Technical Specification Double-side Page

1. TECHNICAL SCOPE

PLATONIC (deeP LeArning neTwOrks for building's eNergy effICiency) develops an occupancy & renewable energy sources (RES)-aware Heating, Ventilation and Air Conditioning (HVAC) recommendation system for commercial buildings. This project was proposed under REACH's "Occupancy-driven monitoring & multi-factor recommendation systems for Energy Efficiency In Buildings" theme.

The challenge was to build a non-intrusive engine, capable of inferring building occupancy based on data collected from the HVAC systems (e.g., energy measurements), as well as indoor/outdoor data points (e.g., temperature, humidity, etc.) to develop optimal, occupancy-informed, heating strategies for HVACs.

PLATONIC tackles this challenge by proposing **4 asynchronous, modular, and reusable building-block components: the occupancy model, the RES model, the energy model, and the HVAC recommendation system (DRL) model.** Each of the building blocks focuses on autonomously providing a rich set of data (e.g., Local Energy Production / Occupancy predictions), which are then inputted to a Deep Renforcement Learning (DRL) Agent, responsible for producing week-ahead optimal schedules for HVACs.

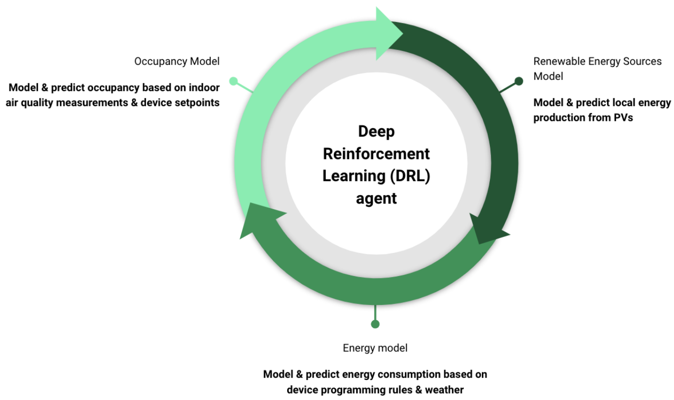

Figure 1. PLATONIC's modular components

Developing a data-based approach, PLATONIC modules are built around the data collected from our data provider, CERTH, as well as our own data and some third party APIs. Data collected during this phase was used to build the prediction models and test the resulting model's performance. Key data features provided by CERTH included on-site generated power from solar panels, HVAC setpoints and energy measurements (e.g., meter & sub-meter data), and indoor (air quality) conditions.

2. ALGORITHMS, TOOLS AND CONCLUSIONS

The **Renewable Energy Sources (RES)** module is based on a Decision Tree-based model. It identifies the relation between weather-related variables (e.g., solar irradiance variables, cloud & visibility, etc.) and local energy production. Localized weather predictions are provided by OpenMeteo's API, while Python's **pvlib** tool provided us with additional data features that improved the model's performance, such as Global Horizontal Irradiance (GHI) and Direct Normal Irradiance (DNI).

The **occupancy module** is based on readily available HVAC data, such as temperature and HVAC setpoints to create a prediction model for indoor CO_2 levels using Tree ensemble methods. Predicted CO_2 levels are then used to infer and predict indoor occupancy levels by using a Regressor Chain method. Google's Popular Time data provided us with the on-site trends and real-time occupancy data used to train our supervised models and evaluate their performance.

The **energy module** uses weather and HVAC programming data to model & predict the energy consumption patterns for a building. Data is modeled using a gradient boosting tree algorithm. Historical meter and sub-meter data is used to train the models and evaluate its performance.

Finally, the HVAC recommendation system model implements a **Deep Reinforcement Learning Algorithm (Q-learning)** to improve HVAC operations based on the data input/output of the previous modules. In this step, we define an environment for our RL agent and a reward function to optimize (i.e., reduce energy consumption) while penalising any recommendations that are outside the user's pre-set comfort range. This process has given us deep insight into HVAC optimization strategies and some early encouraging results:

- As initially expected, gains derived from an AI-powered recommendation system for HVACs **exceed over 40%** when compared to our baseline scenario[1]. However, what is interesting to note is that building's users comfort was also improved (by more than 2% on average) thanks to the introduction of finer constraints on temperature ranges that are not only based on outdoor temperature but on current building occupancy.
- These gains do not depend on a fine-grained calculation of the exact number of people that are in a building at any given point in time, but can rather be based on a less demanding feature (in terms of computing and calculations), which is relative occupancy. This in turn helps improve the flexibility and scalability of the final solution.
- The models proved to be extremely flexible and allowed us to define several other customer use cases in which data from each of the models could be used. For instance, energy consumption forecast data could be used to showcase faulty equipment behaviour or energy leaks in devices. The occupancy module provides precious insight into busiest hours, regardless of building size or structure since is based on environmental variables. This data could then be used to adapt the behaviour of other devices (indoor lighting, etc.), which could in turn increase overall efficiency gains.

[1] Baseline scenarios were built using actual calendars and programming data from HVAC systems, as well as localized weather predictions.

3. SCALABILITY AND FLEXIBILITY OF THE SOLUTION

As previously stated, our goal was to develop optimal, occupancy-informed, heating strategies for HVACs, that could unlock up to 40% in energy savings while preserving user's comfort. The final solution offers an inexpensive and non-intrusive data-based approach to improve HVAC operations, fulfilling all of the requirements stated above. The current TRL level is 7.

PLATONIC's component architecture makes it very flexible: each component outputs autonomously a rich set of data that can be used in other fields or other applications (i.e., occupancy prediction). Moreover, additional developments could be undertaken to enlarge / enrich the project's approach, namely:

- By using occupancy and indoor CO2 data to further develop insights into how to optimally control HVAC fresh air inlet. Additional energy gains can be expected from such operation while preserving indoor safety and comfort.
- By providing our customers with the ability to implement optimal programs with a single click of a button, regardless of underlying technology and/or vendors. This capacity can be easily made available given our solution's focus on interoperability and the back-end developments that have been carried out so far in this sense.

To ensure the scalability of the solution, we have chosen the approach shown in Figure 2. In essence, all developments pertaining to model selection and training have been carried out locally (version control and parallel developments via Gitlab) and persisted using Pickle. The model is then tagged and deployed within our own architecture framework, which contains all of the necessary technical mechanisms to ensure the final's solution performance, flexibility and scalability.

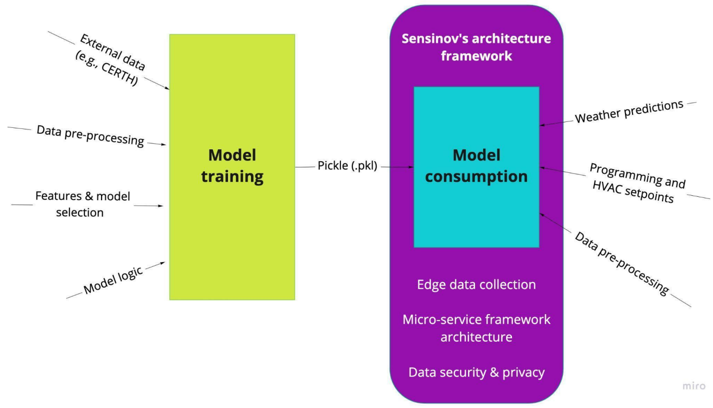

Figure 2. PLATONIC's software architecture

4. DATA GOVERNANCE AND LEGAL COMPLIANCE

As shown in Figure 2, PLATONIC benefits from the data security and privacy mechanisms that are included in our platform. As such, it benefits from a backend web framework (Django) implementing state-of-the-art security practices for protection against all major web security flaws (e.g., XSS, SQL Injections, CSRF, ...). Moreover, we have internal security authorizations for end-users that prevent unauthorized access to sensible data.

It is also important to note that, following the EU's GDPR, no personal, identifiable information will be shared (steps will be taken to ensure that such data is pre-anonymized and handled at a higher level of abstraction).

5. QUALITY ASSURANCE AND RISK MANAGEMENT

Guided by our team's expertise, we have developed a structured project quality and risk management methodology based on the following principles and tools:

- **Defined roles and responsibilities**: clear assignment of responsibilities and roles across team members which in turn improves overall task ownership from participants.
- **Control loops:** complex tasks are split into smaller, more manageable tasks. The latter can be then discussed within focus workgroups, where team members have the space to collaborate efficiently based on their expertise.
- **Manage by stage:** our Checkpoint reports are organized daily, weekly or monthly, depending on project requirements. These checkpoints can be in the form of brief regular meetings or redacted reports, which help us cover all key aspects of the project's evolution, including next-steps, risks, and review of the overall plan.
- **Tailor to the project environment:** our value-oriented approach helps us continuously refocus on the final product/task at hand. Specific tools and methodologies can thus be deployed to fit each project';s specific environment, complexity, time constraints and risks.

Thanks to this approach, all of the issues encountered during the EXPLORE phase were swiftly circumvented, namely for solving:

- **Data-related risks:** The available amount of data was at times not adequate or some of the necessary data features were missing. As planned, the inclusion of our own database allowed us to curtail this risk.
- **High resource consumption:** The initial project scope required us to implement in some cases research-based approaches that necessitated high computing power during training. When optimization methods proved inefficient, we decided to focus on implementing more straightforward Machine Learning (ML) methods that provided a net positive balance between the energy consumed in training and the resulting improvement in building operations and consumption reductions.

Annex 1. Means for accessing the MVP

Here is the link containing a video of the MVP.

The live demo will be accessible on Sept. 27th 2022.

Link: app.sensinov.com
Username: certh@reach.com
Password: j5JqgWsxDAjsb63

If you have any issues accessing these resources, please contact eliana.valles@sensinov.com