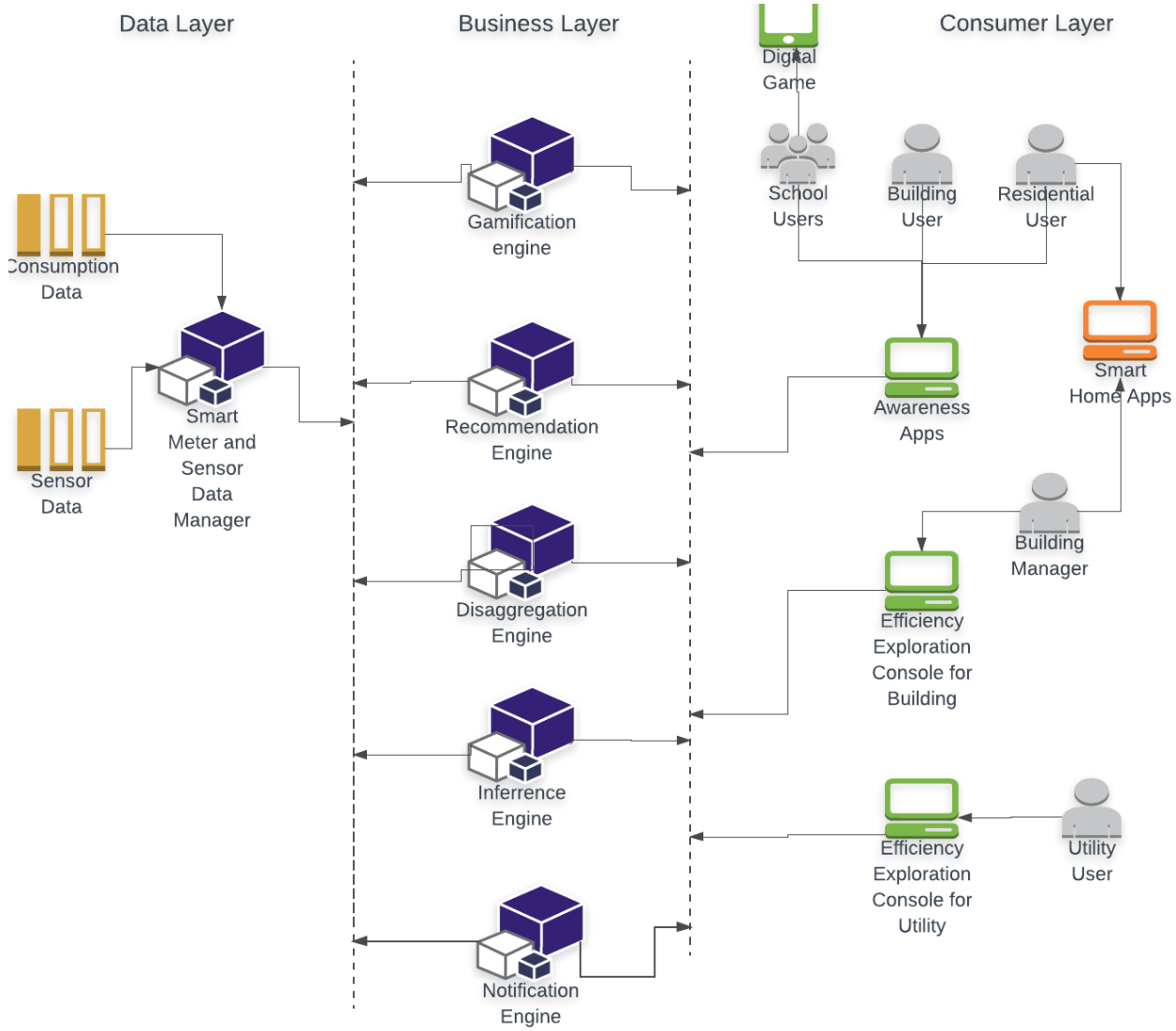

Figure 1 - System View

2.1 Data Acquisition Layer

This layer is responsible with bulk data acquisition and bulk data delivery. Inputs of this layer are:

- Energy usage data files from Utilities;
- Sensor data files from Utilities.

This layer plays the role of a mediation component that handles raw data acquisition, transformation and storage in a format that can be used at upper layers.

2.2 Data Object Layer

This layer is responsible for data management and storage. This layer will expose services for upper level for basic access to data. It will manage data such as:

- Energy usage;
- Sensor;
- Building information;
- User profile;
- Game actions and rewards;
- Recommendation;
- Notifications.

2.3 Business Process Layer

This layer is responsible for implementation of business logic, it will expose business services for Consumer layer. Business level components are:

- **Gamification Engine**. This component will provide game scenarios and will handle user interactions with the platform through social game clients. This is a platform built-in component, as it satisfies a critical enCOMPASS project objective.
- **Recommendation Engine**. This component will provide personalized adaptive energy saving recommendations, based on the user's profile and context.
- **Disaggregation Engine**. This component will provide data disaggregation and analysis services to identify user behaviour patterns.
- **Notification Engine**. This component will provide notification services to other Platform components.
- **Inference Engine**. This component will process sensor and consumption disaggregated data and will generate inferred indicators like: User Comfort Level and Occupancy Level.

2.4 Consumer layer

This layer consists of client applications for services exposed by the Business Process Layer. Consumer of platform business services that are foreseen at this stage:

- **Awareness Applications.** Energy consumer applications for households, public building dwellers, schools. Gamification client application.
- **Digital Game Extension.** Mobile game that bridges the Platform with the board-game.
- **Efficiency Exploration Console for Utility**. Energy utility administrative application.
- **Efficiency Exploration Console for Building Manager**. Administrative application for Building Manager.

As the platform implementation advances it will always possible to connect to the enCOMPASS services any client applications that implements the platform API specification.

3 LOGICAL VIEW

3.1 COMPONENT VIEW

The component view of the Platform is presented below:

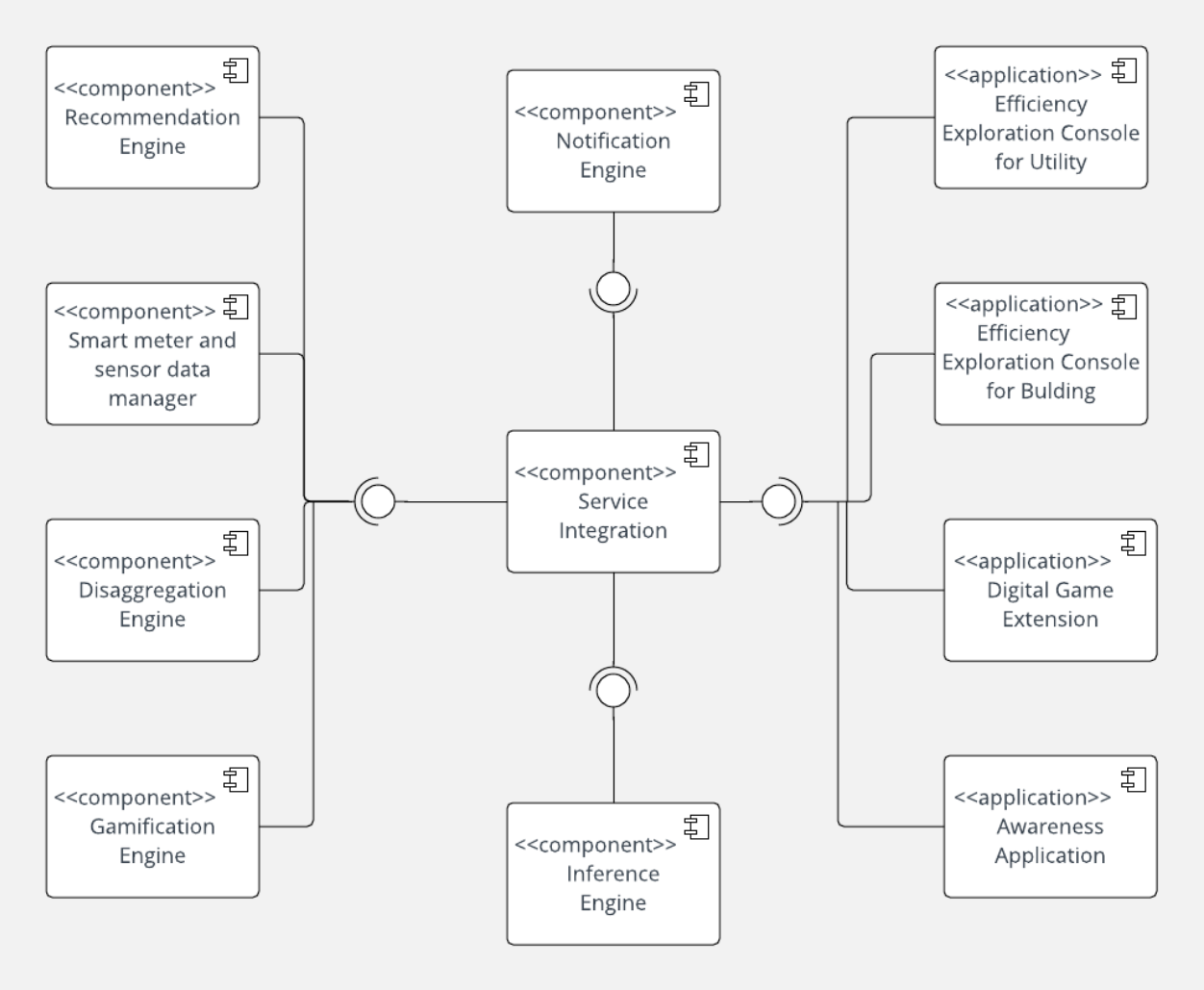

Figure 2 - Platform Component View

Each component will be detailed in the following chapters.

3.2 SERVICE INTEGRATION COMPONENT

This component manages all interaction between Platform components and between Platform component and Central Database. All interactions will be done only through Web Services that will be enlisted in this component.

This component will be implemented using an Open Service Framework (OSF). This architecture design decision also ensures the integration with 3rd parties as stated in DoW: "*allow easy application development by third parties following Platform as a Service (PaaS) and Software as a Service (SaaS) models*".

The supporting technology for this component will be made of following Open Source frameworks and tools:

- Spring Boot(Pivotal Software, SpringBoot). Java based framework that simplifies development of Java based Web Services;
- Hibernate ORM (RedHat, Hibernate ORM). Java based framework that implements data handling and persistence;
- Maven (Apache, Maven). Java tool that automates application packaging and deployment;
- Swagger (SmartBear Software, Swagger). Open Source tool that automatically generates documentation for developed web services.

This component will provide each Partner with a toolkit that will ensure service development in a consistent manner across the Platform. The toolkit will allow each Partner to build 2 types of services:

1. **Data access service**. This type of service will handle simple or complex queries and manipulation from of the data from Central Database. These services will build the Data Object Layer of the Platform.

2. **Wrapper service**. This type of service will allow interaction between components. These services will build the Business Object Layer. Every component that needs to expose component specific data and/or logic to other components will build a wrapper service with the provided toolkit. Service consumers will call the wrapper service of the OSF component which will send the request.

3.3 Smart Meter and Sensor Data Manager

This component manages the consumption and sensor data of the Platform.

The process of data provisioning of the enCOMPASS Platform is achieved by Smart Meter and Sensor Data Management (SMSDM) component. This component performs process periodic (hourly, daily, weekly, monthly) smart meter and sensor readings provided as CSV files by the Energy Utility through SFTP.

The SMSDM component:

- Retrieves user consumption data from smart meters for past time intervals;
- Retrieves data from sensors (presence, luminosity, temperature, humidity, etc.) for past time intervals.

Energy Utility will automatically prepare CSV files containing:

- energy consumption data;
- sensor data.

The files will be uploaded via FTP to a transfer location for further processing by this Platform component. This component is also structured in sub-components as presented below.

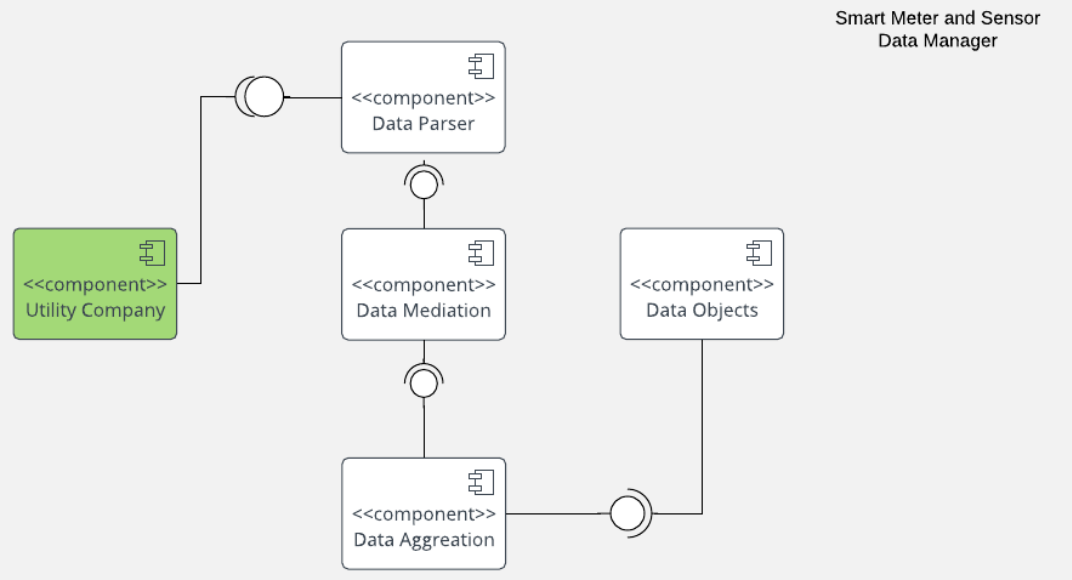

Figure 3 - Smart Meter and Sensor Data Manager Component View

Data Parser

Handles the parsing of raw consumption and sensor data files that were previously uploaded by Utility company through FTP. An integrity check is performed for each file against the associated MD5 signature file.

This subcomponent loads in memory the consumption and sensor file and pass the structured data to Data Mediation component.

After file processing, this component performs file archival.

Data Mediation

Performs operations like data cleaning and data normalization.

Energy consumption information can be expressed as consumption at time intervals:

- Time interval of measuring. Start Time / End Time, Start Time / Offset...;
- Device ID;
- Consumed quantity.

or can be expressed as meter counter at different moments in time:

- Timestamp;
- Device ID;
- Meter counter.

In the first case, the Mediator should validate if time intervals for the same Device ID does not overlap. If time intervals overlap then data correction or data cleaning algorithms will be performed.

In the second case, the Mediator should validate if Meter counter is increasing in accordance to time series. If some abnormal variations of meter counter are detected data correction or data cleaning algorithms will be performed.

Data Aggregator

This component performs various aggregation on the clean (mediated) data:

- aggregated consumption at time period level: day/week/month;
- aggregated consumption at device/household level.

Data Objects

This component manages the data regarding energy consumption and sensor data of the Platform. It services internally Data Mediation and Data Aggregator components and other Platform components requests for data:

- GET/PUT Energy consumption data;
- GET/PUT Sensor data;

The actual data access is done via the Service Integration component.

3.4 Awareness Applications

This component is a mobile and web application that allow Platform user to improve their energy consumption efficiency. This component provides to End User the following main functionalities:

- view consumption data;
- play social games GWAP (Games with a Purpose);
- receive consumption tips;
- receive recommendations.

The Awareness Application interacts directly with the Gamification Engine to retrieve Gamification Data and with the Service Integration component for consumption data.

3.5 Efficiency Exploration Console for Utilities

This component is a simple web application that allows Utility managers to analyze aggregated consumption and sensor data of all the users.

This component retrieves consumption and sensor data information from Service Integration component at different aggregation levels.

3.6 Gamification Engine

The Gamification Engine is a component that listens to the actions of the users and transforms them into a variety of rewards, for improving activity and participation.

The definition of points, achievements and rewards allow energy utility companies to challenge their users with energy saving goals to be achieved each month to obtain individual energy consumption and data provisioning. The **Gamification Engine** is also used as a mean for raising energy consumption awareness by promoting sustainable behavior.

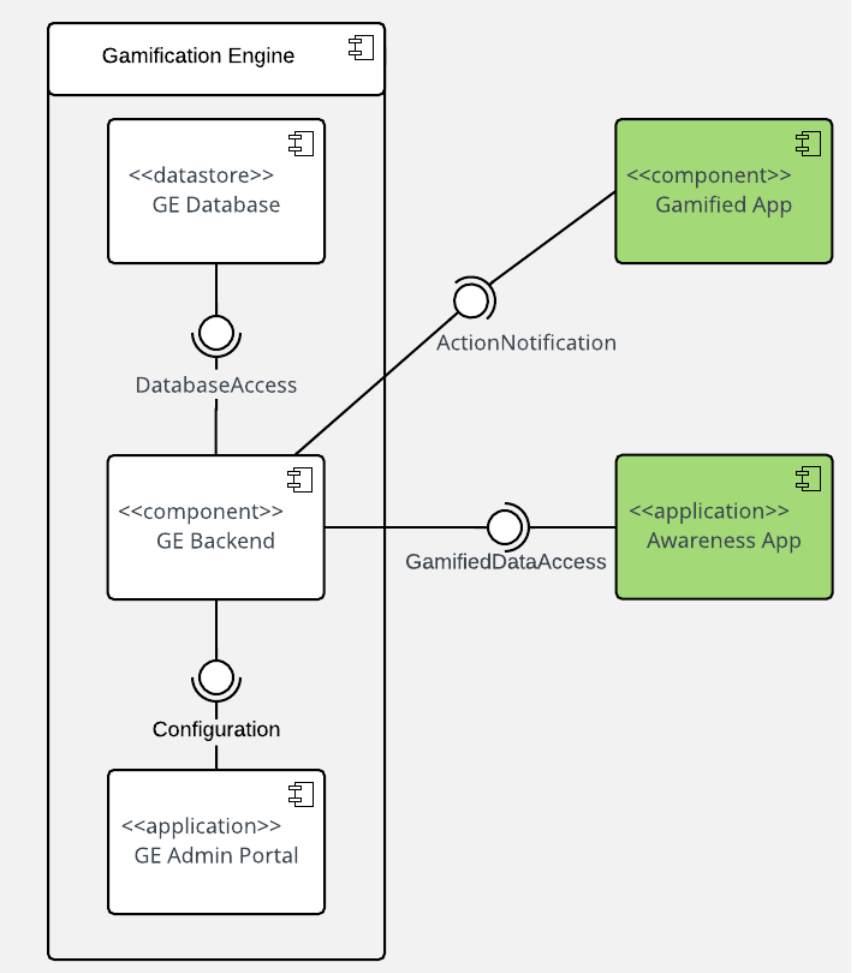

Figure 4 - Gamification Engine Component View

The Gamification Engine includes three components:

- **Gamification Engine DB:** stores all the gamified data, which adhere to the conceptual and logical schema described in the Information View of this deliverable. The Gamification Engine DB serves the purpose to decouple the data from the various energy utilities portals with the one managed by enCOMPASS for the gamification requirements;
- The **Gamification Engine Backend** is the central component that handles the communication with the main enCOMPASS platform components and takes care of orchestrating the activities for registering users, converting users' actions in other components into gamified actions for enCOMPASS in order to compute scores, badges, achievements and other gamified features and being able to track and profile the users. Its services rely on the data managed in the Gamification Engine DB, which the Gamification Engine Backend queries and updates;
- The **Gamification Engine Admin App** is a special purpose interface for configuring the elements of the GE: it serves the purpose of making the gamification process flexible and dynamically tailored to the rapid changes requested by the scenarios and the community at hand; for these reasons, new goals, rewards and achievement can be added in the platform by non-technical users through the Gamification Engine Admin App, which stores the configuration and gamification rule parameters into the Gamification Engine DB, with the mediation of the Gamification Engine Backend.

The Gamification Engine mainly interacts with:

- The **Awareness Application** extend the features of the energy utility customer portals, in order to let the users of this group access their gamified profile, the points collected, the achievements obtained, and redeem the earned rewards. To this purpose, the Awareness Application must interact with the Gamification Engine Backend, in two ways:
    - To extract the gamification data necessary to populate the Awareness Application user interface with the elements of the gamification (achievements, points, badges, rewards, etc.);
    - To insert into the Gamification Engine DB, with the mediation of the Gamification Engine Backend, the traces of the specific actions performed in the Awareness Application by the user, in order to have such gamified actions being translated by the Gamification Engine into achievements, points, badges, rewards, etc.
- The **Gamified Apps** are a third kind of client to the Gamification Engine Backend; they generalize the notion of an external application where the user can make actions related to energy sustainability, which must the processed and rewarded according to the rules implemented in the Gamification Engine. An example of such a Gamified App can be a pre-existing (web or mobile) application by the utility company, e.g., for billing and customer management. The interface of the Gamification Engine Backend towards the Gamified Apps allows the app to:
    - To query the gamification engine for the data necessary to display the elements of the gamification (achievements, points, badges, rewards, etc.);
    - To insert into the Gamification Engine DB, with the mediation of the Gamification Engine Backend, the traces of the specific actions performed in the Gamified app by the user, in order to have such gamified actions being translated by the Gamification Engine into achievements, points, badges, rewards, etc.

3.7 Recommendation Engine

This component produces action recommendations based on disaggregated consumption and sensor data. Proposed recommendations will be sent to Central database to be further sent to End User by Notification Engine.

The Recommendation Engine ingests the following data available in the Platform:

- consumption and sensor data;
- user profile data;
- building information;
- user actions on the Platform including user responses to Recommendation Engine suggestions.

The main purpose of the Recommendation Engine is to give the users suggestions on energy saving and lowering energy consumption based on the data available in the platform. By responding to these suggestions, the user can further enhance later recommendations.

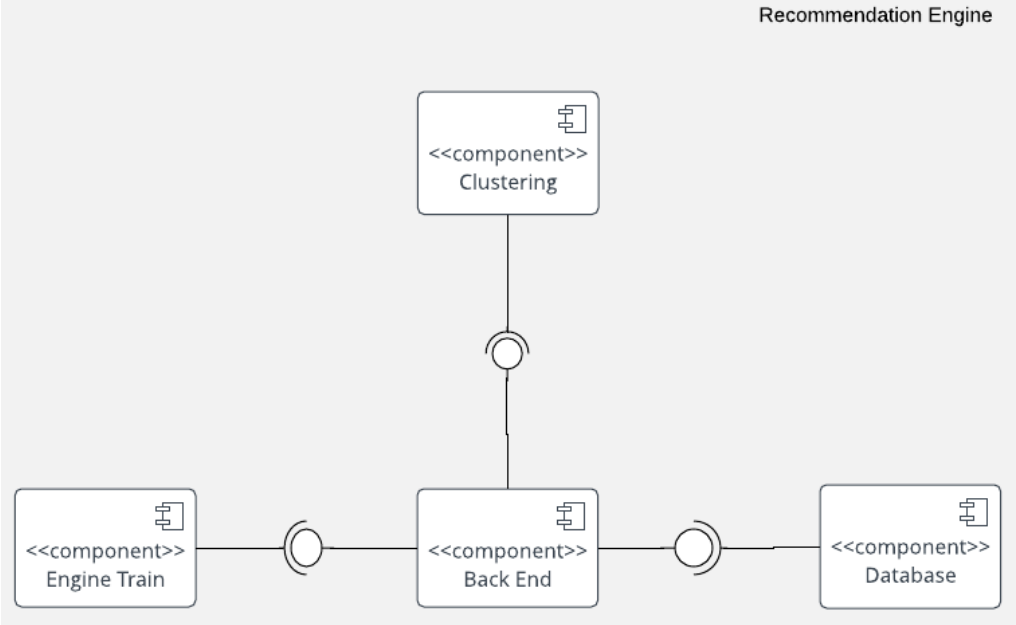

Figure 5 - Recommendation Engine Component View

This component includes:

- **Database** component stores the parsed data from the Platform, and models, which are used in creating the recommendation;
- **Backend** component is responsible for calculating the recommendation based on the rules and models. These recommendations will be sent to Platform User by the Notification Engine to the Awareness App;
- **Engine Train** component is responsible for calculating the models based on the parsed data (user profile, building information, user actions, etc.), that was ingested by the Recommendation Engine;
- **Clustering** component is responsible for identification of users with similar consumption and behaviour patterns and creation of such groups - clusters. This component is used to create recommendations that are relevant to a specific user cluster.

3.8 Inference Engine

Occupancy and Activity Inference Tool

The Occupancy and Activity Inference Tool is responsible for estimating the number of occupants (or presence/ absence) and the activity performed in a building (e.g. cooking, resting, sleeping, etc.).

The engine takes as input the raw data from the sensors, i.e. the energy consumption and the indoor environmental conditions, and the disaggregated information (i.e. more detailed information regarding the energy consumption in a building). All these data are processed utilizing advanced analysis and classification techniques, towards extracting the number of occupants and/or presence/ absence in a building. The extracted number of occupants is stored in the database along with the reference timestamp.

On the same basis, the engine utilizes the raw and disaggregated information in order to inference the activities in a building, e.g. cooking, resting, sleeping, etc.

User Comfort Levels Inference Tool

The Comfort Levels Inference Engine is responsible for estimating the comfort of the occupants in a building.

The engine takes as input the raw data from the sensors, i.e. the indoor environmental condition such as temperature, luminance, etc., and the disaggregated information (i.e. detailed information regarding the energy consumption in a building). All these data are processed utilizing advanced analysis and classification techniques, towards extracting the comfort levels of the users/ occupants in a building. The comfort is divided into two main classes, the visual and the thermal comfort. The visual comfort is referred to the perception of the users regarding the luminance in a building. On the other hand, the thermal comfort is related to the human perception regarding the sense of indoor temperature.

The thermal comfort is based on the Mean Predicted Values (PMV) and its output is a number ranging from [-3, 3] (very cold to very hot). The visual comfort is based on the Kruithof diagram[1], while the output is normalized to [-3, 3] range, in order to be conformed to the thermal comfort levels. All the extracted values are stored in the database along with the reference timestamp, for further use by the recommendation engine and other enCOMPASS components.

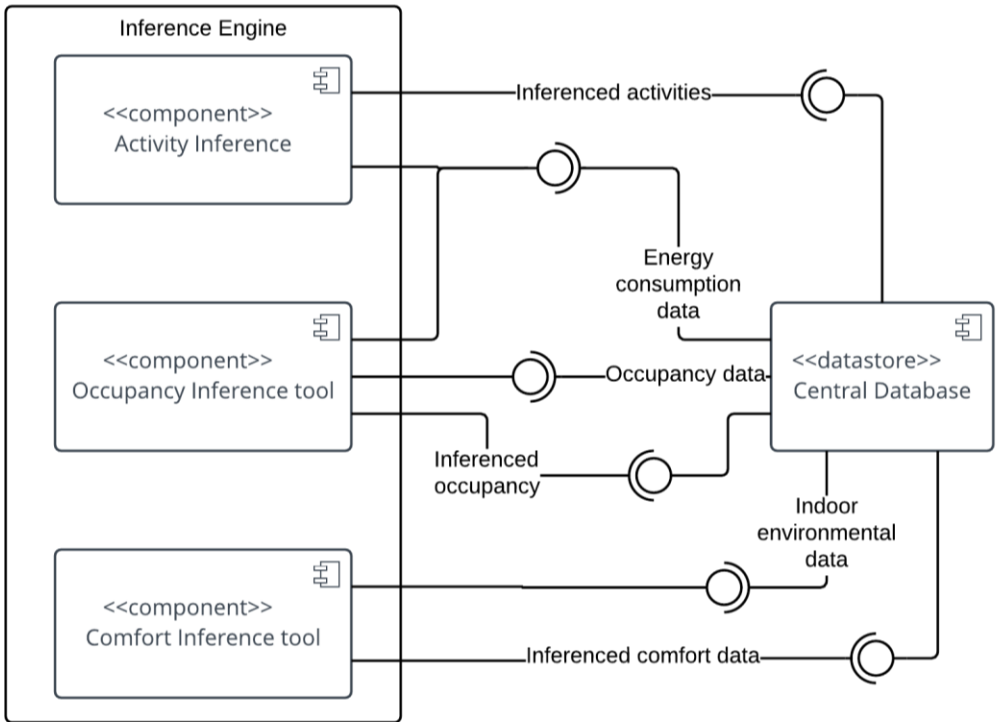

Figure 6 - Inference Engine Component View

3.9 Efficiency Exploration Console for Building Managers

The Efficiency Exploration Console is a web platform for Building Managers (BM) responsible for visualizing all the information that is related to the building.

The Efficiency Exploration Console is a dynamic platform allowing users to have multiple dashboards. Thus, a BM is able to create its own dashboards and to modify existing ones. The BM can have multiple

[1] https://en.wikipedia.org/wiki/Kruithof_curve

dashboards illustrating each of them a different perspective of the building, e.g. one for the current status, one for the aggregated status of the previous month, one for the aggregated status of the current year, another one for comparison purposes, etc. It depends on the imagination and the needs of the BM. It also supports multiple different widgets (e.g. plots, bars, pies, maps, donuts, bubbles, polar, spider, etc.), allowing the BM to select the ones that more fits to him/her. All these widgets can be parameterized so as to illustrated specific data for a specific period. Also, parameters such as color, position, size, etc., can be parameterized as well.

The Efficiency Exploration Console retrieves all the stored information from the enCOMPASS database, and provides though its dynamic dashboards to the BMs. The admin of the system just provides the credentials to the BM and assign to him/her specific properties, such as building ownership, which determines the information that he/she is able to visualize.

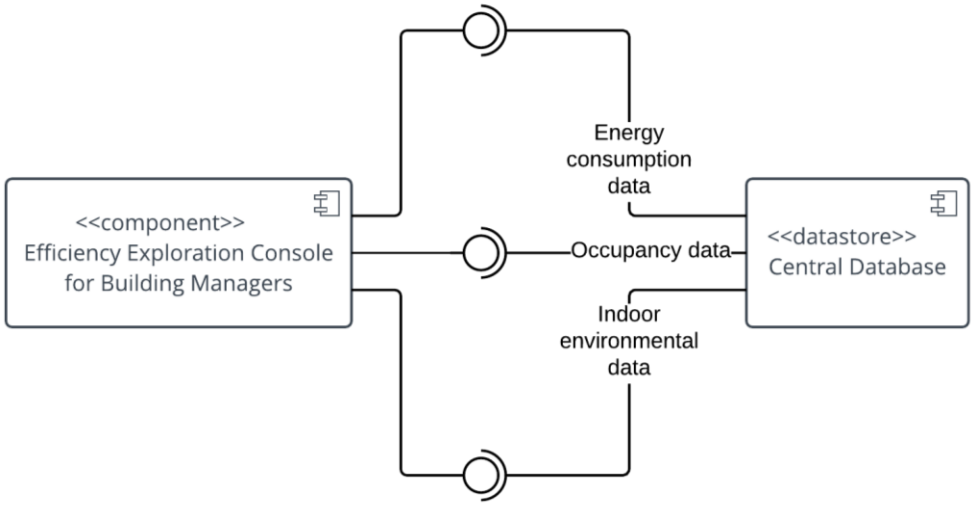

Figure 7 - Efficiency Exploration Console for Building Managers Component View

3.10 Disaggregation Engine

The Disaggregation engine processes aggregated consumption data and it returns the estimated end-uses of the single devices in a household.

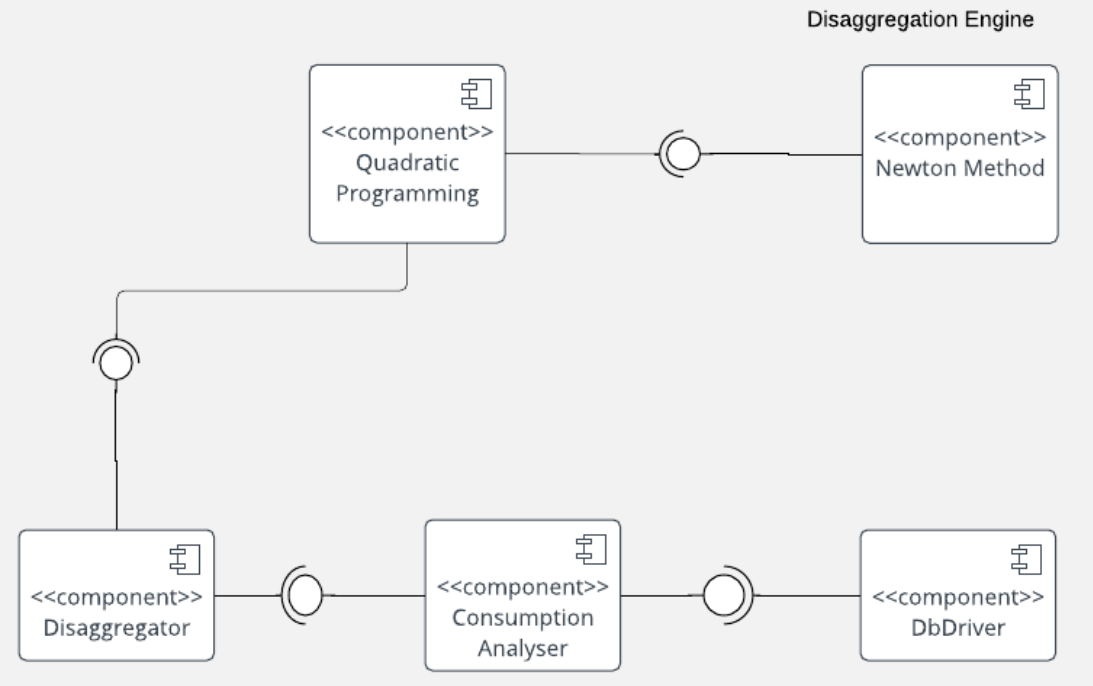

Figure 8 - Disaggregation Engine Component Diagram

To take advantage of multicore hardware solutions, the disaggregation algorithm is subdivided in multiple threads by the ConsumptionAnalyzer Class.

We now shortly describe the classes we implemented. The class diagram is provided in Figure 8.

- **ConsumptionAnalyzer**: Class that contains the main functionalities: training and disaggregation. The training computes the weight explained before and through the DbDriver it stores them into the database;
- **QuadraticProgramming interface**: allows to change solver without need of modifying the other classes;
- **Disaggregator**: Class that creates all the matrices and launches the solver for the sub-problems given. This Class extends from *Callable* in order to run as task, allowing for an optimal multi-threads scaling. The disaggregation method loads the weights and the optimization parameters from the database (through the DbDriver) and it decomposes the disaggregation in multiple tasks (Disaggregator instances). Once all the tasks finish, it sums up all the partial results in order to get the final disaggregation result;
- **NewtonMethod**: Implementation of the Newton's optimization method, using logarithmic barriers for inequalities constraints and the Backtracking line search;
- **DbDriver**: Class that offers all the interaction methods with the enCOMPASS database.

Moreover, there are three supporting classes not listed in the above diagram:

- **PropertiesLoader**: Class that loads the database information properties: username, password, server name, port number and database name;
- **Array**: Utility Class containing append functions and conversion between Arrays and ArrayLists;
- **Matrix**: Utility Class containing linear algebra for matrices and functionalities to add one or more rows to an existing matrix.

3.11 Digital Game Extension

Gaming is one of the techniques used in enCOMPASS to convey the sustainable energy consumption message. A board/card game is designed (Funergy) to let users approach energy sustainability in their school and family contexts, also independently of the use of the enCOMPASS digital platform. The introduction non-digital games in the enCOMPASS project derives from the need of attracting the interests of a broad audience of users, who are not necessarily linked with energy utilities adopting the enCOMPASS Awareness application approach; the link to the digital world must nonetheless be established for those users that are both players of the Funergy real game and users of the enCOMPASS digital platforms and its Awareness applications. This is the role of the **Digital Game Extension (DGE)**, illustrated in figure below, which handles the communication between the enCOMPASS platform and the digital game that complements the card/board game and takes care of managing users' registration, profiling them, orchestrating and keeping track of game instances and the gameplay sessions of the players. A dedicated database of the Digital Game Extension takes care of storing the users and their details, gameplay session's data, point and achievements obtained within the games platform, so to serve the needs of both casual players and of players registered to the enCOMPASS digital platforms.

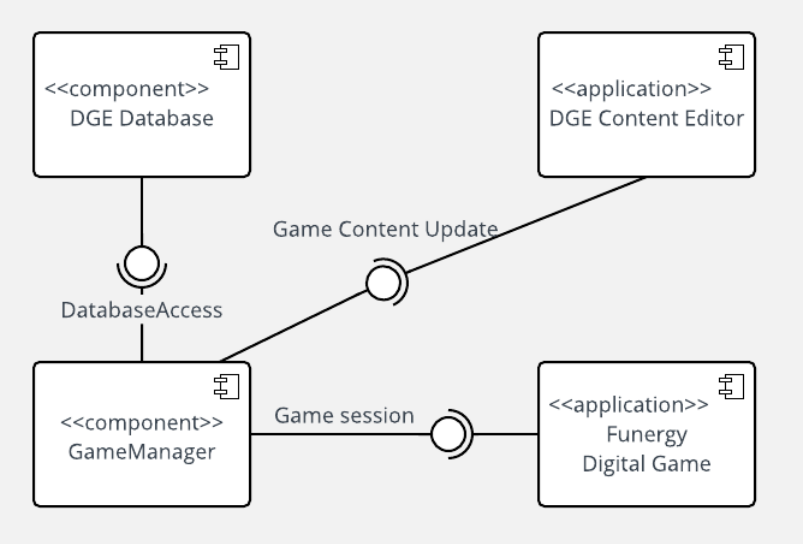

Figure 9 - Digital Extension Component View

The component is exploited to implement a Quiz Game, internally called **Funergy Digital Game**, which is the digital extension of the **Funergy** real game and is used to improve the awareness of users.

The Digital Game extension comprises / interacts with, the following components:

- **Digital Game Extension (DGE) DB** stores all the gameplay data, which adhere to the conceptual and logical schema described in the Information View of this deliverable. The DGE DB serves the purpose to host both casual players and players registered into the enCOMPASS platform and to decouple the data of the various energy utilities portals from the one managed by enCOMPASS for addressing the gaming requirements;
- The **Digital Game Extension Backend** is the central component that handles the communication with the Funergy Game App to provide questions and save answers;
- The **Funergy Game** extends the Funergy board/card game with a digital appendix. It interacts with the DGE Backend, in two ways:

- To extract the game content and the game play statistics data necessary to populate the game user interface with the elements of the game (e.g., trivia quizzes, difficulty levels, correct responses, points, players' skill level and achievements, etc.);
- To communicate the players' actions to the DGE Backend;

- The **DGE Content Editor** is a special purpose interface for the definition of content for the games (e.g. the text of energy related quizzes and of their responses, the names and icons of game badges and achievements, the communication messages in the game play, etc.); the DGE Content editor UI that allows also non-technical users to enrich the content offered by the game without the need of modifying the underlying architecture or create new builds.

3.12 Notification Engine

This component is responsible for sending PUSH notifications to End User Client application, like Awareness Application. It reads messages to be sent from the Central Database and queues them for sending. Any Platform component that needs to send a message needs to store the message in a message table specifying the destination application (endpoint), message content and the schedule.

The entities that are involved in notification sending are:

- **External Entities**. Entities that Notification Engine interacts with:
  - **Active Object** – Application or component originating the notifications. They will store the notifications in the central DB. A notification message will contain: destination application, the sending schedule, the message content;
  - **Client App** – The destination application that will receive the notification;
- **Internal Entities.** Components that builds the Notification Engine.

The component diagram of Notification Engine is presented below.

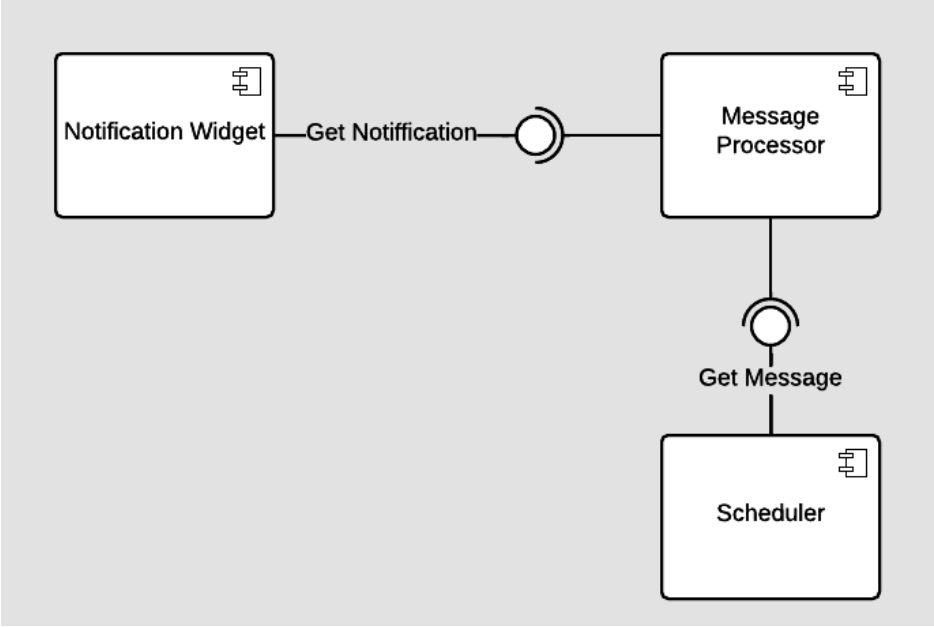

Figure 10 - Notification Engine component view

- **Notification Widget** – A client component embedded in the Awareness Application and Efficiency Exploration Console for Utilities that will actually receive the Notification PUSHED by Message

Processor. It will allow displaying of messages in the container Client App. The message may contain a link to a page in the Awareness Application and the End User can access the link and open that page in the Awareness Application;

- **Scheduler** – This component loads a Notification Queue from the database with the messages to be sent according to their schedule;
- **Message Processor** – A worker component that delivers messages from the Notification Queue according to its schedule. The states of the worker process are presented in the table below

Notification state	**Description**
Dormant	The notification task is set up.
Ready	The notification task awakes.
Pre-empted	When running, the task is pre-empted.
Delayed	The notification task is waiting for a signal or a resource.
Running	A processor is assigned to the notification task.

Table 1 - States of Message Processor

Notification Engine component can function as a **reusable component** using different data sources for Client application and Destination application.

4 INFORMATION VIEW

4.1 PLATFORM DATABASE

Platform data will be organized in 2 main data categories:

- **Shared data.** Data that is shared among more components of the platform, e.g. user data, consumption and sensor data. This data is accessed via the Service Integration component;
- **Specific data**. Data that is specific to only one component and it serves the business logic of that specific component, e.g. gamification engine specific data, recommendation engine specific data, etc.

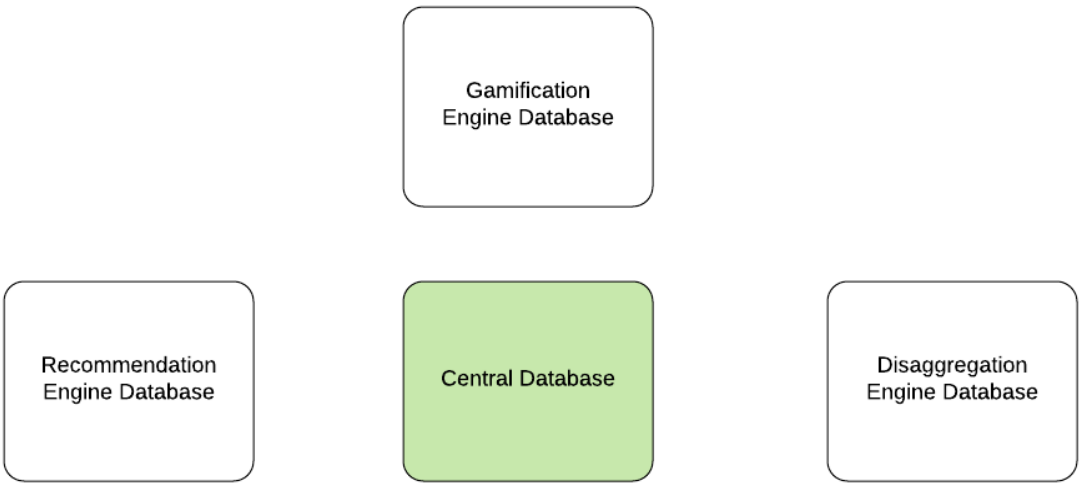

Figure 11 - Platform Data Model Overview

4.2 CENTRAL DATABASE

The figure below depicts the main entities of Platform central database. The level of detail is kept at architectural level with the purpose of identifying main entities and relationships among them. As mentioned also in the Executive Summary, a technical description of this database can be found in deliverable D3.1

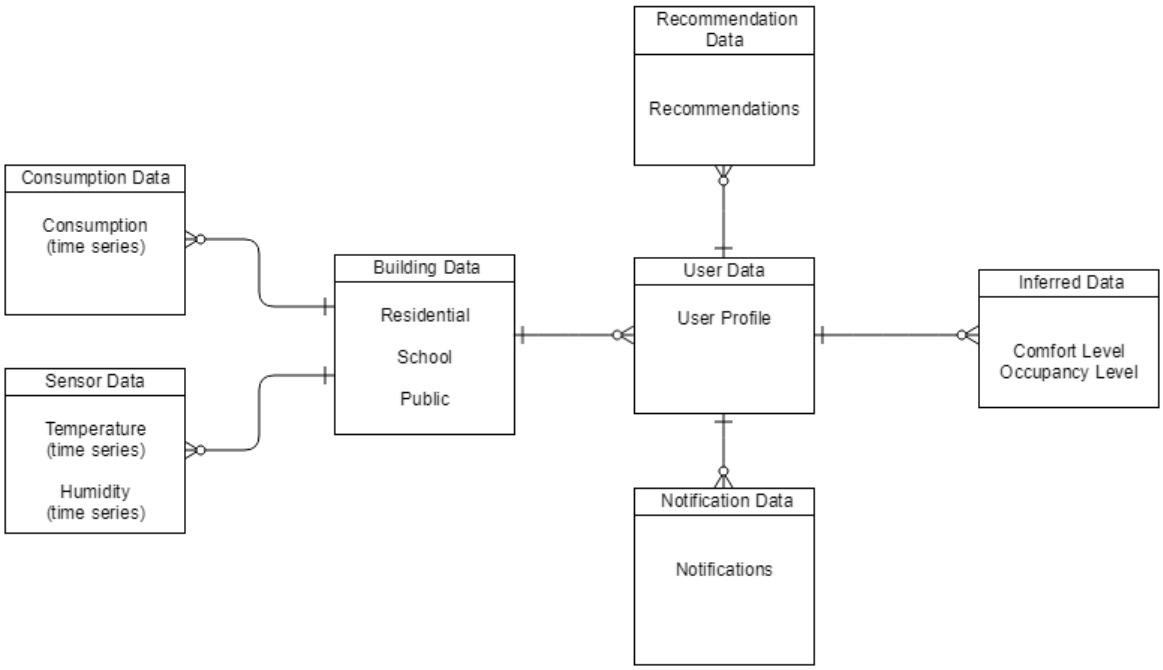

Figure 12 - Central Database Data Model

Consumption Data will store data regarding energy consumption for each user and meter device.

Sensor Data will store various environmental information: like temperature, motion and luminosity. Such data will serve to infer the current user comfort level to detect user presence. This data will serve as input to Recommendation engine and Data Disaggregation Engine and Data Inference Engine components.

Building Data will store information like: Geolocation, installed Devices, halls, rooms, location characteristics (surface, volume). This data will link Users that register in the Platform to Consumption Data.

User Profile will store End Users (Residential, School, Public Building) that registered to Platform, Utility Users, Building Managers, Platform Administrators. Information like Identification information: Name, Email, Password, Building / Apartment, Profile: Date of Birth, Gender, Occupation, Is Subscriber for Notifications will be stored.

Recommendation Data will store the recommendations produced by Recommendation Engine and sent by Notification Engine.

Notification Data will store all the notifications produced by Gamification Engine, Recommendation Engine and other components that were sent to user. Notification data will be produced by the Notification Engine after the successful sending of a notification.

Inferred Data will store the results of data disaggregation and inference processes. These processes will produce indicators like: Comfort Level and Occupancy Level

This database will serve as the data source for Client Apps like: Awareness Application, Efficiency Exploration Console for Building Managers but also for Inference Engine component from the Business Layer. These components will access data using services implemented in Service Integration component.

4.3 GAMIFICATION ENGINE

The **Gamification Engine** is a back-end component that embodies rules for transforming users' actions into gamification scores and achievements. It is exploited in order to "gamify" the energy consumption of the users, according to the awareness approach implemented by enCOMPASS. It has an interface for the end-user, who appraises the results of her energy consumption actions; and an administrative interface for the utility operators, who can supervise the outcome of the awareness policies and define the rules that reward the actions of energy consumers.

The schema of the information managed by the gamification engine comprises the:

- The **Gamification engine data model**, which describes the entities and relationships necessary to represent the users of awareness applications extended with gamification features;

The schema in figure below shows the conceptual model of the gamification engine database. The following entities have been considered:

- **Community User**s: the entity is a specialization of **User**, that identifies the user as a member of a community (like profile pic, bio information, etc.) and summarizes the points accumulated as result of the actions perform in the platform;
- **Action Type:** the entity contains the dictionary of the actions of the gamification engine. The attribute values of an action are the specific features of the considered action;
- **Action Instance:** the entity stores all the action instances performed by users;
- **Badge Type:** the entity contains the dictionary of the badges that a user can acquire;
- **Badge Instance:** the entity contains all the badge instances acquired by the user;
- **Reward Type:** the entity contains the dictionary of the rewards;
- **Reward Instance:** the entity contains the instances of the rewards acquired by the users;
- **Text Mail:** the entity contains information about the notification to send to users after a particular event in the gamification engine (e.g. a user gains a badge);
- **Notification**: this entity contains the notification sent to users;

In order to allow the Content Editor of the Gamification Engine portal to organize actions and badges according to topics, the thematic area entity has been introduced:

- **Thematic Area**: each thematic area is identified by a unique id and a name. Each action or badge belongs to a specific thematic area;

In order to enable user to compete with themselves and obtain points by achieving saving objectives, the entity goal has been introduced:

- **Goal**: this entity contains the consumption goals assigned to users. Each goal is identified by a unique id, a title, a consumption value, and optionally the completion date. Goals can be assigned to a given user, and can be associated to a Badge Type, obtained by the user when the consumption goal is achieved.

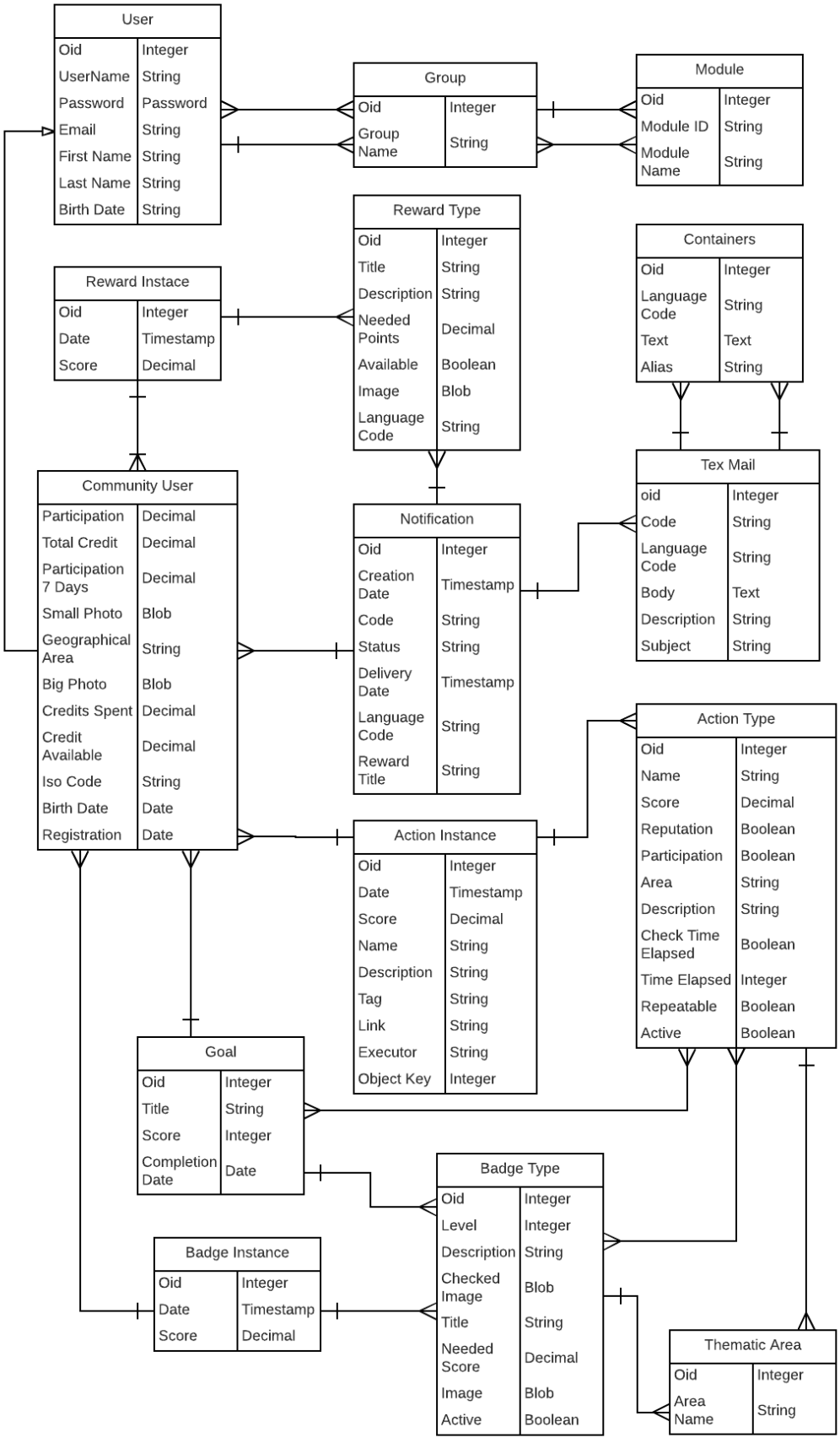

Figure 13 - Gamification Engine Data Model

4.4 DIGITAL GAME EXTENSION

The **Digital Game Extension** supports the execution of the digital games of enCOMPASS, including the game played as part of the interaction with the Funergy board game. In principle, extension could also be used by casual players, and thus there is an independent users' registration procedure. The Digital Game Extension exposes two kinds of interfaces: a digital game directed to the end users; an administrative interface, directed to the content editors of the Digital Game Extension. The GUIs are served by a dedicated local database (the Games DB), which stores information that is pertinent only to the game play (e.g., the gaming history of players).

The Digital Game Extension data models expands the Gamification Engine data model with the representation of additional entities and relationships that capture the essential data about the users who play with the enCOMPASS digital games.

The Entity Relationship Schema is represented in figure below.

- **Game** is the core entity: the *Mode* attribute represents the gameplay modes (e.g. Single Player, Multi Player, Cooperative), while the *Genre* attribute identifies its genre (e.g. Puzzle, Educational). Each game is also characterized by a *Title*, a *Theme* and the *Minimum/Maximum number of players*;
- An **Achievement** has an *Icon*, which describes it in a visual way, a *Category* that specifies the task (Instructor, Grinder), an attribute *PointsGiven*, which contains the amount of points to be granted, and a Boolean attribute *OfTheDay* defining whether the achievement has to be completed on a specific day in order to obtain virtual goods, more points, or increased levels;
- The **Player** entity accommodates game-specific personal and social features. *Avatar* and *Nickname* allow the user to be recognizable by using a custom image or a unique fictional name, while *Player Type*, *Player Level* and *Experience Points* convey player progress. *Reputation* in online gaming communities is fundamental and distinctive feature of any player; being able to recognize wheter a player is bad mannered, prone to cheating, unpleasant to play with is of utterly importance to assure a satisfying gaming experience for the user of an entertainment platform; it is usually measured as an integer number ranging from 0 to 5;

The model describes also the game-relevant statistics (**Game Stats**): the proficiency and the experience of a player in a given game are represented by aggregating in a compact way such indicators as points gathered and hours spent playing.

- **GameBadges** represent the achievements that have been unlocked by a player. The *CompletionPercentage* field shows how much the player has already achieved in a specific task. *StartDate* and *EndDate* record the dates in which the player has started to work on the achievement's goals and the date in which he has obtained it. The *TrialsN* attribute tracks how many times the user tried to fulfill the achievement;
- A **GamePlay Action** of a player, associated with a specific **Gameplay**, records the *StartDate* and *EndDate* of the gaming session and the actual actions performed by the player on that specific time frame and the *Role* defines which are the allowed actions in the game for the role associated to a player;

In order to store questions and answers required by the Funergy trivia game, **Question** and **Answer** entities have been provided. **Question Instance** keeps track of players game play information related to the specific quiz game.

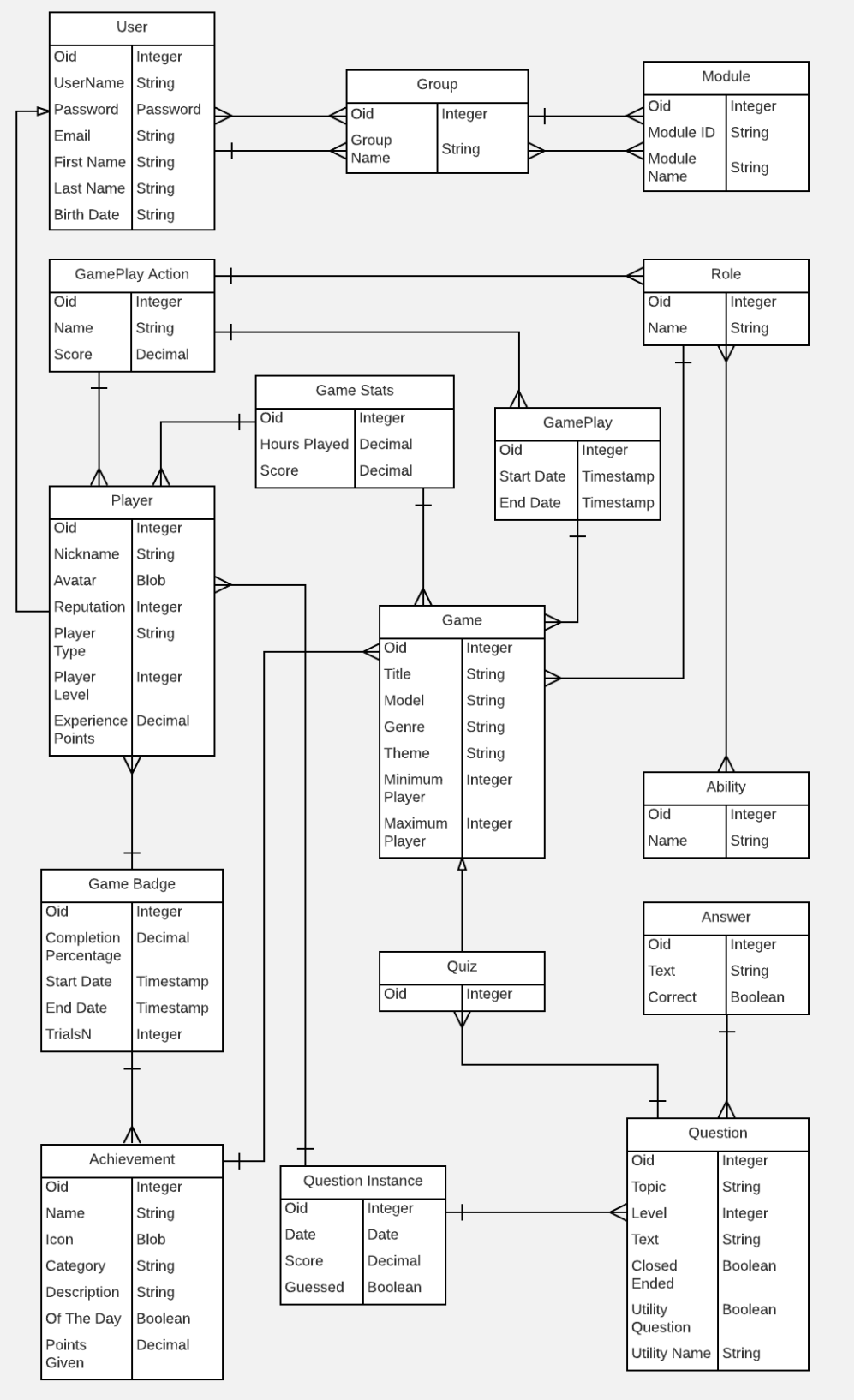

Figure 14 - Digital Extension Data Model

4.5 RECOMMENDATION ENGINE

The Recommendation Engine is the back-end component responsible for creating energy consumption awareness recommendations for individual users based on their user profile, building information, consumption and sensor data, and user actions on the enCOMPASS Platform. It has no direct interface towards then end-user, only through the AwarnessApp, and the Notification Engine.

Because Input data used by the Recommendation Engine is provided by other components, these data models will be further explained in the respective component's part.

The schema used by the data model of the Recommendation engine comprises following entities:

- **Users** entity will store anonymised user ID's. This entity will only contain Platform's internal User ID;
- **Recommendations** entity will store generated recommendations by the component;
- **Clusters** entity will store identified user clusters. These clusters will store group of users with similar consumption and behaviour pattern. One user can be included in more than one cluster. One cluster will include more than one user.

The relationship between data entities is presented below.

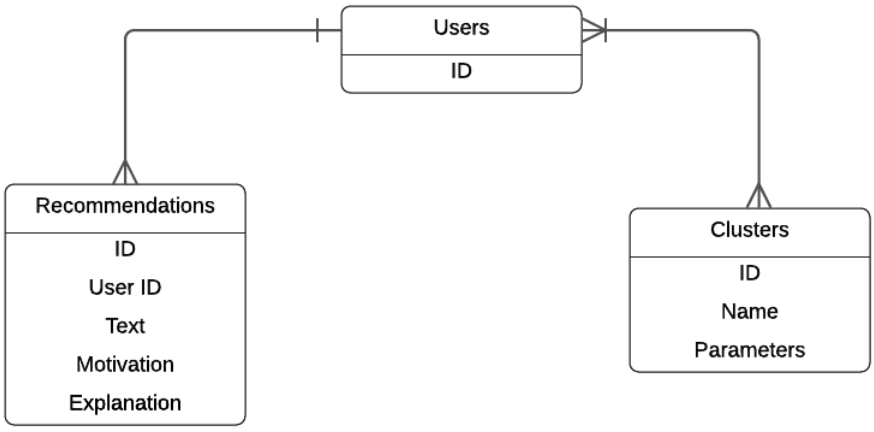

Figure 15 - Recommendation Engine Data Model

4.6 DISAGGREGATION ENGINE

A data model for the disaggregation engine is reported in figure below the Disaggregation Algorithm (DisaggregationAlgo), it is characterized by training (trainingParams) and optimization parameters (optimParams) and the weights used in the objective function (lambda). A detailed description of the role of the parameters is contained in Deliverable D3.3. The algorithm also needs to know how many appliances are present in the house, and for each appliance it needs an operational base (applianceBase) that is the power consumption in each operational state. The algorithm returns the appliance states (applianceStates) that describe if an appliance is on or off at any given time interval.

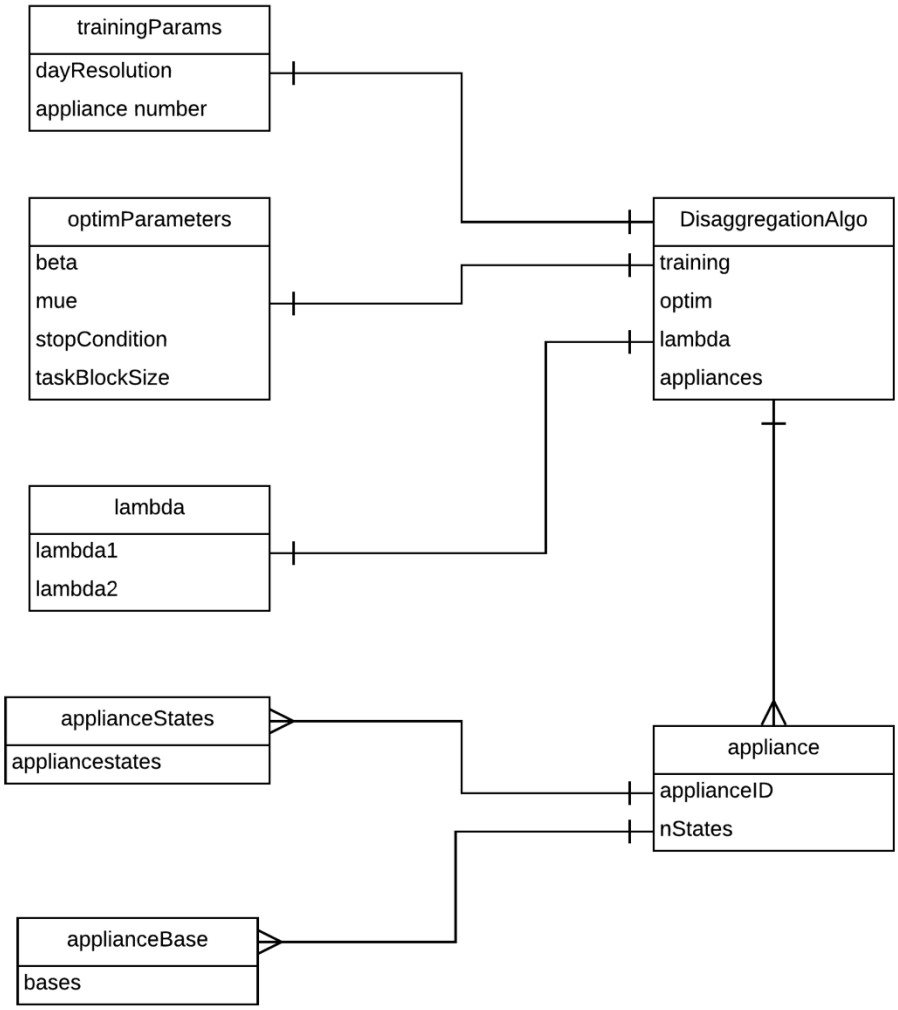

Figure 16 - Disaggregation Engine Data Model

5 PROCESS VIEW

5.1 COMPONENT INTEGRATION

The integration between Platform components will be done via REST Web Services via the Service Integration component.

Each Partner will develop its own services that will follow a unified development framework and will enlist developed services in Service Integration Component. Other components will consume such services by invoking these enlisted services.

5.2 USE CASE - COMPONENT MAPPING

The role of each component for each Use Case will be codified as:

P – Provider

C – Consumer

Platform components codification

DM – Smart Meter and Sensor Data Manager

SI – Service Integration

GE – Gamification Engine

RE – Recommendation Engine

DE – Disaggregation Engine

IE – Inference Engine

NE – Notification Engine

DG – Digital Game Extension

EC – Efficiency Exploration Console for Utility

EB – Efficiency Exploration Console for Building

AA – Awareness Application

In the table below we enlisted all Use Cases from D2.2 - deliverable, identified Platform components that are involved in the Use Case and the role, Publisher and/or Consumer, the component has in a particular Use Case. This mapping table has several purposes:

- Ensure that all Use Cases can be realized by Platform components
- Serve as a basis to design the services each component must implement

The table below enlist Use Cases triggered and performed by End Users in client applications: Awareness Application, Efficiency Exploration Consoles, Digital Game Extension

The table below also contains the technical Use Cases. These Use Cases are initiated and performed by back-end Platform component and they are not part of End User interaction. Technical Use Cases are related to following system flows:

- Energy consumption and sensor data acquisition;
- Consumption and sensor data disaggregation;
- Inference of disaggregated data to produce Comfort Level and Occupancy Level indicators;
- Generation of recommendations for energy saving.

Use Case	DM	SI	GE	RE	NE	DE	IE	EC	EB	AA	DG
Awareness Application - Signing up to enCOMPASS											
Register on the enCOMPASS platform		P	P							C	
Complete questionnaire on awareness measures	-	-	-	-	-	-	-	-	-	-	-
Get an introduction to the main features			P							C	
Modify user settings			P		P					C	
Awareness Application - Completing the User Profile											
Contribute user profile information		P								C	
Awareness Application - Visual exploration of energy consumption information											
View energy consumption in overview visualization		P								C	
View energy consumption impact		P								C	
Compare energy consumption to energy saving goal		P	P							C	
Modify energy saving goal			P							C	
View detailed energy consumption visualization		P								C	
View disaggregated energy consumption		P								C	
Awareness Application - Receiving static energy saving tips											
Browse through tips to save energy		P								C	
Awareness Application - Receiving energy saving recommendations											
Receive notifications of context-aware recommendations					C					P	
Browse through context-aware recommendations		P								C	
Semi-automatic execution of recommendations	-	-	-	-	-	-	-	-	-	-	-
Awareness Application - Earning points, badges, rewards											

Use Case	DM	SI	GE	RE	NE	DE	IE	EC	EB	AA	DG
Earn points for portal actions			P							C	
Earn points for measured effects of user actions			P							C	
Receive individual badges			P							C	
Receive physical rewards			P							C	
Compare achievements to others on leaderboard			P							C	
Receive notifications about activity and gamification achievements					C					P	
View activity and gamification achievements			P							C	
Awareness Application - Viewing and adjusting the user's comfort level											
View inferred comfort level		P								C	
Provide explicit comfort level feedback		P								C	
Digital Game Extension - Playing the energy saving game with its digital extension											
Play board game	-	-	-	-	-	-	-	-	-	-	-
Play the board game with its digital extension											P/C
Notification Widget - Receiving notifications											
Delivering notifications					P					C	
Gamification Engine - Gamification Engine Setup											
Set thematic areas			P					C			
Set actions			P					C			
Set badges			P					C			
Set rewards			P					C			
Set energy saving goal			P					C			
Gamification Engine - Managing customer's gamification actions											
View gamification statistics			P/C								
View customer action and goal history, and consumption overview			P/C								
Assign monthly winner			P/C								
Efficiency Console for Utilities - Monitoring consumption behaviour and household characteristics											
View energy consumption and basic sensor		P						C			

Use Case	DM	SI	GE	RE	NE	DE	IE	EC	EB	AA	DG
data in overview visualization.											
View detailed energy consumption and sensor data visualization.		P						C			
View indoor climate and comfort level information		P						C			
Send notifications to utility customers					C			P			
Efficiency Console for Utilities - Signing-up to Energy Efficiency Console											
Login in the enCOMPASS Console platform		P							C		
Modify viewing windows options		P							C		
Efficiency Console for Utilities - Monitoring of end-users consumption data and visualization of insights											
View energy consumption and environmental values in overview visualization.		P							C		
View chosen energy saving recommendation		P							C		
View statistical analysis of real vs optimal consumption data		P							C		
Efficiency Console for Building Managers - Customize viewing data											
Represent data in multiple statistical graphs		P							C		
Customize data viewing options, based on specific details		P							C		
System Use Cases											
Processing of consumption data	P										
Processing of sensor data	P										
Generate user optimal consumption data		P		C							
Generate Disaggregated Data		P				C					
Generate Inferred Indicators		P				P	C				
Generate Recommendations		P		C							
Track and log user actions on the Awareness App			P							C	
Compute Energy saving Goals			C/P								

Table 2 - Use Case - Component mapping

5.3 SERVICES

Based on the mapping between Use Cases and Platform Components, we picked the occurrences of each component that have Publisher role in a Use Case. This component – use case mapping serves as a base to

build the minimum and complete set of services that a component should provide to other Platform components.

Component	Interaction requirement
DM – Smart Meter and Sensor Data Manager	View energy consumption and environmental values in overview visualization.
	Processing of consumption data
	Processing of sensor data
	Generate user optimal consumption data
SI – Service Integration	Register on the enCOMPASS platform
	Contribute user profile information
	View energy consumption in overview visualization
	View energy consumption impact
	Compare energy consumption to energy saving goal
	View detailed energy consumption visualization
	View disaggregated energy consumption
	Browse through tips to save energy
	Browse through context-aware recommendations
	View inferred comfort level
	Provide explicit comfort level feedback
	View energy consumption and basic sensor data in overview visualization.
	View detailed energy consumption and sensor data visualization.
	View indoor climate and comfort level information
	Login in the enCOMPASS Console platform
	Modify viewing windows options
	View energy consumption and environmental values in overview visualization.
	View chosen energy saving recommendation

Component	Interaction requirement
	View statistical analysis of real vs optimal consumption data
	Represent data in multiple statistical graphs
	Customize data viewing options, based on specific details
	Generate Disaggregated Data
	Generate Inferred Indicators
	Generate Recommendations
	View energy consumption and environmental values in overview visualization.
	Generate user optimal consumption data
GE – Gamification Engine	Register on the enCOMPASS platform
	Get an introduction to the main features
	Modify user settings
	Compare energy consumption to energy saving goal
	Modify energy saving goal
	Earn points for portal actions
	Earn points for measured effects of user actions
	Receive individual badges
	Receive physical rewards
	Compare achievements to others on leaderboard
	View activity and gamification achievements
	Set thematic areas
	Set actions
	Set badges
	Set rewards
	Set energy saving goal

Component	Interaction requirement
	View gamification statistics
	View customer action and goal history, and consumption overview
	Assign monthly winner
	Track and log user actions on the Awareness App
	Compute Energy saving Goals
DE - Disaggregation Engine	Disaggregated Data: get
NE – Notification Engine	Modify user settings
	Delivering notifications
AA – Awareness App **Actually the Publisher is the Notification Widget sub-component of Notification Engine that will be included in AA**	Receive notifications of context-aware recommendations
	Receive notifications about activity and gamification achievements
EC – Efficiency Console for Utilities **Actually the Publisher is the Notification Widget sub-component of Notification Engine that will be included in AA**	Send notifications to utility customers

Table 3 - Use Cases - Component Mapping according to D2.2 – Final Requirements

We can aggregate similar requirement to accomplish Use Cases into a list of Service Families:

Component	Service
DM – Smart Meter and Sensor Data Manager	Process consumption data
	Process sensor data file
SI – Service Integration	User: RegisterUser, Update User profile
	Comfort Level: save, get
	Occupancy Level: save, get

Component	Service
	Recommendations: save, get
	Notifications: save, get
	Consumption data: get detailed, aggregated energy consumption data
	Sensor data: get detailed, aggregated sensor data
	Inferred Data: save, get comfort Level, occupancy Level
	Optimal Consumption: save, get
GE – Gamification Engine	User: save, get settings
	UserFeatures: save, get
	UserSavingGoals: save, get
	UserPoints: save, get
	UserPhysicalRewards: saveinfo
	UserAchievements: get
	AdminActions: save, get
	AdminBadges: save, get
	AdminRewards: save, get
	AdminSavingGoals: save, get
	AdminStatistics: get gamification, consumption statistics
	AdminWinner: save, get
DE – Disaggregation Engine	Disaggregated Data: process data, get results
NE – Notification Engine	SendNotification
	Receive Notification **The Publisher is the Notification Widget sub-component of Notification Engine that will be included in AA and EC**

Table 4 – Services exposed by Platform components

The interaction between services will be done via Service Integration component. A typical interaction for 1 Provides – 1 Consumer scenario is presented below:

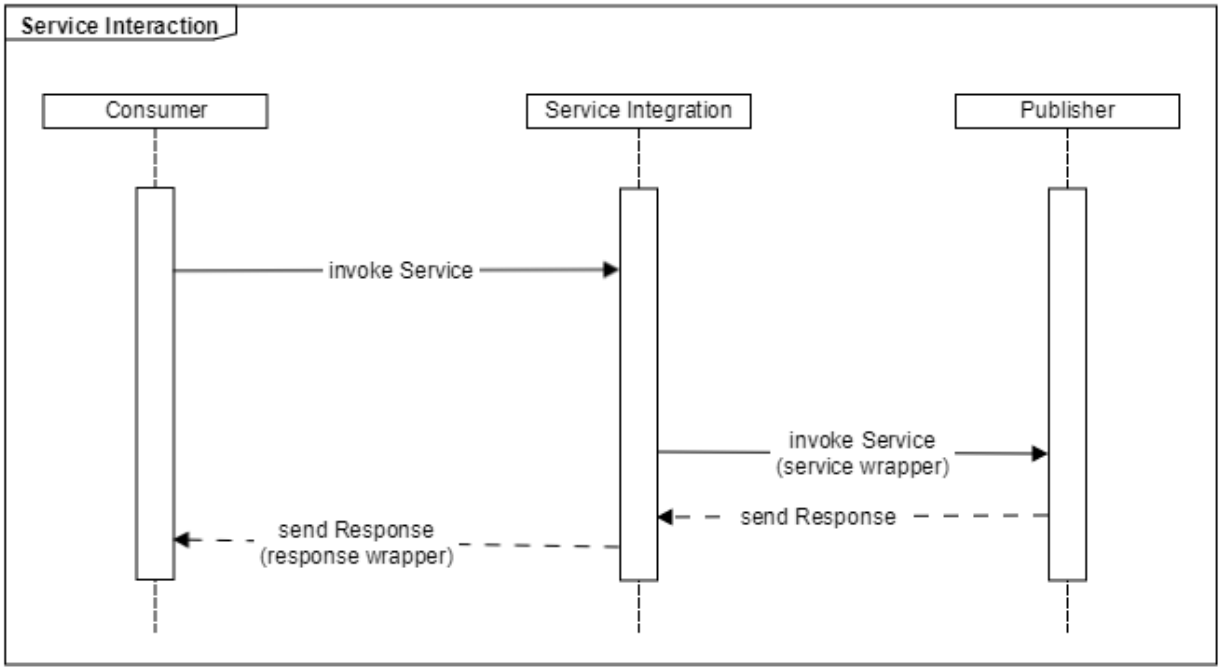

Figure 17 - 1 Provider - 1 Consumer Interaction Scenario

5.4 Service Integration

Based on the Use Case <-> Component Mapping one can identify the need for service integration whenever there are more than 2 Publisher components involved in the Use Case.

Use Case	DM	SI	GE	RE	NE	DE	IE	EC	EB	AA	DG
Compare energy consumption to energy saving goal		P	P							C	
View energy consumption and environmental values in overview visualization.	P	P							C		
Generate Inferred Indicators		P				P	C				
Generate Recommendations		P		C		P	P				

General service composition interaction diagram is presented below.

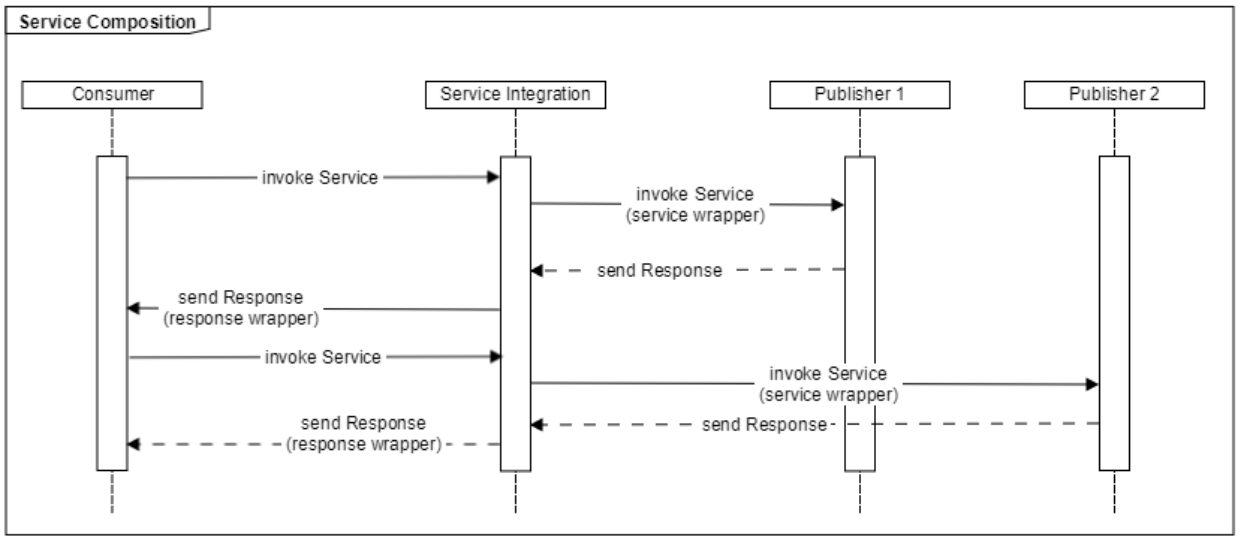

Figure 18 - Service composition diagram

In the Service Integration component will be implemented wrapper services that allows invocation of services by other client components. The actual service integration will be performed by the client component, which will need to wait / sync and compose the responses of more than 1 called service.

Still, for synchronisation of batch, long running processes like: data disaggregation, data inference, recommendation generation a different mechanism was designed. This mechanism will be semaphore based and it will be presented in the next chapter.

5.5 Batch Process Synchronization

There are long running processes like energy and sensor data acquisition, data disaggregation, data inference and production of recommendations that needs to run in a specific order. These processes start independently, they do not start each other, so an external synchronisation mechanism was designed. The design solution was a semaphore-like mechanism. This mechanism should stop a process from starting until the external semaphore is not in a state that allows him to start. Each process should update the state of the external semaphore.

The required running sequence for these processes is:

1. Energy consumption and sensor data acquisition. This process takes as input energy consumption and sensor data produces by smart meters and sensors. This process applies data cleaning and normalisation operations and save the resulted data in Central Database.
2. Data disaggregation. This process takes as input consumption data and based on mathematical models will allocate raw consumption data on various user actions: 10% heating, 30% cooking, 20% ironing;
3. Data inference. This process takes as input the disaggregated data produced by Disaggregation component and computes various inferences to assess the User Comfort Level and Occupancy and Activity Levels;
4. Recommendations. This process takes as input the disaggregated data and inferred data and produces energy saving recommendations.

The diagram below presents the synchronisation/semaphore-like mechanism:

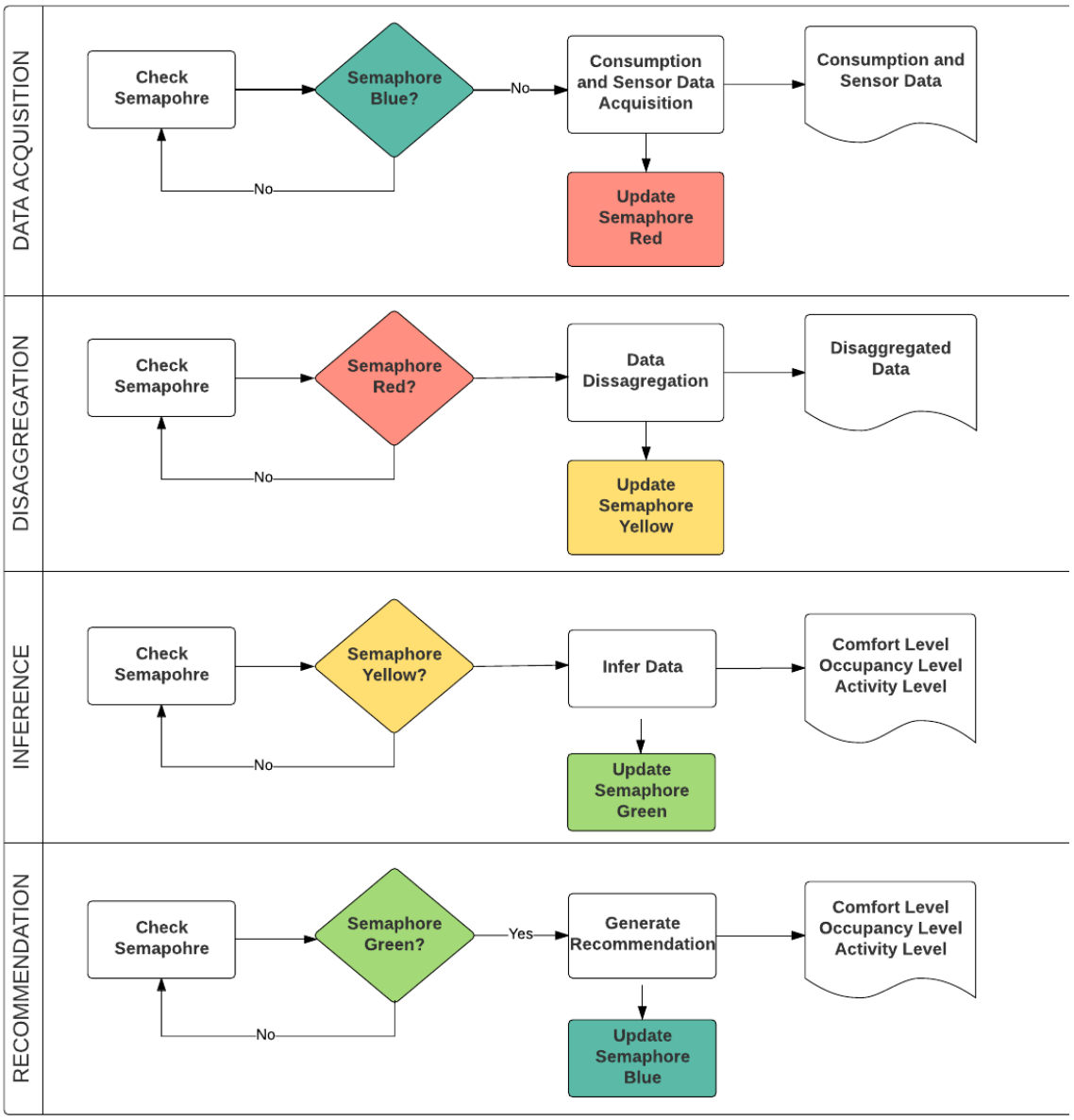

Figure 19 - Batch process synchronisation mechanism

Batch processing components will check the semaphore and will change its state through a service exposed by the Service Integration component. The semaphore state will be stored in the Central Database.

6 DEPLOYMENT VIEW

From the deployment point of view, we can qualify Platform components in following packages.

Application Server Package:

- Smart Meter and Sensor Manager;
- Service Integration;
- Gamification Engine;
- Notification Engine;
- Disaggregation Engine;
- Inference Engine.

Database Package:

- Central Database;
- Gamification Engine Database;
- Notification Engine Database;
- Disaggregation Engine Database;
- Inference Engine Database.

Recommendation Engine Package:

- Recommendation Engine.

Mobile Apps Package for iOS:

- Digital Extension for iOS;
- Awareness Application iOS wrapper.

Mobile Apps Package for Android:

- Digital Extension for Android;
- Awareness Application Android wrapper.

Note: The Recommendation Engine component will be deployed as a SaaS service and it will communicate via Internet with other Platform components.

For legal and data privacy reasons, most of the shared and specialized components packages will be deployed as a **distinct Platform instance at each Pilot Utility company premises.** There will be 4 deployed Platforms:

- **Development and Testing environment;**
- **WVT (Watt and Volt) Pilot;**
- **SHF (Stadtwerke Haßfurt) Pilot;**
- **SES (Società Electrica Sopracenerina) Pilot.**

The Mobile Apps Packages will be uploaded to Apple and Google applications store and it will be installed by the End User on their mobile phone according to their operating system iOS or Android from the appropriate apps store.

In order to minimize the administration and maintenance effort of Utility Partner, all on premise deployed components will be deployed on the same server.

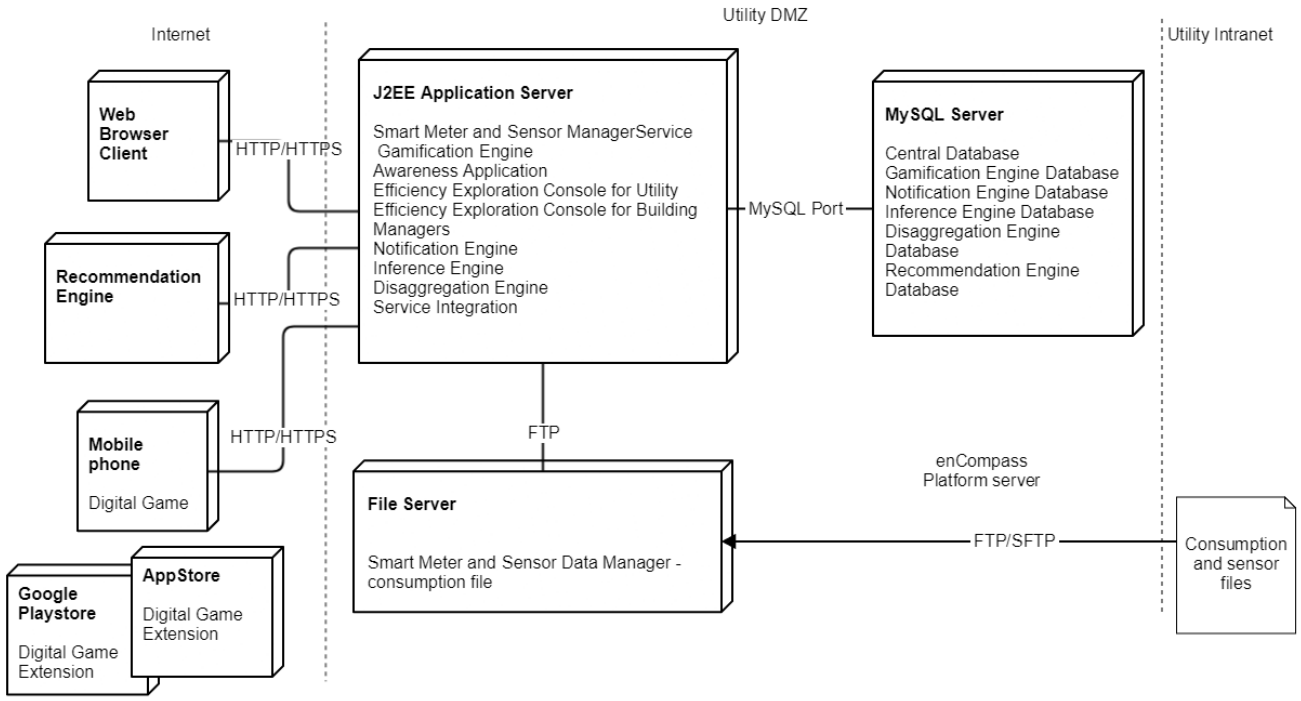

Figure 20 - Deployment View for one Utility Partner

Application Server software infrastructure requirements:

- MySQL Database Server;
- J2EE;
- Apache Tomcat;
- Java Spring;
- Maven;
- Hibernate ORM;
- HTTPS Server certificate.

Database Server software infrastructure requirements:

- MySQL Server.

In order to accurately reflect the production context, the same deployment architecture will also be used for Development and Testing server.

The initial deployment process to a Utility Partner will consist in following operations:

1. Take image of Development server;
2. Copy and restore image on Utility Partner server;
3. Modify environment configurations to match current context: IP addresses, ports, folders...;
4. Install HTTPS Server Certificate;
5. Make necessary setup in local network, DMZ, firewall to allow End Users to access component.

The upgrade process of will consist of a release package that will be installed by each Utility Partner. A release package may contain:

- software updates;
- database structure updates;
- data updates.

6.1 SECURITY

Server authentication. Platform Server will be authenticated to its Clients by installing a Server SSL Certificate and using HTTPS protocol for communication. This design solution will ensure both Server authentication and communication encryption.

Client authentication. End Users will be authenticated against the Platform in the Client Apps by user/password. The password will need to a certain degree of complexity that will be ensured via registration process.

Service authentication. On Premise components do not require authentication to access each other via Service Integration component as they are deployed on the same server, in the Utility network.

Recommendation Engine will implement a custom authentication solution in agreed with each Utility Partner. Different security solutions could be employed, after a common decision with each Utility Partner. Such solutions could be: VPN connection and O-Auth authentication will be implemented for every service exposed and consumed by the Recommendation Engine.

Database authentication. Service Integration component will access the Platform Database using user/password with a high degree of complexity.

REFERENCES

Apache, Maven. (n.d.). Retrieved from https://maven.apache.org/

IBM, Service Oriented Analysis and Design. (n.d.). Retrieved from https://www.ibm.com/developerworks/library/ws-soad1/index.html

Pivotal Software, SpringBoot. (n.d.). Retrieved from Spring: https://projects.spring.io/spring-boot/

RedHat, Hibernate ORM. (n.d.). Retrieved from http://hibernate.org/orm/

RM-ODP. (n.d.). Retrieved from http://www.rm-odp.net/

SmartBear Software, Swagger. (n.d.). Retrieved from https://swagger.io/

Zachman, J. A. (n.d.). Retrieved from https://www.zachman.com/about-the-zachman-framework