Workshop Nº2

Tomás Cárdenas Benítez - 20221020021,
Diego Angelo Ruano Vergara - 20242020278,
Sebastian David Trujillo Vargas - 20242020217,
Arlo Nicolas Ocampo Gallegos 20221020104

1. FIRST FINDINGS

The analysis conducted in Workshop Nº1 provided a comprehensive understanding of the forecasting challenge, highlighting the nature of the data, the constraints imposed by the evaluation process, and the presence of chaotic or highly sensitive behaviors within the system. The project focuses on short-term load and wind-power forecasting across several geographic zones, complemented by a system-wide total forecast. Each zone is represented by a time series of hourly observations that extend over a historical window, accompanied by meteorological data obtained from nearby weather stations. This configuration indicates a multivariate forecasting problem where time, spatial relationships, and exogenous environmental factors play crucial roles in predictive performance.

One of the main findings concerns the complexity introduced by the **Weighted Root Mean Square Error (WRMSE)** metric used for evaluation. This metric assigns significantly higher importance to system-level errors than to zonal errors, implying that a small deviation in the aggregate forecast may outweigh large improvements at the individual zone level. Therefore, the forecasting process must be guided by objectives that balance local precision with global stability. The weighting system also implies that the model's optimization process should be aware of the hierarchical nature of the data and should not treat each series as independent.

The workshop identified several **critical constraints**. First, the raw datasets exhibit missing entries, inconsistent timestamps, and occasional noise from faulty sensors. These issues make preprocessing—such as data cleaning, imputation, and normalization—an essential step rather than an optional refinement. Second, temporal alignment across all data sources is vital, as mismatched timestamps between load and weather data can distort the input-output relationships that the model attempts to learn. Third, the forecasting task requires adherence to temporal integrity, meaning that only data available at the forecast time may be used for prediction. This restriction ensures fairness and prevents data leakage, a common pitfall in time-series modeling.

The **data characteristics** further support the need for robust and adaptive modeling strategies. Hourly data reveal both short-term periodicities (such as daily load cycles) and long-term seasonal trends influenced by temperature and weather variability. Additionally, the multi-zone structure introduces correlations and potential dependencies across spatial locations, as zones may respond similarly to external climatic patterns. Understanding and leveraging these correlations will be essential for improving forecast accuracy, particularly under the weighted evaluation scheme.

Finally, the workshop discussed the influence of **chaos and sensitivity** within the system. Because energy demand and renewable generation are influenced by nonlinear and sometimes unpredictable environmental conditions, even small perturbations in temperature or input data alignment can cause substantial deviations in forecasts. These chaotic elements limit the predictability horizon and introduce uncertainty that cannot be eliminated entirely, only managed. The findings suggest that the forecasting framework must account for such variability by incorporating

mechanisms for robustness and uncertainty estimation, ensuring the system can remain reliable even when exposed to unexpected inputs or incomplete data.

2. SYSTEM REQUIEREMENTS

The system requirements translate the analytical findings from Workshop Nº1 into concrete, measurable objectives that guide the technical design. From a performance perspective, the forecasting framework must achieve a Weighted Root Mean Square Error (WRMSE) at least ten percent lower than the benchmark persistence model. This ensures that both zonal and system-level forecasts deliver tangible improvements over simple baselines. Because the evaluation metric assigns greater weight to system-level accuracy, the models must prioritize minimizing aggregate forecast errors, even if minor trade-offs occur at the zonal level. In addition, performance consistency across time horizons is essential; the error for longer horizons (for example, 48 hours ahead) should not deteriorate by more than a fixed percentage relative to near-term forecasts.

Reliability and availability are equally critical. The forecasting system must consistently generate predictions at each scheduled cycle, maintaining an uptime of at least 99 percent during operational periods. To safeguard against data or model failures, a persistence baseline will serve as an automated fallback mechanism, ensuring continuous service even under abnormal conditions. Data quality management forms another core requirement—time-series inputs must be accurately synchronized across all zones and weather stations, with missing data not exceeding five percent in any 24-hour window. Any larger data gaps or misalignments should trigger automated alerts and initiate conservative imputation strategies.

Operationally, the system should support a weekly retraining schedule or more frequent updates when concept drift is detected in input variables or prediction errors. All models must provide interpretability features, such as global feature importance scores or SHAP visualizations, to allow operators to understand the key drivers behind forecast behavior. From a user-centric perspective, the interface should be intuitive and transparent, displaying both system and zonal forecasts alongside confidence intervals to enhance trust and decision-making. Finally, the architecture must embed strong security principles, with all data encrypted in storage and transmission, access controlled through role-based authentication, and user activities logged for traceability. Collectively, these requirements ensure that the final design is accurate, robust, interpretable, and secure—fully aligned with the analytical insights and operational constraints established in Workshop Nº1.

3. ARCHITECTURE

The proposed architecture is designed to support end-to-end forecasting — from data ingestion to model serving — in a modular and maintainable way. It follows systems-engineering principles of modularity, redundancy, and observability to ensure reliability and adaptability as the system evolves. The architecture begins with:

- *Data Ingestion and Storage layer*, which collects information from multiple sources such as historical load data, weather station readings, and system metadata. These data streams are cleaned, validated, and stored in a time-series database or distributed storage system to guarantee consistency and accessibility across the pipeline.
- *Preprocessing and Alignment module* performs crucial data-cleaning operations, including missing-value imputation, temporal synchronization across zones, normalization, and feature encoding. Because small data errors can significantly affect forecasting accuracy, this stage acts as the first defense against sensitivity and chaotic effects observed in the analysis.
- *Feature Store*, stores the preproccessed data, ensuring that the same feature transformations used during training are consistently applied during inference. This enhances reproducibility and stability across model versions.
- *Modeling Layer* forms the analytical core of the architecture. It houses multiple model families — such as gradient-boosted trees, recurrent networks, or multi-output regressors —

trained either per horizon or jointly for all forecast horizons. An ensemble or stacking strategy integrates the predictions, explicitly optimizing for the Weighted Root Mean Square Error (WRMSE) metric to balance zonal and system-level performance.

- *Postprocessing module follows*, enforcing prediction constraints (such as clipping values to the [0,1] range) and aggregating zonal results into a coherent system-level forecast.
- *Evaluation and Backtesting* component performs rolling-origin cross-validation, computing weighted errors and comparing model performance against baselines, in order to guarantee continuous validation and traceability
- *Monitoring and Operations* unit continuously tracks key indicators — such as WRMSE trends, data drift, and model bias — and automatically triggers alerts or fallbacks when anomalies occur.
- *Serving and Dashboard layer*, helps forecasts be delivered through a which exposes an API for automated consumption and provides a graphical interface for users to visualize forecasts, confidence intervals, and model explanations.

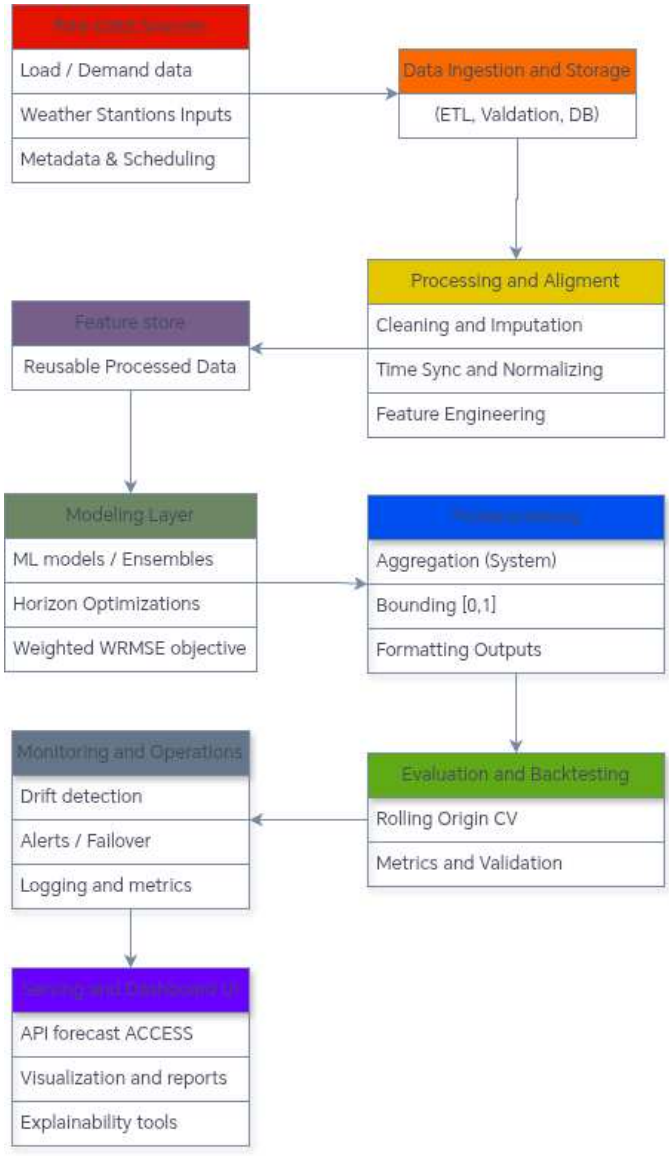

Fig. 1. Architecture grafic

This architecture embodies the core systems-engineering principles of **modularity** (each layer can evolve independently), **robustness** (redundant fallback mechanisms and monitoring loops), and **traceability** (data lineage, feature tracking, and explainability). Its modular design also supports incremental scaling — allowing additional data sources, model variants, or feedback mechanisms to be integrated without disrupting existing operations.

4. SENSIBILITY AND CHAOS

The load forecasting system analyzed in the previous workshop exhibits a high degree of sensitivity and partial chaotic behaviour due to its dependence on environmental and temporal variables. Electricity demand is strongly influenced by temperature, humidity, and human activity patterns; thus, even small deviations in weather data or errors in temporal alignment can produce disproportionately large changes in predicted load curves. This sensitivity extends to model configuration:

changes in feature selection, scaling, or hyperparameter tuning may yield significantly different outcomes across zones and forecast horizons.

To address these issues, the system incorporates multiple stability layers designed under systems engineering principles. At the data level, a validation module continuously checks for missing or corrupted temperature and load values, using interpolation and anomaly detection to prevent noise propagation. Feature engineering steps are standardized and version-controlled to ensure deterministic transformations across forecasting cycles. At the modeling level, ensemble and regularized regression methods are employed to reduce the variance produced by random initialization and prevent overfitting to local fluctuations. Moreover, the system applies rolling-origin validation to mimic real operational forecasting and detect when model behaviour becomes unstable under new weather regimes.

Chaotic effects also emerge from nonlinear dependencies between meteorological inputs and electricity consumption. For example, a sudden cold front in one region may increase heating demand while simultaneously decreasing air-conditioning loads elsewhere, creating contradictory feedbacks that destabilize aggregate predictions. To manage these chaotic responses, the system integrates monitoring routines that track prediction drift and trigger retraining when performance metrics (e.g., WRMSE) exceed predefined thresholds. In extreme cases, a rollback mechanism restores the last stable model configuration to maintain operational reliability.

- **Data control:** automatic detection and correction of anomalies in temperature and load datasets.
- **Model robustness:** ensemble forecasting and regularization to reduce variance under input perturbations.
- **Feedback monitoring:** continuous evaluation of prediction errors and retraining triggers based on performance drift.
- **Operational resilience:** rollback and alert systems to prevent chaotic propagation of faulty forecasts.

By combining these measures, the proposed design enhances stability and predictability in the inherently sensitive and dynamic context of electrical load forecasting. This approach transforms a potentially chaotic process into a controllable, monitored, and resilient forecasting pipeline capable of adapting to environmental and behavioural variability.

5. TECHNICAL STACK AND IMPLEMENTATION SKETCH

The proposed system will be implemented using an accessible and efficient set of tools that support data analysis and predictive modeling. The goal is to build a functional and interpretable forecasting system.

Programming Language: Python is chosen as the main programming language because it is widely used for data science and machine learning. It provides a clear syntax and a large number of libraries for data processing, visualization, and model training.

Main Libraries and Tools: leftmargin=*

- **pandas** and **NumPy** will be used for cleaning, transforming, and organizing time-series data such as temperature and electricity load.
- **scikit-learn** will support model creation and evaluation, including regression models and cross-validation.
- **matplotlib** and **seaborn** will generate plots that help visualize correlations, forecasts, and performance metrics.
- **XGBoost** or **RandomForestRegressor** may be used for the final predictive model, since they handle nonlinear relationships effectively.

Data Handling: The datasets provided in the competition (load history, temperature history, and benchmark data) could be stored locally in CSV format. Each execution of the program will read

and process these files, ensuring that missing or inconsistent values are detected and corrected before training.

Model Training and Evaluation: The model will be trained and tested in **Jupyter Notebook**, which allows step-by-step experimentation and visualization of results. The evaluation metric will be the Weighted Root Mean Square Error (WRMSE), the same used in the Kaggle competition. This ensures consistency between the academic design and the real-world evaluation standard.

Implementation Plan: leftmargin=*

1) Load and explore the data to understand its structure.
2) Preprocess the data by creating useful variables (lags, averages, or time-based features).
3) Train and validate the predictive model using different algorithms from scikit-learn.
4) Compare model performance with the benchmark and select the one that achieves the lowest WRMSE.
5) Visualize results and export predictions in the required submission format.

References

leftmargin=*

- pandas: `https://pandas.pydata.org/`
- NumPy: `https://numpy.org/`
- scikit-learn: `https://scikit-learn.org/`
- matplotlib: `https://matplotlib.org/`
- seaborn: `https://seaborn.pydata.org/`
- XGBoost: `https://xgboost.readthedocs.io/`
- Jupyter Notebook: `https://jupyter.org/`
- Streamlit: `https://streamlit.io/`
- Dash: `https://dash.plotly.com/`