PROJECT DELIVERABLE REPORT

Innovative Technology for District Heating and Cooling
EE-13-2015 -Technology for district heating and cooling

Deliverable number	**D6.1**
Deliverable title	First Loop of Preliminary design
Submission month of deliverable	M26
Issuing partner	PROMAR
Contributing partners	NET, CERTH, CITY, CETRI
Dissemination Level (PU/PP/RE/CO):	PU
Project coordinator	**Prof Karcanias, CITY**
Tel:	**+44 (0) 20 7040 8125**
Fax:	**+44 (0) 20 7040 8568**
Email:	**N.Karcanias@city.ac.uk**
Project web site address	www.indeal-project.eu

Document Information

Filename(s)	InDeal_D6.1 _v06.docx
Owner	InDeal Consortium
Distribution/Access	InDeal Consortium, PO
Quality check	CITY
Report Status	Release

Revision History

Version	Date	Responsible	Description/Remarks/Reason for changes
01	2018/09/14	BM(PROMAR)	Draft
02.	2018/09/30	JS(PROMAR)	Draft reviewed
03	2018/10/04	BM(PROMAR)	Draft 2.0
04	2018/10/05	JS(PROMAR)and inputs for the rest contributing partners	Draft 2.0 reviewed
05	2018/12/30	BM(PROMAR)	Draf 2.0 completed with Figures description
06	2019/01/14	Review and preparation of the final draft (CITY)	Final draft released

Contents

Abbreviations

InDeal	Innovative Technology for District Heating and Cooling
CO	Confidential, only for members of the consortium (including the Commission Services)
EC	European Commission

1. Summary

The overall objective of Task 6.1 is to present a first loop for the preliminary design of the Indeal solution. An overview of all the system components is initially given along with a preliminary architecture of the integrated solution. Emphasis is given on the description of both hardware and software components as well as their interconnection. The specifications of the hardware components are given in Section 3. Section 4 describes the infrastructure installed at the substations and the communication technology that is required to establish a reliable link with the main server. Section 5 finally presents the characteristics and functionalities of the developed InDeal APIs that facilitate the access in the collected data and distribute them into the interested components.

2. Introduction

The main topic of deliverable 6.1 is the first loop for preliminary design for DHCS together with methods and modules for reliable data collection, processing and communication as well as seamless interconnection between the hardware and software components. The best practice techniques will be applied in order to demonstrate the overall solution with clear identification of innovation aspects and positive criteria for the end-users which should be analyzed and achieved. There are several subsystems which have been carefully specified in relation with one of the most important consideration which is the interoperability of each elements as part of the DHC network. These subsystems could be mainly separated in 4 basic categories – communication, control, remote communication and monitoring and hardware subsystem. Each of those have been defined based on set of criteria in such a way that mutual solution will offer innovative and improved realization for achieving optimal results. Following the specification of the main concept, a preliminary integration design has been developed to automate and streamline the complex process of collecting meter data from multiple meter data collection points, evaluate the quality of that data, achieve continuous and reliable data communication between the hardware components and the main control platform. The finalised integration design will be presented in D6.2 where the solutions found will be linked with the real case test sites which will be realized in Vransko (Slovenia) and Montpellier (France).

3. Overview of InDeal system components

3.1 InDeal system components

In general, two levels of the InDeal system have been taken into account in the integrated system design. First is the building side where all the installations and equipment devices are physically built-in. The second level is the Central Monitoring and Control Platform, which is a virtualized space where all data are processed, analyzed and visualized for the end-user. Interchange of information between those fields is provided by the InDeal-developed Advanced Data Management Modules (ADMM) units.

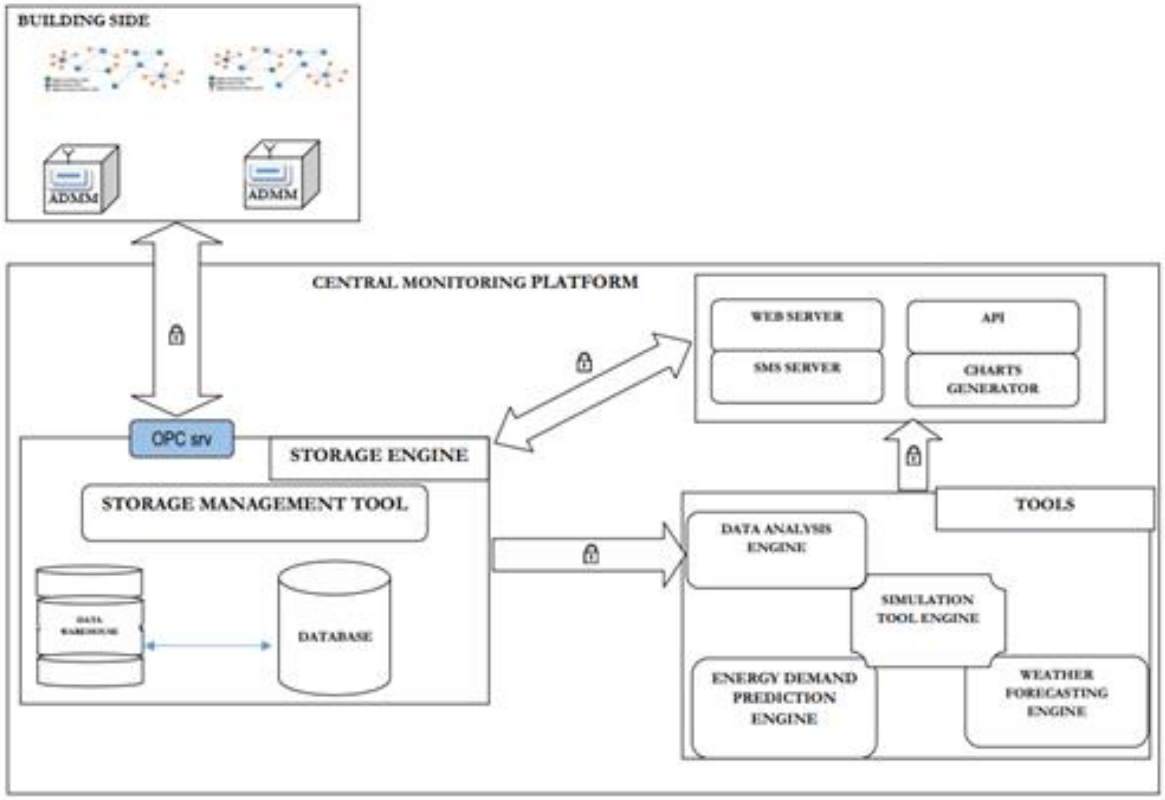


Figure 3.1.1 Overall architecture of Indeal solution

At the building side, the InDeal system is responsible for managing internal heating and cooling installations. On the one hand, ADMM transmits data to CMCP and is able to (i) receive commands and (ii) send to specific executive devices such as controllers, pumps, actuators, etc.

Due to the fact that the majority of the currently used devices on the market already have communication interfaces the idea is to manage as many installations devices as possible without the need to add new devices and thus increase the cost of implementation on the side of the building.

3.2 Advanced Data Management Module

The advanced data management module is a telemetric unit that provides data acquisition from the primary or secondary side installations and enables the remote change of parameters of the functioning side devices. It's connected directly or indirectly to meters, sensors and other automatic equipment and extends their functionality. An additional function of the device is that it extends the capabilities of the building devices by allowing continuous operation control and communication between devices that could not exchange data with each other so far. Using the implemented mechanisms ADMM is able to detect the incorrect (faulty) operation status of either the installation or the actuator or even single sensors.

Figure 3.2.1 ADMM overall data acquisition model

3.3 CMCP

The Central Control and Monitoring Platform (CMCP) is an information technology solution developed as a part of WP5, that offers web GUI for the end user. The platform can be accessed anytime, anywhere from different devices having internet connection. CMCP hosts several modules (tools) which can receive inputs from modules and visually present them in the defined web environment.

As stated above, CMCP has bidirectional communication for controlling and adaptation purposes. It can analyze and report (i) the measured, calculated and simulated parameters for energy consumption, production and storage, (ii) comfort requirements of end-users, (iii) meteorological conditions and (iv) costs in near real time. It can also (i) generate future predictions for the stochastic parameters of the system, (ii) apply the DSMPC algorithm, (iii) send remotely the control commands to the distributed field devices and (iv) send bidding decisions to the energy market to maximize the efficiency of energy transferring.

Detailed information with regard to different user interfaces, user equipment, components and subsystems of the platform are also provided in D5.3.

4. Building side specification

Energy is consumed by various types of buildings and installations. Some of them may be subject to easy-to-implement optimization or rationalization processes, while some require rather the application of technological improvements, but these are not in the field of interest of Indeal project.

4.1 Building types

For InDeal purposes, we have specified three main types of energy consuming building types.

· Residential buildings – where all the energy is used for the purpose of heating/cooling living spaces and preparing hot domestic water.

· Civic / commercial buildings (covering medical, educational, religious, government uses) – where the vast majority of energy is used for the purpose of heating/cooling large spaces such as offices, warehouses, theaters and preparing hot water.

· Industrial buildings - where (usually smaller) part of the energy is used for the purposes of heating / cooling (covering the heat loss of buildings), preparation of hot water for users needs, and part of the energy is used for the needs of technological processes from the specificity of a given plant or industrial line.

In the scope of InDeal project are all of these buildings and types of installation outside of those that provide energy for the needs of industrial technologies.

4.2 Typical equipment of an substation

Regardless of the specific conditions, substations are equipped with some unchangeable elements that perform specific functions and process data. Such elements are:

· PI Controller - is a control loop feedback mechanism widely used in industrial control systems and a variety of other applications requiring continuously modulated control. The controller continuously calculates an error value e(t) as the difference between a desired setpoint (SP) and a measured process variable (PV) and applies a correction based on proportional and integral terms. In DHC installations most popular are temperature regulators with weather compensation. In heating and cooling systems, the function of the controller is to maintain a required temperature of the medium in the system by means of a pump or control valve actuator.

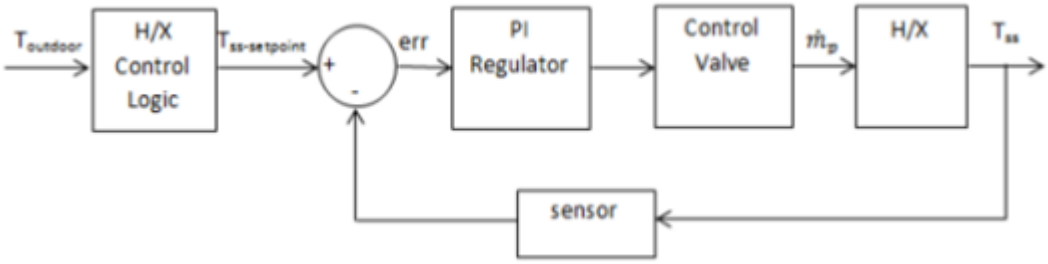


Figure 4.2.1 Scheme of a substation management process

.

Figure 4.2.2 Example of an automation device installed in substations

· Meter - is a device which measures thermal energy provided by a source, by measuring the flow rate of the heat transfer fluid and the change in its temperature (ΔT) between the outflow and return of the system. A heat meter consists of: fluid meter, pair of thermocouples and calculator.

Figure 4.2.2 Example of a heat/cold metering device installed in substations

· Sensor (detector) - is a device, module, or subsystem whose purpose is to detect events or changes in its environment and send the information to other device. Most sensors have a linear transfer function. The sensitivity is then defined as the ratio between the output signal and measured property. Other sensors detects the state change and sends output signal in binary mode (two state '0' or '1').

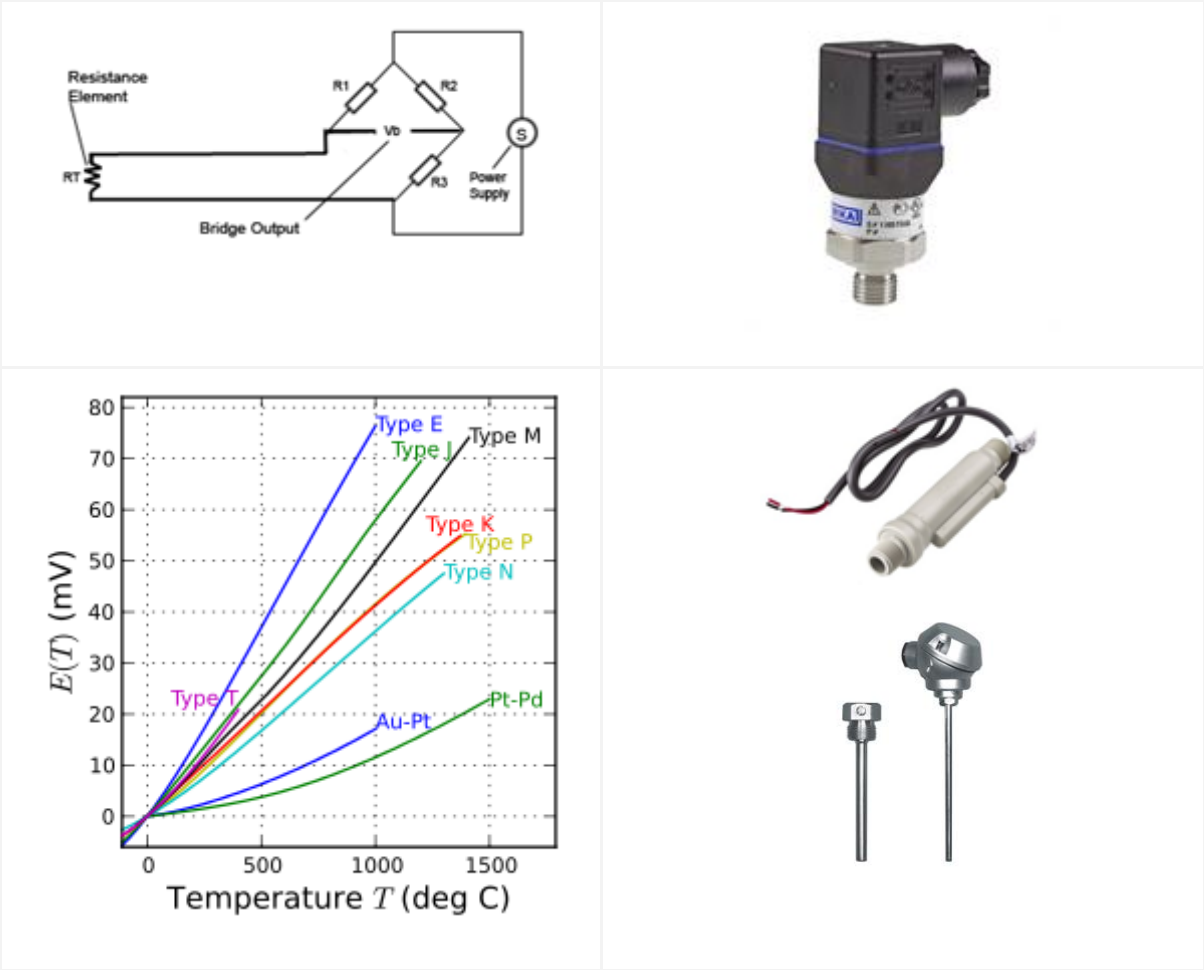


Figure 4.2.3 Example of sensors installed in substations

Control valve and actuator - is a valve used to control fluid flow by varying the size of the flow passage as directed by a signal from a controller. This enables the direct control of flow rate and the consequential control of process quantities such as pressure, temperature or liquid level.

Figure 4.2.4 Example of control valves with actuators installed in substations

· Variable-frequency driver - is a type of adjustable-speed drive used in electro-mechanical drive systems to control AC motor speed and torque by varying motor input frequency and voltage. That's an Rother way to control of flow rate and the consequential control of process.

Figure 4.2.5 Example of variable frequency driver installed in substations

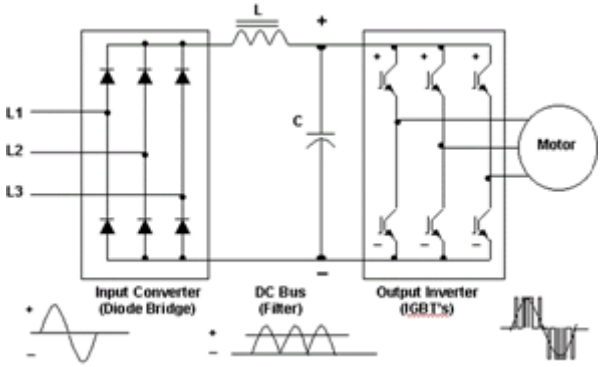


Figure 4.2.4 Scheme of variable frequency driver operation

All those devices have communication capabilities (send outputs to the control systems) with different standards. Some of them communicate using communication protocols others puts an electrical outputs (in this case we need to convert electrical signal to communication protocol using suitable converters).

4.3 ADMM data acquisition

Required communication protocols for ADMM

According to the market research done by PROMAR of metering and automation devices producers for DHCS it has been decided to use (embedded in the ADMM) MODBUS and MBUS protocols which is in line with industry standards. For the Indeal project it is required to communicate with :

- SAMSON automation devices with MODBUS protocol
- DANFOSS automation devices with MODBUS protocol
- Kamstrup heat/cold meters - with MBUS protocol

At the hardware level ADMM is required to have RS485 and RS232 electrical interfaces.

4.4 Communication between ADMM and server

It has been verified based on extensive experimentation and on our partners' expertise that internet connection is not available in substations for both real case studies. It is strongly required to establish a unified and reliable mean of communication. For this purpose, the consortium decided to use GSM network which is standardized for EU so one solution will be applicable around EU.

5. Integration methodology

This section presents a preliminary integration design for the software components (Figure 5.1). The functionality of the developed APIs along with the way that the different components use those APIs are given below.

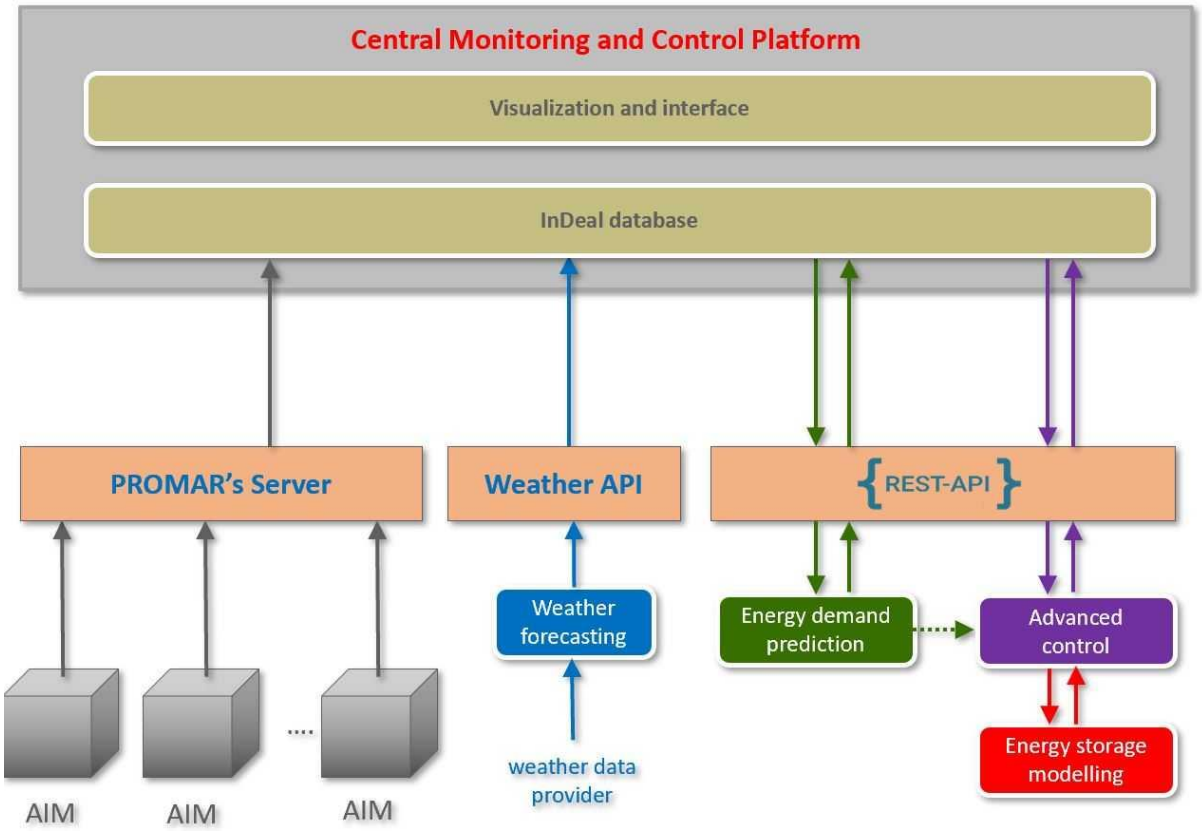


Figure 5.1 Graphical representation of the integrated InDeal system

5.1 APIs and their functionalities

A REST API is provided to facilitate the access in the collected data and distribute them into the interested components. The API is defined in php and 'acts' on top of the InDeal database. It contains all the necessary functionalities to support the retrieval of the sensor data. The provided functionalities deal with the access to one or more columns of the corresponding tables while the result is returned in a json format. The json format facilitates the transfer of data into the network increasing the efficiency. Json parsers and processors are adopted by any interested party that ants to have access on the sensor data. The call to the API is realized through a specific URL included in the interested components. Figure 5.1.1 presents the envisioned access in the database. Every client should instruct a request to the API and consume the provided json result. An example request is as follows:

http://5.44.241.82/api/read.php?city=vransko&sub=substation1.

The API receives the location and the name of the sensor for getting the corresponding data. The following lines depict the part of the code through which we are able to return the json object to the interested clients.

```
$query = "SELECT @n := @n + 1 Indx, DATE_FORMAT(date_time, '%d/%m/%Y %H:%i') as Dates, MIN(input_temp) AS inpTemp, MIN(output_temp) AS outTemp, MIN(diff_temp) AS diffTemp FROM $subName, (SELECT @n := 0) m WHERE status_code='!2' GROUP BY date_time";

$statement = $connection -> prepare($query);

$statement -> execute();

$data = $statement -> fetchAll(PDO::FETCH_OBJ);

echo json_encode($data);
```

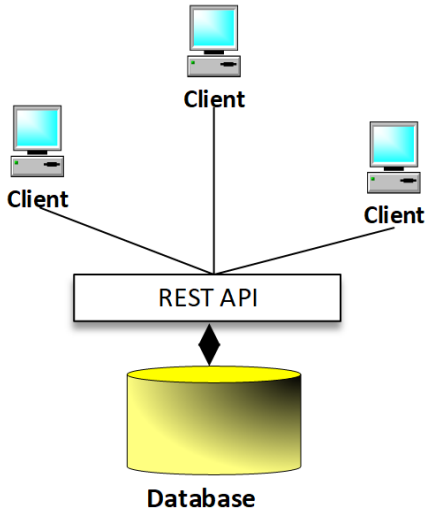


Figure 5.1.1. The envisioned access in the database.

5.2 Weather forecasting module integration

The aim of Task 4.3 is to develop an end-to-end automated simulation tool which will predict future temperature levels for the two locations under consideration, Vransko and Montpelier, to use in the energy demand forecasting tool. The process is nearly identical for the two locations, differing apparently on the names of the Tables that store the data, to respond to each specific location.

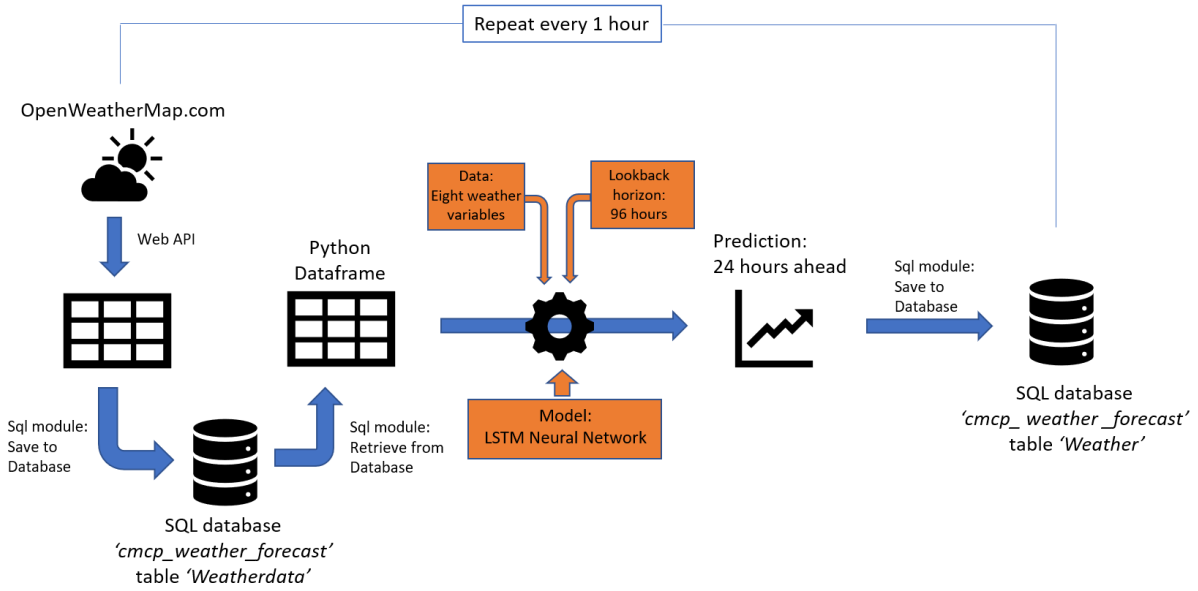


Figure 5.2.1 Weather forecasting pipeline

To collect weather data we use the web API Python library *pyowm* to connect to OpenWeatherMap.com, a website offering historical weather data. The data variables that we have selected to collect in order to build the predictive models are: Temperature, Humidity, Pressure, Precipitation, Snowfall, Cloudness, Wind Speed and Wind Direction.

We also mark the current timestamp at the time of data collection. Afterwards, we use Python's *mysql* module to store this data into the MySQL database 'cmcp_weather_forecast' and more specifically in Tables 'Weatherdata' plus the name of the location (thus, 'WeatherdataMontpelier' and 'WeatherdataVransko' for Montpelier and Vransko respectively). In these tables, each row contains the timestamp of data extraction and the eight measured variables on the specific timestamp. This procedure has been scheduled to repeat every one hour with the use of the online scheduler tool Heroku.

For the prediction phase, we need to ingest the relevant weather data into the developed weather forecast models in Python; i.e., Long Short-Term Memory (LSTM) Neural Networks. To perform this, we use Python's *mysql* module to connect to the MySQL database 'cmcp_weather_forecast' in which weather variables are being saved every one hour. We have configured our models to use the past 4 days' of data (96 hourly observations) to predict the temperature 24 hours into the future. Thus, we retrieve the last (at the time of retrieval) 96 rows of the 'Weatherdata' tables, containing nine columns each (timestamp plus eight input variables). We insert the input variables into the LSTM Neural Network and we forecast the temperature values for the next 24 hours in each location.

The last step of the process is to save the predicted temperature from the LSTM model. Having the SQL connection still open, we send back the model's output (24 temperature values) into the database 'cmcp_weather_forecast' but this time we save to different tables whose names are the concatenation of the word 'Weather' plus the location's name (thus, 'WeatherMontpelier' and 'WeatherVransko' for Montpelier and Vransko respectively).

```
SELECT * FROM cmcp_weather_forecast.weatherMontpellier;
```

Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

date	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12	t13	t14	t15	t16	t17	t18	t19	t20	t21	t22	t23	t24
2018-06-17 18:...	16.9...	18...	18...	20....	20.8...	21.3...	21.87...	21.7...	21.6...	21.90...	20.90...	21.16...	21.00...	18.24...	19.73...	18.97...	18.87...	19.94...	18.43...	18.61...	21.25...	21.65...	21.2...	21.8...
2018-06-17 19:...	16.0...	16...	18...	19....	20.5...	21.0...	21.47...	21.6...	21.9...	21.49...	21.91...	20.60...	21.17...	20.58...	18.31...	19.40...	18.80...	18.57...	20.43...	18.46...	18.78...	21.38...	22.1...	21.8...
2018-06-17 20:...	16.8...	17...	18...	19....	20.2...	21.2...	21.71...	21.3...	22.0...	21.80...	21.60...	21.65...	20.55...	21.15...	20.49...	18.66...	19.53...	19.21...	18.56...	20.71...	19.30...	19.04...	22.0...	22.2...
2018-06-17 20:...	16.5...	16...	17...	18....	19.8...	20.6...	21.73...	21.9...	21.6...	21.95...	21.78...	21.39...	21.75...	20.28...	21.39...	20.49...	18.46...	19.47...	19.00...	18.54...	20.75...	19.63...	18.7...	21.3...
2018-06-17 21:...	15.3...	16...	16...	17....	19.1...	19.9...	20.60...	21.3...	21.6...	21.21...	21.56...	20.82...	20.71...	20.96...	19.59...	20.84...	19.55...	17.52...	18.97...	18.38...	18.27...	20.18...	19.5...	18.3...
2018-06-17 22:...	14.8...	14...	16...	16....	17.9...	19.8...	20.55...	20.4...	21.8...	21.50...	21.18...	21.17...	20.42...	20.28...	20.77...	19.28...	20.48...	19.44...	16.37...	18.73...	18.49...	17.89...	20.1...	18.2...
2018-06-17 23:...	14.8...	14...	14...	16....	17.6...	18.6...	20.66...	20.8...	21.0...	21.66...	21.64...	20.99...	21.34...	20.21...	20.76...	20.64...	19.30...	20.18...	19.45...	16.40...	18.85...	18.93...	17.6...	20.0...

Figure 5.2.2 Temperature prediction table

As in the case of the weather data extraction from OpenWeatherData, in this case as well the process is fully automated, recurring every one hour with the help of the online Scheduler tool Heroku. This means that each hour a new line is created in 'Weatherdata' tables, containing the last 96 hours of data and a new line is created in tables 'Weather', containing the predictions with the next 24 hours' temperatures for each location.

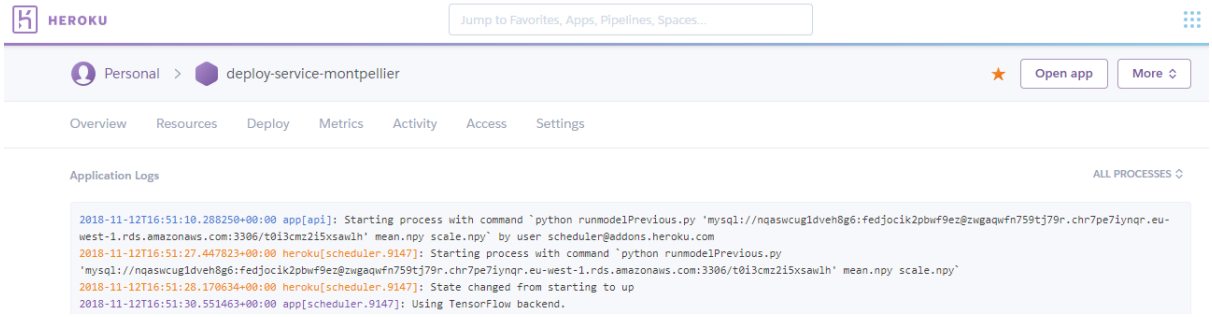


Figure 5.2.3 Online scheduler 'Heroku'

5.3 Energy demand prediction module integration

The aim of Task 4.4 is to develop an end-to-end automated simulation tool which will take advantage of the output from the weather modelling tool in Task 4.3 to predict future energy demand levels for the two locations under consideration, Vransko and Montpelier. To achieve this goal, we have used a wide stack of technologies that enable us to ingest data from different sources, make predictions using state-of-the-art Neural Networks, deliver predicted output to a centralized database and update the entire procedure in a pre-selected, recurring schedule.

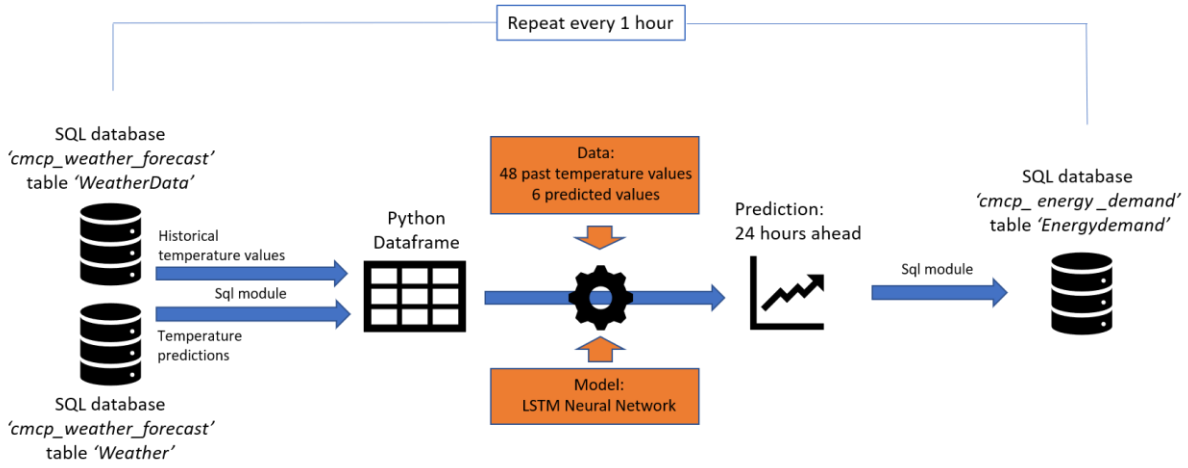


Figure 5.3.1 Demand forecasting pipeline

In order to develop suitable energy demand prediction models, we use a) past data and future predictions regarding the temperature and b) past data of the energy demand levels, in each location. These data do not reside into a single database thus we need two separate processes to unify these data before progressing further.

To collect temperature data we use Python's *mysql* module to connect to the MySQL database 'cmcp_weather_forecast'. In this database we then focus on two tables: Table 'Weatherdata' that contains historical temperature values at a specific timestamp and 'Weather' which contains temperature predictions for 24 hours from the particular timestamp onwards.

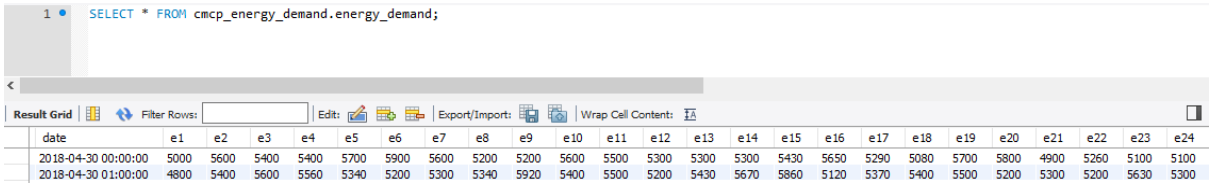
To collect energy demand data we use a provided REST API to connect to the database where each energy-measuring station is storing the realized energy demand information.

We concatenate the data from the two sources in a coherent table to insert into the Neural Network. We then call the saved model and perform the energy demand prediction. For the time-series prediction phase, state-of-the-art Long Short Term Memory (LSTM) Neural Networks have been selected. These models are capable of incorporating multiple variables in an auto-regressive fashion and are able of uncovering complex data patterns through a dynamic optimization process.

The prediction can be made for any time horizon, at the cost of increased error as the time horizon increases; we have selected a 24-hour horizon as the 'perfect balance' spot between model accuracy and information benefit for the DHCS. The backward-looking horizon can also be defined according to the user's goal. A longer horizon tends to lead to a better mapping of the given data. However, increasing the horizon beyond a certain point only adds noise to the modelling process and deteriorates performance. In this case, the backward-looking horizon has been set at 48 hours for which optimal forecast results have been achieved. Thus, we collect from the MySQL database, the 48 past temperature values and 24 hourly predicted values to feed into our Neural Network model.

The predicted energy demand values are finally stored into a MySQL database, for further utilization. This time we use the SQL Python module to connect to another MySQL database

named 'cmcp_energy_demand'. We have created two different tables to store the predicted energy demand levels, one for each location. The tables are named 'EnergydemandMontpelier' and 'EnergydemandVransko' for Montpelier and Vransko respectively. Every row in each of these tables contains 25 values, which correspond to the timestamp of the prediction and the predicted energy demand levels for the next 24 hours.



```
1 • SELECT * FROM cmcp_energy_demand.energy_demand;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

date	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10	e11	e12	e13	e14	e15	e16	e17	e18	e19	e20	e21	e22	e23	e24
2018-04-30 00:00:00	5000	5600	5400	5400	5700	5900	5600	5200	5200	5600	5500	5300	5300	5300	5430	5650	5290	5080	5700	5800	4900	5260	5100	5100
2018-04-30 01:00:00	4800	5400	5600	5560	5340	5200	5300	5340	5920	5400	5500	5200	5430	5670	5860	5120	5370	5400	5500	5200	5300	5200	5630	5300

Figure 5.3.2 Demand prediction table

This entire process must be fully automated, without the need for human intervention at any phase, to achieve maximum efficiency. This is performed through the online scheduler Heroku in which we have assigned the entire pipeline to occur every one hour. In this fashion the energy demand forecasting simulation tool is fully automated from data ingestion to reporting and writing to the database and 'Energydemand' table are updated with new prediction values hourly.

5.4 Energy storage module integration

Energy storage module: access the necessary data

The energy storage module will receive a .csv (or similar type) file, containing the temperature and other values measured from the sensors. This file will be produced with the data that are in the InDeal database.

	IDX	DATUM	ZEIT	DPKT0	DPKT1	DPKT2	DPKT3	DPKT4	DPKT5	DPKT6	...
0	23321205	2014-05-05	08:45	88.7	85.2	88.1	79.2	86.9	77.2	81.4	...
1	23321210	2014-05-05	08:50	95.1	91.4	94.1	84.6	93.6	84.1	87.2	...
2	23321215	2014-05-05	08:55	83.0	79.9	81.7	74.4	80.4	74.1	76.4	...
3	23321220	2014-05-05	09:00	85.9	83.2	85.7	77.5	85.5	77.3	79.4	...
4	23321225	2014-05-05	09:05	91.7	88.7	91.1	82.1	90.8	82.1	83.9	...

Figure 5.4.1 Visualization of the temperature data loaded as a dataframe in Python

Outputs generated and communication

The final output of the heat storage tank model is the temperature of the water that exits the heat tank, and enters the network. A single numerical value is produced every x minutes, where x

is the time frequency the operator requests. This value is uploaded to the InDeal database via the REST API.

5.5. Decision support module integration

In order to proceed with the deployment of the decision-making/optimisation algorithm we need first to integrate the control module with the information from the prediction tools (WFT, EDP) regarding the weather (outside temperature) and energy demand for each of the networks under study (Vransko and Montpellier). This data resides in our server database and act as exogenous inputs to the control/optimization algorithm.

Procedure for control/optimisation:

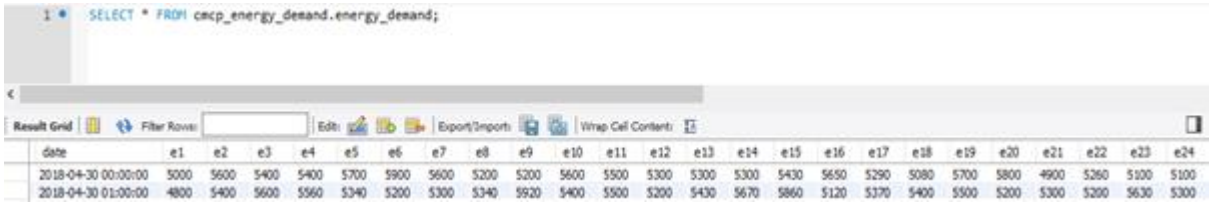
The advanced control module collects the forecast data (for the next 24 h) to make the optimal plant scheduling for the future prediction horizon, i.e. next 24 hours. To make the control outputs/decisions more robust to stochastic uncertainties, the receding horizon control approach (RHC) is embedded in the control algorithm, which simply means that we repeatedly solve a new optimization problem over a moving time horizon i.e. every 1 hour, by using the newly available predictions. This implies that we need to update the forecasts at each sampling time (i.e. typically every 1h) and discard the old ones. More precisely:

1. at time t we receive the current available forecasts for a time interval extending Th=24 samples in the future: t, $t+1$,...,$t+Th$
2. we solve the optimization problem subject to the available (predicted) information and derive the optimal sequence of decision variables that constitute the plant scheduling for the period: t,.., $t+24$
3. then increase the sampling time instant to t+1 and repeat the procedure from (1).

Integration with prediction tools:

The procedure to collect/retrieve the (forecast) energy demand data is achieved via a custom MATLAB script and the main steps are as follows:

- we use the provided ENERGY-API to connect to the SQL database "cmcp_energy_demand" and locate the corresponding tables, one for each location, where the predicted energy demand loads are stored.
- the structured tables are named '*EnergydemandMontpelier*' and '*EnergydemandVransko*' for Montpelier and Vransko networks respectively. Each row in each of these tables represent a sampling time instant with 25 values, i.e. the corresponding timestamp (day/time) of the prediction and the predicted energy demand levels for the next 24 hours.



- make an SQL query for the particular data we wish to extract, i.e. the load demand values for the next 24 hours.

```
$query = "SELECT * FROM EnergydemandMontpelier ";
$query = "SELECT * FROM EnergydemandVransko ";
```

- receive the required data in .json format and translate it into a usable format for further processing.

The overall integration of the control module with the prediction tools and the InDeal CMCP (central monitoring and control platform) is depicted in Fig. xx

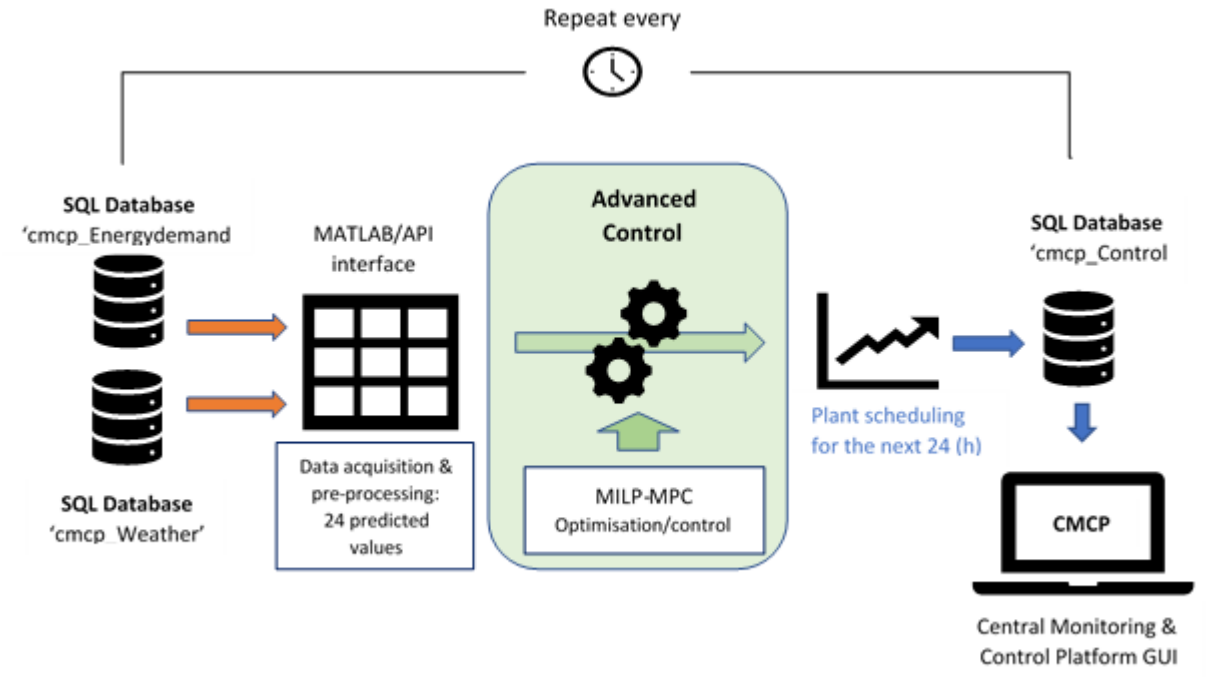


Figure 5.5.1: Integration of advanced control module with CMCP pipeline.

Integration with the CMCP:

The proposed scheduling commands will be transferred to the SQL database "cmcp_control" in a usable format for visualization to the central control platform interface. To achieve that goal, we use a custom MATLAB script that allows us to connect with the database server and store the scheduling data in the corresponding table (one for each location/network).

- First, we establish a connection with the SQL database "cmcp_control" with the credentials (user-name and password) given by NETFI.

Data Source: 'cmcp_control'

User Name: 'city'

URL: 'jdbc:mysql://5.44.241.82: ...'

Type: 'JDBC Connection Object'

Database Product Name: 'MySQL'

Database Product Version: '5.7.20-0ubuntu0.17.04.1'

Driver: 'com.mysql.jdbc.Driver'

Driver Name & version: 'MySQL Connector/J'; 'mysql-connector-java-8.0. ...'

- Next, we locate the structured tables where we are going to store the scheduling data from the control module. As before we use one table per case study/network, i.e. 'Vransko_DH' and 'SERM_DHC' for Vransko and Montpellier networks respectively.

- Next, we export the control data from MATLAB (into the required usable format) and send them to the SQL database to populate the entries of the corresponding tables. Each row represents a sampling time instant which contains the following variables:
    - the timestamp (day/time),
    - the predicted energy demand levels, and
    - the values for the power levels of the corresponding production units.

- Finally, the output data are visualised in the graphical user interface of the central monitoring and control platform (CMCP) in order to assist the network operators.

The (higher-level) control/optimization strategy generates the optimal plant scheduling in terms of plant operations (on/off status of production units and load distribution amongst them) and send them to the control platform for visualization by the network operators.

6. Conclusions

The description of first loop of preliminary design followed the process of understanding test sites user needs, the technical characteristics of the existing infrastructure and users' requests for monitoring and managing of the main network parameters. Based on those and additional specifications as presented in D1.1 System Specifications, the preliminary design of Indeal, a first solution was developed for verification of the operability principles. Moreover, additional points were taken into account such as actual installation in the real test sites. The first loop of preliminary design fully covers the InDeal project targets offering reliable communication between the InDeal hardware and software components allowing at the same time advanced data gathering and processing of measurable DHC network parameters.