Project Summary: EV Truck Charging
Phase 1

Frequenz EaaS

2. September 2025

Inhaltsverzeichnis

1 Introduction

The transition to electric vehicle (EV) fleets, particularly for heavy-duty trucks, presents a significant challenge for the energy infrastructure. The high power demand for charging can strain local grids, leading to instability and increased costs. To address this, Siemens Energy and Frequenz EaaS initiated the "EV Truck Charging" project to develop a solution for intelligent on-site energy management.

Phase 1 of this project, the subject of this report, focused on the design and development of a foundational software application. The key objectives for this phase were:

- **High-Level Application Design:** Create an extendable and robust software architecture for optimizing energy flows within a microgrid.
- **On-Site Measurement Focus:** Integrate with simulated on-site hardware, including truck chargers, batteries, PV installation, and the public grid, to collect real-time energy data.
- **Edge Control Logic:** Implement a local control system capable of making real-time decisions to ensure reliable and efficient operations.
- **System Transparency:** Provide a live dashboard for monitoring energy flows and system status.
- **Knowledge Transfer:** Offer ongoing SDK training to Siemens Energy staff to ensure long-term project success and maintainability.

This report details the architecture, implementation, and outcomes of Phase 1, demonstrating the successful achievement of these objectives.

2 System Architecture and Design

The application is built upon the Frequenz SDK, which provides a robust framework for developing energy management applications. The architecture is based on the Actor model, a paradigm that facilitates the development of concurrent and distributed systems. Each component of the system is an 'Actor' that runs independently and communicates with other actors through asynchronous message passing. This design ensures high availability and scalability.

The core of the application is the `TruckChargingActor`, which serves as the central orchestrator. It is responsible for:

- Subscribing to real-time data streams from various microgrid components.
- Processing the incoming data and maintaining the system's state.
- Invoking the control logic to make decisions based on the current state.
- Optionally reporting key metrics to a time-series database for monitoring and visualization.

This modular, actor-based architecture makes the system highly extensible, allowing for new functionalities and control strategies to be added with minimal changes to the existing codebase.

2.1 Sensor & Meter Integration

A key outcome of Phase 1 was the successful integration with simulated on-site energy measurement devices. The application leverages the Frequenz SDK to establish connections to various microgrid components and subscribe to their real-time data streams.

The `TruckChargingActor` initializes and manages subscriptions to the following data sources:

- **Grid Power:** The main power feed from the public grid, obtained via `microgrid.grid().power`.

- **Battery Pool:** The aggregated power and state of charge (SoC) of the on-site batteries, managed through `microgrid.new_battery_pool()`.
- **EV Charger Pool:** The combined power consumption of all truck chargers, accessed via `microgrid.new_ev_c`
- **Producer Power:** The power generated by on-site assets like solar panels, available through `microgrid.producer().power`.

The actor uses an asynchronous `select` statement to efficiently listen to all these data streams simultaneously. When a new measurement is received, the actor updates its internal state, ensuring that the control logic always operates on the most recent data. This real-time data collection is the foundation of the intelligent decision-making capabilities of the system.

2.2 Edge Control Implementation

The core of the application's intelligence lies in the `TcControlLogic` class, which implements the edge control logic. This component is responsible for making real-time decisions to maintain grid stability and optimize energy usage. The control logic is executed whenever a new grid power measurement is received.

The primary goal of the control logic is to keep the grid power consumption below a predefined target (e.g., 2.5 MW). The logic follows these steps:

1. **Check for Over-consumption:** If the current grid power exceeds the target, the system takes corrective action.
    - If the battery has sufficient charge (SoC $> 10\%$), the battery is instructed to discharge to cover the excess power demand.
    - If the battery is depleted or unavailable, the power supply to the truck chargers is curtailed to bring the grid power back within the target.
2. **Check for Excess Power:** If the grid power is negative (i.e., power is being exported), it indicates an excess of on-site generation. In this case, the battery is instructed to charge using this surplus power.
3. **No Action:** If the grid power is within the target range, no control action is taken and truck charging is not throttled.

The following code snippet from the `TcControlLogic.perform` method illustrates the logic for handling over-consumption:

```
if latest_grid_power > self._target_power:
    if latest_battery_soc > self._min_soc:
        restrict_power = latest_grid_power - self._target_power
        battery_discharge_power = max(
            -restrict_power, latest_battery_max_discharge_power
        )

        ev_restriction_power = max(
            restrict_power + latest_battery_max_discharge_power, Power.zero()
        )
        ev_charge_power = max(
            latest_ev_charger_max_power - ev_restriction_power, Power.zero()
        )

        # Discharge Battery & Restrict EV
        await self._battery_pool.propose_power(battery_discharge_power)
        await self._ev_charger_pool.propose_power(ev_charge_power)
    else:
        # Restrict EV Power only
        ev_restriction_power = max(
            latest_ev_charger_max_power
            - (latest_grid_power - self._target_power),
            Power.zero(),
        )
        await self._ev_charger_pool.propose_power(ev_restriction_power)
```

Listing 1: Control logic for handling grid congestion.

This logic ensures that the microgrid operates efficiently and avoids high charges for peak power consumption from the grid.

2.3 Energy Flow Visualization

To provide transparency and enable real-time monitoring of the microgrid's performance, the application includes a component for energy flow visualization. This is achieved through the integration of InfluxDB, a time-series database, and Grafana, a popular data visualization tool.

The `InfluxReporter` class is responsible for collecting key metrics from the application and reporting them to an InfluxDB instance. The following metrics are reported:

- Grid Power (watts)
- Battery Power (watts)
- Battery State of Charge (SoC)
- EV Charger Power (watts)
- Production Power (watts)

These metrics are timestamped and stored in the database, creating a historical record of the system's operation. A pre-configured Grafana dashboard connects to this database to provide live visualizations of these metrics. This dashboard allows operators to monitor the health of the system, understand energy flow patterns, and verify that the control logic is performing as expected.

The repository's `README.md` file contains detailed instructions for setting up the Grafana and InfluxDB services, allowing for easy deployment of the monitoring solution.

3 Testing and Validation

The project scope for Phase 1 emphasized the development of an extendable application with a foundation for robust software unit and integration tests. The modular architecture, with its clear separation of concerns between the main actor, control logic, and data reporting, is designed to be easily testable.

Comprehensive unit tests for the `TcControlLogic` have been implemented and can be found in `tests/test_ev_charging_main.py`. These tests cover a variety of scenarios, including:

- Situations where no control action is needed because the grid power is within the target range.
- Cases where the battery is discharged and EV charging is restricted due to high grid power.
- Scenarios where only EV charging is restricted because the grid power is high, but the battery state of charge is too low.
- Instances where the battery is charged with excess power when the grid power is negative.

These tests ensure the correctness of the control logic and validate its behavior under different conditions. The repository also includes an example of a more advanced, system-level test case, `TestDynamicConditionOnGrid`, which is designed to run in a Hardware-in-the-Loop (HiL) simulation environment. This demonstrates the application's capability to integrate into a larger testing and simulation environment.

4 Business Value

The EV Truck Charging Microgrid application, delivered in Phase 1, provides significant business value to Siemens Energy by addressing key operational and financial challenges associated with large-scale EV charging.

The primary business benefits include:

- **Improved Operational Efficiency:** The automated control logic ensures that the microgrid operates optimally without the need for manual intervention. The system intelligently balances power supply and demand, ensuring that EV chargers are available when needed while protecting the local grid from excessive strain.
- **Reduced Energy Costs:** By actively managing the power flow, the application helps to reduce energy costs. It minimizes consumption from the grid during peak hours by utilizing on-site battery storage and can avoid high demand charges by curtailing non-essential loads when necessary.
- **Enhanced Reliability and Stability:** The edge control logic provides a fast-acting, localized response to changing conditions, which enhances the reliability and stability of the charging infrastructure. This is crucial for ensuring that the EV truck fleet remains operational.
- **Future-Proof, Scalable Solution:** The extendable architecture allows for the integration of new assets, more complex control strategies, and the participation in energy markets planned for this project's phase, making it a valuable long-term investment.

5 Conclusion and Next Steps

Phase 1 of the "EV Truck Charging" project has successfully delivered a foundational application for on-site energy optimization. The project achieved all its primary objectives, including the integration of on-site sensors, the implementation of a real-time edge control system, and the delivery of a live visualization dashboard. The resulting application is robust, extendable, and provides immediate business value by improving efficiency and reducing costs.

The following next steps are recommended for future project phases:

- **Enhance Control Strategies:** Implement more sophisticated intelligent control algorithms that can take into account factors like energy prices, demand response signals, and weather forecasts to further optimize energy usage.
- **Deployment and Field Testing:** Deploy the application in a live production environment to gather real-world performance data and refine the control logic based on operational experience.

The successful completion of Phase 1 has laid a strong foundation for the continued development and deployment of this innovative energy management solution.